Methods in R

A topic reference for biomedical research and peer review

R. Heller

2026-11-14

Contents

Welcome! **193**
- Open Source Repository . . . 193
- How To Use The Guide . . . 194
- Contributing . . . 194
- Citation . . . 194

Impressum **195**
- Suggested citation . . . 195
- Disclaimer . . . 195
- Contact . . . 195

Acknowledgments **197**
- Inspiration . . . 197
- Use of LLM tools . . . 198

Notation **199**
- Greek symbols . . . 199
- Latin symbols . . . 199
- Operators and notation . . . 200
- Abbreviations . . . 200
- Code conventions . . . 201

About the author **203**
- Sibling volume . . . 203

How to use this book **205**
- Three reading modes . . . 205
- Conventions . . . 205
- Shared reviewer checklist . . . 206
- What this book is not . . . 206
- Building locally . . . 206

Find your method **209**
- The static decision tree . . . 209

Quick lookup 209

1 Statistical Foundations **213**
1.1 Method pages 213
1.2 Labs 214
1.3 Bias and Variance of Estimators 214
1.3.1 Introduction 214
1.3.2 Prerequisites 214
1.3.3 Theory 215
1.3.4 Assumptions 215
1.3.5 R Implementation 215
1.3.6 Output & Results 216
1.3.7 Interpretation 216
1.3.8 Practical Tips 216
1.3.9 For Reviewers 217
1.3.10 See also — labs in this chapter 217
1.4 Big-Op and Little-op Notation 217
1.4.1 Introduction 217
1.4.2 Prerequisites 217
1.4.3 Theory 217
1.4.4 Assumptions 218
1.4.5 R Implementation 218
1.4.6 Output & Results 219
1.4.7 Interpretation 219
1.4.8 Practical Tips 219
1.4.9 For Reviewers 219
1.4.10 See also — labs in this chapter 219
1.5 The Cauchy-Schwarz Inequality 220
1.5.1 Introduction 220
1.5.2 Prerequisites 220
1.5.3 Theory 220
1.5.4 Assumptions 221
1.5.5 R Implementation 221
1.5.6 Output & Results 221
1.5.7 Interpretation 222
1.5.8 Practical Tips 222
1.5.9 For Reviewers 222
1.5.10 See also — labs in this chapter 222
1.6 The Central Limit Theorem 222
1.6.1 Introduction 222
1.6.2 Prerequisites 223
1.6.3 Theory 223
1.6.4 Assumptions 223
1.6.5 R Implementation 223
1.6.6 Output & Results 224
1.6.7 Interpretation 224

1.6.8 Practical Tips . . . 225
1.6.9 For Reviewers . . . 225
1.6.10 See also — labs in this chapter . . . 225
1.7 Characteristic Functions . . . 225
1.7.1 Introduction . . . 225
1.7.2 Prerequisites . . . 226
1.7.3 Theory . . . 226
1.7.4 Assumptions . . . 226
1.7.5 R Implementation . . . 226
1.7.6 Output & Results . . . 227
1.7.7 Interpretation . . . 227
1.7.8 Practical Tips . . . 227
1.7.9 For Reviewers . . . 228
1.7.10 See also — labs in this chapter . . . 228
1.8 Markov and Chebyshev Inequalities . . . 228
1.8.1 Introduction . . . 228
1.8.2 Prerequisites . . . 228
1.8.3 Theory . . . 228
1.8.4 Assumptions . . . 229
1.8.5 R Implementation . . . 229
1.8.6 Output & Results . . . 230
1.8.7 Interpretation . . . 230
1.8.8 Practical Tips . . . 230
1.8.9 For Reviewers . . . 231
1.8.10 See also — labs in this chapter . . . 231
1.9 Confidence Intervals: Introduction . . . 231
1.9.1 Introduction . . . 231
1.9.2 Prerequisites . . . 231
1.9.3 Theory . . . 231
1.9.4 Assumptions . . . 232
1.9.5 R Implementation . . . 232
1.9.6 Output & Results . . . 233
1.9.7 Interpretation . . . 233
1.9.8 Practical Tips . . . 233
1.9.9 For Reviewers . . . 233
1.9.10 See also — labs in this chapter . . . 233
1.10 Convergence in Distribution . . . 234
1.10.1 Introduction . . . 234
1.10.2 Prerequisites . . . 234
1.10.3 Theory . . . 234
1.10.4 Assumptions . . . 235
1.10.5 R Implementation . . . 235
1.10.6 Output & Results . . . 235
1.10.7 Interpretation . . . 235
1.10.8 Practical Tips . . . 236
1.10.9 For Reviewers . . . 236

1.10.10 See also — labs in this chapter 236
1.11 Convergence in Probability 236
1.11.1 Introduction 236
1.11.2 Prerequisites 236
1.11.3 Theory 237
1.11.4 Assumptions 237
1.11.5 R Implementation 237
1.11.6 Output & Results 238
1.11.7 Interpretation 238
1.11.8 Practical Tips 238
1.11.9 For Reviewers 239
1.11.10 See also — labs in this chapter 239
1.12 The Cramer-Rao Lower Bound 239
1.12.1 Introduction 239
1.12.2 Prerequisites 239
1.12.3 Theory 239
1.12.4 Assumptions 240
1.12.5 R Implementation 240
1.12.6 Output & Results 241
1.12.7 Interpretation 241
1.12.8 Practical Tips 241
1.12.9 For Reviewers 242
1.12.10 See also — labs in this chapter 242
1.13 The Delta Method 242
1.13.1 Introduction 242
1.13.2 Prerequisites 242
1.13.3 Theory 242
1.13.4 Assumptions 243
1.13.5 R Implementation 243
1.13.6 Output & Results 244
1.13.7 Interpretation 244
1.13.8 Practical Tips 244
1.13.9 For Reviewers 244
1.13.10 See also — labs in this chapter 245
1.14 The Empirical Distribution Function 245
1.14.1 Introduction 245
1.14.2 Prerequisites 245
1.14.3 Theory 245
1.14.4 Assumptions 246
1.14.5 R Implementation 246
1.14.6 Output & Results 247
1.14.7 Interpretation 247
1.14.8 Practical Tips 247
1.14.9 For Reviewers 247
1.14.10 See also — labs in this chapter 247
1.15 Estimators and Estimation 248

1.15.1 Introduction . . . 248
1.15.2 Prerequisites . . . 248
1.15.3 Theory . . . 248
1.15.4 Assumptions . . . 248
1.15.5 R Implementation . . . 249
1.15.6 Output & Results . . . 249
1.15.7 Interpretation . . . 249
1.15.8 Practical Tips . . . 249
1.15.9 For Reviewers . . . 250
1.15.10 See also — labs in this chapter . . . 250
1.16 Fisher Information . . . 250
1.16.1 Introduction . . . 250
1.16.2 Prerequisites . . . 250
1.16.3 Theory . . . 250
1.16.4 Assumptions . . . 251
1.16.5 R Implementation . . . 251
1.16.6 Output & Results . . . 252
1.16.7 Interpretation . . . 252
1.16.8 Practical Tips . . . 253
1.16.9 For Reviewers . . . 253
1.16.10 See also — labs in this chapter . . . 253
1.17 The Glivenko-Cantelli Theorem . . . 253
1.17.1 Introduction . . . 253
1.17.2 Prerequisites . . . 253
1.17.3 Theory . . . 254
1.17.4 Assumptions . . . 254
1.17.5 R Implementation . . . 254
1.17.6 Output & Results . . . 255
1.17.7 Interpretation . . . 255
1.17.8 Practical Tips . . . 255
1.17.9 For Reviewers . . . 256
1.17.10 See also — labs in this chapter . . . 256
1.18 Jensen's Inequality . . . 256
1.18.1 Introduction . . . 256
1.18.2 Prerequisites . . . 256
1.18.3 Theory . . . 256
1.18.4 Assumptions . . . 257
1.18.5 R Implementation . . . 257
1.18.6 Output & Results . . . 257
1.18.7 Interpretation . . . 258
1.18.8 Practical Tips . . . 258
1.18.9 For Reviewers . . . 258
1.18.10 See also — labs in this chapter . . . 258
1.19 The Law of Large Numbers . . . 259
1.19.1 Introduction . . . 259
1.19.2 Prerequisites . . . 259

1.19.3 Theory . . . 259
1.19.4 Assumptions . . . 260
1.19.5 R Implementation . . . 260
1.19.6 Output & Results . . . 260
1.19.7 Interpretation . . . 260
1.19.8 Practical Tips . . . 261
1.19.9 For Reviewers . . . 261
1.19.10 See also — labs in this chapter . . . 261
1.20 Likelihood Ratio Tests . . . 261
1.20.1 Introduction . . . 261
1.20.2 Prerequisites . . . 261
1.20.3 Theory . . . 262
1.20.4 Assumptions . . . 262
1.20.5 R Implementation . . . 262
1.20.6 Output & Results . . . 263
1.20.7 Interpretation . . . 263
1.20.8 Practical Tips . . . 263
1.20.9 For Reviewers . . . 264
1.20.10 See also — labs in this chapter . . . 264
1.21 Maximum Likelihood Estimation . . . 264
1.21.1 Introduction . . . 264
1.21.2 Prerequisites . . . 264
1.21.3 Theory . . . 264
1.21.4 Assumptions . . . 265
1.21.5 R Implementation . . . 265
1.21.6 Output & Results . . . 266
1.21.7 Interpretation . . . 266
1.21.8 Practical Tips . . . 266
1.21.9 For Reviewers . . . 266
1.21.10 See also — labs in this chapter . . . 267
1.22 The Method of Moments . . . 267
1.22.1 Introduction . . . 267
1.22.2 Prerequisites . . . 267
1.22.3 Theory . . . 267
1.22.4 Assumptions . . . 268
1.22.5 R Implementation . . . 268
1.22.6 Output & Results . . . 268
1.22.7 Interpretation . . . 268
1.22.8 Practical Tips . . . 269
1.22.9 For Reviewers . . . 269
1.22.10 See also — labs in this chapter . . . 269
1.23 Moments and Moment Generating Functions . . . 269
1.23.1 Introduction . . . 269
1.23.2 Prerequisites . . . 270
1.23.3 Theory . . . 270
1.23.4 Assumptions . . . 270

1.23.5 R Implementation . . . 270
1.23.6 Output & Results . . . 271
1.23.7 Interpretation . . . 271
1.23.8 Practical Tips . . . 272
1.23.9 For Reviewers . . . 272
1.23.10 See also — labs in this chapter . . . 272
1.24 Order Statistics . . . 272
1.24.1 Introduction . . . 272
1.24.2 Prerequisites . . . 272
1.24.3 Theory . . . 273
1.24.4 Assumptions . . . 273
1.24.5 R Implementation . . . 273
1.24.6 Output & Results . . . 274
1.24.7 Interpretation . . . 274
1.24.8 Practical Tips . . . 274
1.24.9 For Reviewers . . . 275
1.24.10 See also — labs in this chapter . . . 275
1.25 Pivotal Quantities . . . 275
1.25.1 Introduction . . . 275
1.25.2 Prerequisites . . . 275
1.25.3 Theory . . . 275
1.25.4 Assumptions . . . 276
1.25.5 R Implementation . . . 276
1.25.6 Output & Results . . . 276
1.25.7 Interpretation . . . 277
1.25.8 Practical Tips . . . 277
1.25.9 For Reviewers . . . 277
1.25.10 See also — labs in this chapter . . . 277
1.26 Population vs. Sample . . . 277
1.26.1 Introduction . . . 277
1.26.2 Prerequisites . . . 278
1.26.3 Theory . . . 278
1.26.4 Assumptions . . . 278
1.26.5 R Implementation . . . 279
1.26.6 Output & Results . . . 279
1.26.7 Interpretation . . . 279
1.26.8 Practical Tips . . . 280
1.26.9 For Reviewers . . . 280
1.26.10 See also — labs in this chapter . . . 280
1.27 Sampling Distributions . . . 280
1.27.1 Introduction . . . 280
1.27.2 Prerequisites . . . 280
1.27.3 Theory . . . 281
1.27.4 Assumptions . . . 281
1.27.5 R Implementation . . . 281
1.27.6 Output & Results . . . 282

1.27.7 Interpretation . . . 282
1.27.8 Practical Tips . . . 282
1.27.9 For Reviewers . . . 283
1.27.10 See also — labs in this chapter . . . 283
1.28 Sampling Methods . . . 283
1.28.1 Introduction . . . 283
1.28.2 Prerequisites . . . 283
1.28.3 Theory . . . 283
1.28.4 Assumptions . . . 284
1.28.5 R Implementation . . . 284
1.28.6 Output & Results . . . 285
1.28.7 Interpretation . . . 285
1.28.8 Practical Tips . . . 285
1.28.9 For Reviewers . . . 286
1.28.10 See also — labs in this chapter . . . 286
1.29 Scales of Measurement . . . 286
1.29.1 Introduction . . . 286
1.29.2 Prerequisites . . . 286
1.29.3 Theory . . . 286
1.29.4 Assumptions . . . 287
1.29.5 R Implementation . . . 287
1.29.6 Output & Results . . . 287
1.29.7 Interpretation . . . 288
1.29.8 Practical Tips . . . 288
1.29.9 For Reviewers . . . 288
1.29.10 See also — labs in this chapter . . . 288
1.30 Slutsky's Theorem . . . 289
1.30.1 Introduction . . . 289
1.30.2 Prerequisites . . . 289
1.30.3 Theory . . . 289
1.30.4 Assumptions . . . 290
1.30.5 R Implementation . . . 290
1.30.6 Output & Results . . . 290
1.30.7 Interpretation . . . 290
1.30.8 Practical Tips . . . 291
1.30.9 For Reviewers . . . 291
1.30.10 See also — labs in this chapter . . . 291
1.31 The Standard Error . . . 291
1.31.1 Introduction . . . 291
1.31.2 Prerequisites . . . 291
1.31.3 Theory . . . 292
1.31.4 Assumptions . . . 292
1.31.5 R Implementation . . . 292
1.31.6 Output & Results . . . 293
1.31.7 Interpretation . . . 293
1.31.8 Practical Tips . . . 293

1.31.9 For Reviewers . . . 294
1.31.10 See also — labs in this chapter . . . 294
1.32 Sufficient Statistics . . . 294
1.32.1 Introduction . . . 294
1.32.2 Prerequisites . . . 294
1.32.3 Theory . . . 294
1.32.4 Assumptions . . . 295
1.32.5 R Implementation . . . 295
1.32.6 Output & Results . . . 296
1.32.7 Interpretation . . . 296
1.32.8 Practical Tips . . . 296
1.32.9 For Reviewers . . . 296
1.32.10 See also — labs in this chapter . . . 297
1.33 Unbiasedness, Consistency, Efficiency . . . 297
1.33.1 Introduction . . . 297
1.33.2 Prerequisites . . . 297
1.33.3 Theory . . . 297
1.33.4 Assumptions . . . 298
1.33.5 R Implementation . . . 298
1.33.6 Output & Results . . . 298
1.33.7 Interpretation . . . 299
1.33.8 Practical Tips . . . 299
1.33.9 For Reviewers . . . 299
1.33.10 See also — labs in this chapter . . . 299
1.34 Week 1, Session 3 — Data types, tidy data, accuracy and precision . . . 300
1.34.1 Learning objectives . . . 300
1.34.2 Prerequisites . . . 300
1.34.3 Background . . . 300
1.34.4 Setup . . . 300
1.34.5 1. Goal . . . 301
1.34.6 2. Approach . . . 301
1.34.7 3. Execution . . . 301
1.34.8 4. Check . . . 302
1.34.9 5. Report . . . 302
1.34.10 Common pitfalls . . . 303
1.34.11 Further reading . . . 303
1.34.12 Session info . . . 303
1.34.13 See also — chapter index . . . 303
1.35 Week 1, Session 4 — Import, joins, and missingness with dplyr . . . 303
1.35.1 Learning objectives . . . 303
1.35.2 Prerequisites . . . 304
1.35.3 Background . . . 304
1.35.4 Setup . . . 304
1.35.5 1. Goal . . . 304

1.35.6 2. Approach . . . 304
1.35.7 3. Execution . . . 305
1.35.8 4. Check . . . 306
1.35.9 5. Report . . . 306
1.35.10 Common pitfalls . . . 306
1.35.11 Further reading . . . 307
1.35.12 Session info . . . 307
1.35.13 See also — chapter index . . . 307
1.36 Week 1, Session 2 — R, RStudio, Quarto, and renv toolchain 307
1.36.1 Learning objectives . . . 307
1.36.2 Prerequisites . . . 307
1.36.3 Background . . . 307
1.36.4 Setup . . . 308
1.36.5 1. Goal . . . 308
1.36.6 2. Approach . . . 308
1.36.7 3. Execution . . . 308
1.36.8 4. Check . . . 309
1.36.9 5. Report . . . 310
1.36.10 Common pitfalls . . . 310
1.36.11 Further reading . . . 310
1.36.12 Session info . . . 310
1.36.13 See also — chapter index . . . 310
1.37 Week 1, Session 1 — Scientific process and research workflow 310
1.37.1 Learning objectives . . . 311
1.37.2 Prerequisites . . . 311
1.37.3 Background . . . 311
1.37.4 Setup . . . 311
1.37.5 1. Goal . . . 311
1.37.6 2. Approach . . . 311
1.37.7 3. Execution . . . 312
1.37.8 4. Check . . . 312
1.37.9 5. Report . . . 312
1.37.10 Common pitfalls . . . 313
1.37.11 Further reading . . . 313
1.37.12 Session info . . . 313
1.37.13 See also — chapter index . . . 313

2 Descriptive Statistics 315
2.1 Method pages . . . 315
2.2 Labs . . . 316
2.3 Coefficient of Variation . . . 316
2.3.1 Introduction . . . 316
2.3.2 Prerequisites . . . 316
2.3.3 Theory . . . 316
2.3.4 Assumptions . . . 317
2.3.5 R Implementation . . . 317

2.3.6 Output & Results . . . 317
2.3.7 Interpretation . . . 317
2.3.8 Practical Tips . . . 317
2.3.9 For Reviewers . . . 318
2.3.10 See also — labs in this chapter . . . 318
2.4 Contingency Tables . . . 318
2.4.1 Introduction . . . 318
2.4.2 Prerequisites . . . 318
2.4.3 Theory . . . 318
2.4.4 Assumptions . . . 319
2.4.5 R Implementation . . . 319
2.4.6 Output & Results . . . 319
2.4.7 Interpretation . . . 320
2.4.8 Practical Tips . . . 320
2.4.9 For Reviewers . . . 320
2.4.10 See also — labs in this chapter . . . 320
2.5 Cross-Tabulations . . . 320
2.5.1 Introduction . . . 320
2.5.2 Prerequisites . . . 320
2.5.3 Theory . . . 321
2.5.4 Assumptions . . . 321
2.5.5 R Implementation . . . 321
2.5.6 Output & Results . . . 322
2.5.7 Interpretation . . . 322
2.5.8 Practical Tips . . . 322
2.5.9 For Reviewers . . . 322
2.5.10 See also — labs in this chapter . . . 323
2.6 Descriptive vs Inferential Statistics . . . 323
2.6.1 Introduction . . . 323
2.6.2 Prerequisites . . . 323
2.6.3 Theory . . . 323
2.6.4 Assumptions . . . 323
2.6.5 R Implementation . . . 324
2.6.6 Output & Results . . . 324
2.6.7 Interpretation . . . 324
2.6.8 Practical Tips . . . 324
2.6.9 For Reviewers . . . 325
2.6.10 See also — labs in this chapter . . . 325
2.7 Five-Number Summary . . . 325
2.7.1 Introduction . . . 325
2.7.2 Prerequisites . . . 325
2.7.3 Theory . . . 325
2.7.4 Assumptions . . . 326
2.7.5 R Implementation . . . 326
2.7.6 Output & Results . . . 326
2.7.7 Interpretation . . . 327

2.7.8 Practical Tips . . . 327
2.7.9 For Reviewers . . . 327
2.7.10 See also — labs in this chapter . . . 327
2.8 Frequency Tables . . . 327
2.8.1 Introduction . . . 327
2.8.2 Prerequisites . . . 328
2.8.3 Theory . . . 328
2.8.4 Assumptions . . . 328
2.8.5 R Implementation . . . 328
2.8.6 Output & Results . . . 329
2.8.7 Interpretation . . . 329
2.8.8 Practical Tips . . . 329
2.8.9 For Reviewers . . . 330
2.8.10 See also — labs in this chapter . . . 330
2.9 The Geometric Mean . . . 330
2.9.1 Introduction . . . 330
2.9.2 Prerequisites . . . 330
2.9.3 Theory . . . 330
2.9.4 Assumptions . . . 330
2.9.5 R Implementation . . . 331
2.9.6 Output & Results . . . 331
2.9.7 Interpretation . . . 331
2.9.8 Practical Tips . . . 331
2.9.9 For Reviewers . . . 332
2.9.10 See also — labs in this chapter . . . 332
2.10 The Harmonic Mean . . . 332
2.10.1 Introduction . . . 332
2.10.2 Prerequisites . . . 332
2.10.3 Theory . . . 332
2.10.4 Assumptions . . . 333
2.10.5 R Implementation . . . 333
2.10.6 Output & Results . . . 333
2.10.7 Interpretation . . . 333
2.10.8 Practical Tips . . . 334
2.10.9 For Reviewers . . . 334
2.10.10 See also — labs in this chapter . . . 334
2.11 The Interquartile Range . . . 334
2.11.1 Introduction . . . 334
2.11.2 Prerequisites . . . 334
2.11.3 Theory . . . 334
2.11.4 Assumptions . . . 335
2.11.5 R Implementation . . . 335
2.11.6 Output & Results . . . 335
2.11.7 Interpretation . . . 336
2.11.8 Practical Tips . . . 336
2.11.9 For Reviewers . . . 336

2.11.10 See also — labs in this chapter . . . 336
2.12 Kurtosis . . . 336
2.12.1 Introduction . . . 336
2.12.2 Prerequisites . . . 337
2.12.3 Theory . . . 337
2.12.4 Assumptions . . . 337
2.12.5 R Implementation . . . 337
2.12.6 Output & Results . . . 338
2.12.7 Interpretation . . . 338
2.12.8 Practical Tips . . . 338
2.12.9 For Reviewers . . . 338
2.12.10 See also — labs in this chapter . . . 338
2.13 Mean Absolute Deviation . . . 338
2.13.1 Introduction . . . 338
2.13.2 Prerequisites . . . 339
2.13.3 Theory . . . 339
2.13.4 Assumptions . . . 339
2.13.5 R Implementation . . . 339
2.13.6 Output & Results . . . 339
2.13.7 Interpretation . . . 340
2.13.8 Practical Tips . . . 340
2.13.9 For Reviewers . . . 340
2.13.10 See also — labs in this chapter . . . 340
2.14 Measures of Dispersion . . . 340
2.14.1 Introduction . . . 340
2.14.2 Prerequisites . . . 340
2.14.3 Theory . . . 341
2.14.4 Assumptions . . . 341
2.14.5 R Implementation . . . 341
2.14.6 Output & Results . . . 342
2.14.7 Interpretation . . . 342
2.14.8 Practical Tips . . . 343
2.14.9 For Reviewers . . . 343
2.14.10 See also — labs in this chapter . . . 343
2.15 Measures of Central Tendency . . . 343
2.15.1 Introduction . . . 343
2.15.2 Prerequisites . . . 343
2.15.3 Theory . . . 344
2.15.4 Assumptions . . . 344
2.15.5 R Implementation . . . 344
2.15.6 Output & Results . . . 345
2.15.7 Interpretation . . . 345
2.15.8 Practical Tips . . . 345
2.15.9 For Reviewers . . . 345
2.15.10 See also — labs in this chapter . . . 346
2.16 Measures of Shape . . . 346

2.16.1 Introduction . . . 346
2.16.2 Prerequisites . . . 346
2.16.3 Theory . . . 346
2.16.4 Assumptions . . . 347
2.16.5 R Implementation . . . 347
2.16.6 Output & Results . . . 347
2.16.7 Interpretation . . . 348
2.16.8 Practical Tips . . . 348
2.16.9 For Reviewers . . . 348
2.16.10 See also — labs in this chapter . . . 348
2.17 Outlier Detection Rules . . . 348
2.17.1 Introduction . . . 348
2.17.2 Prerequisites . . . 349
2.17.3 Theory . . . 349
2.17.4 Assumptions . . . 349
2.17.5 R Implementation . . . 349
2.17.6 Output & Results . . . 350
2.17.7 Interpretation . . . 350
2.17.8 Practical Tips . . . 350
2.17.9 For Reviewers . . . 350
2.17.10 See also — labs in this chapter . . . 350
2.18 Percentile Ranks . . . 351
2.18.1 Introduction . . . 351
2.18.2 Prerequisites . . . 351
2.18.3 Theory . . . 351
2.18.4 Assumptions . . . 351
2.18.5 R Implementation . . . 351
2.18.6 Output & Results . . . 352
2.18.7 Interpretation . . . 352
2.18.8 Practical Tips . . . 352
2.18.9 For Reviewers . . . 352
2.18.10 See also — labs in this chapter . . . 353
2.19 Quantiles and Percentiles . . . 353
2.19.1 Introduction . . . 353
2.19.2 Prerequisites . . . 353
2.19.3 Theory . . . 353
2.19.4 Assumptions . . . 353
2.19.5 R Implementation . . . 353
2.19.6 Output & Results . . . 354
2.19.7 Interpretation . . . 354
2.19.8 Practical Tips . . . 354
2.19.9 For Reviewers . . . 355
2.19.10 See also — labs in this chapter . . . 355
2.20 Robust Statistics: An Overview . . . 355
2.20.1 Introduction . . . 355
2.20.2 Prerequisites . . . 355

2.20.3 Theory . . . 355
2.20.4 Assumptions . . . 356
2.20.5 R Implementation . . . 356
2.20.6 Output & Results . . . 357
2.20.7 Interpretation . . . 357
2.20.8 Practical Tips . . . 357
2.20.9 For Reviewers . . . 357
2.20.10 See also — labs in this chapter . . . 358
2.21 Skewness . . . 358
2.21.1 Introduction . . . 358
2.21.2 Prerequisites . . . 358
2.21.3 Theory . . . 358
2.21.4 Assumptions . . . 358
2.21.5 R Implementation . . . 358
2.21.6 Output & Results . . . 359
2.21.7 Interpretation . . . 359
2.21.8 Practical Tips . . . 359
2.21.9 For Reviewers . . . 359
2.21.10 See also — labs in this chapter . . . 359
2.22 Standardisation and z-Scores . . . 360
2.22.1 Introduction . . . 360
2.22.2 Prerequisites . . . 360
2.22.3 Theory . . . 360
2.22.4 Assumptions . . . 360
2.22.5 R Implementation . . . 360
2.22.6 Output & Results . . . 361
2.22.7 Interpretation . . . 361
2.22.8 Practical Tips . . . 361
2.22.9 For Reviewers . . . 361
2.22.10 See also — labs in this chapter . . . 362
2.23 Stem-and-Leaf Plots . . . 362
2.23.1 Introduction . . . 362
2.23.2 Prerequisites . . . 362
2.23.3 Theory . . . 362
2.23.4 Assumptions . . . 362
2.23.5 R Implementation . . . 362
2.23.6 Output & Results . . . 363
2.23.7 Interpretation . . . 363
2.23.8 Practical Tips . . . 363
2.23.9 For Reviewers . . . 363
2.23.10 See also — labs in this chapter . . . 364
2.24 Summary by Group . . . 364
2.24.1 Introduction . . . 364
2.24.2 Prerequisites . . . 364
2.24.3 Theory . . . 364
2.24.4 Assumptions . . . 364

2.24.5 R Implementation . . . 364
2.24.6 Output & Results . . . 365
2.24.7 Interpretation . . . 366
2.24.8 Practical Tips . . . 366
2.24.9 For Reviewers . . . 366
2.24.10 See also — labs in this chapter . . . 366
2.25 Trimmed and Winsorised Means . . . 366
2.25.1 Introduction . . . 366
2.25.2 Prerequisites . . . 367
2.25.3 Theory . . . 367
2.25.4 Assumptions . . . 367
2.25.5 R Implementation . . . 367
2.25.6 Output & Results . . . 367
2.25.7 Interpretation . . . 368
2.25.8 Practical Tips . . . 368
2.25.9 For Reviewers . . . 368
2.25.10 See also — labs in this chapter . . . 368
2.26 Variance and Standard Deviation . . . 368
2.26.1 Introduction . . . 368
2.26.2 Prerequisites . . . 369
2.26.3 Theory . . . 369
2.26.4 Assumptions . . . 369
2.26.5 R Implementation . . . 369
2.26.6 Output & Results . . . 370
2.26.7 Interpretation . . . 370
2.26.8 Practical Tips . . . 370
2.26.9 For Reviewers . . . 370
2.26.10 See also — labs in this chapter . . . 370
2.27 The Weighted Mean . . . 370
2.27.1 Introduction . . . 370
2.27.2 Prerequisites . . . 371
2.27.3 Theory . . . 371
2.27.4 Assumptions . . . 371
2.27.5 R Implementation . . . 371
2.27.6 Output & Results . . . 372
2.27.7 Interpretation . . . 372
2.27.8 Practical Tips . . . 372
2.27.9 For Reviewers . . . 372
2.27.10 See also — labs in this chapter . . . 373
2.28 Week 2, Session 1 — Descriptive statistics and Table 1 . . . 373
2.28.1 Learning objectives . . . 373
2.28.2 Prerequisites . . . 373
2.28.3 Background . . . 373
2.28.4 Setup . . . 374
2.28.5 1. Goal . . . 374
2.28.6 2. Approach . . . 374

2.28.7 3. Execution 374
2.28.8 4. Check 375
2.28.9 5. Report 375
2.28.10 Common pitfalls 376
2.28.11 Further reading 376
2.28.12 Session info 376
2.28.13 See also — chapter index 376

3 Probability Theory **377**
3.1 Method pages 377
3.2 Labs 378
3.3 Bayes' Theorem 378
3.3.1 Introduction 378
3.3.2 Prerequisites 379
3.3.3 Theory 379
3.3.4 Assumptions 379
3.3.5 R Implementation 379
3.3.6 Output & Results 380
3.3.7 Interpretation 380
3.3.8 Practical Tips 380
3.3.9 For Reviewers 381
3.3.10 See also — labs in this chapter 381
3.4 The Bernoulli Distribution 381
3.4.1 Introduction 381
3.4.2 Prerequisites 381
3.4.3 Theory 381
3.4.4 Assumptions 382
3.4.5 R Implementation 382
3.4.6 Output & Results 382
3.4.7 Interpretation 383
3.4.8 Practical Tips 383
3.4.9 For Reviewers 383
3.4.10 See also — labs in this chapter 383
3.5 The Beta Distribution 383
3.5.1 Introduction 383
3.5.2 Prerequisites 384
3.5.3 Theory 384
3.5.4 Assumptions 384
3.5.5 R Implementation 384
3.5.6 Output & Results 385
3.5.7 Interpretation 385
3.5.8 Practical Tips 385
3.5.9 For Reviewers 386
3.5.10 See also — labs in this chapter 386
3.6 The Binomial Distribution 386
3.6.1 Introduction 386

3.6.2 Prerequisites . . . 386
3.6.3 Theory . . . 386
3.6.4 Assumptions . . . 387
3.6.5 R Implementation . . . 387
3.6.6 Output & Results . . . 387
3.6.7 Interpretation . . . 388
3.6.8 Practical Tips . . . 388
3.6.9 For Reviewers . . . 388
3.6.10 See also — labs in this chapter . . . 388
3.7 The Chi-Squared Distribution . . . 389
3.7.1 Introduction . . . 389
3.7.2 Prerequisites . . . 389
3.7.3 Theory . . . 389
3.7.4 Assumptions . . . 389
3.7.5 R Implementation . . . 390
3.7.6 Output & Results . . . 390
3.7.7 Interpretation . . . 390
3.7.8 Practical Tips . . . 391
3.7.9 For Reviewers . . . 391
3.7.10 See also — labs in this chapter . . . 391
3.8 Conditional Distributions . . . 391
3.8.1 Introduction . . . 391
3.8.2 Prerequisites . . . 391
3.8.3 Theory . . . 392
3.8.4 Assumptions . . . 392
3.8.5 R Implementation . . . 392
3.8.6 Output & Results . . . 393
3.8.7 Interpretation . . . 393
3.8.8 Practical Tips . . . 393
3.8.9 For Reviewers . . . 394
3.8.10 See also — labs in this chapter . . . 394
3.9 Conditional Probability . . . 394
3.9.1 Introduction . . . 394
3.9.2 Prerequisites . . . 394
3.9.3 Theory . . . 394
3.9.4 Assumptions . . . 395
3.9.5 R Implementation . . . 395
3.9.6 Output & Results . . . 396
3.9.7 Interpretation . . . 396
3.9.8 Practical Tips . . . 396
3.9.9 For Reviewers . . . 397
3.9.10 See also — labs in this chapter . . . 397
3.10 Convolutions of Distributions . . . 397
3.10.1 Introduction . . . 397
3.10.2 Prerequisites . . . 397
3.10.3 Theory . . . 397

3.10.4 Assumptions . . . 398
3.10.5 R Implementation . . . 398
3.10.6 Output & Results . . . 399
3.10.7 Interpretation . . . 399
3.10.8 Practical Tips . . . 399
3.10.9 For Reviewers . . . 399
3.10.10 See also — labs in this chapter . . . 400
3.11 Copulas: An Introduction . . . 400
3.11.1 Introduction . . . 400
3.11.2 Prerequisites . . . 400
3.11.3 Theory . . . 400
3.11.4 Assumptions . . . 401
3.11.5 R Implementation . . . 401
3.11.6 Output & Results . . . 401
3.11.7 Interpretation . . . 402
3.11.8 Practical Tips . . . 402
3.11.9 For Reviewers . . . 402
3.11.10 See also — labs in this chapter . . . 402
3.12 The Correlation Coefficient . . . 402
3.12.1 Introduction . . . 402
3.12.2 Prerequisites . . . 403
3.12.3 Theory . . . 403
3.12.4 Assumptions . . . 403
3.12.5 R Implementation . . . 403
3.12.6 Output & Results . . . 404
3.12.7 Interpretation . . . 404
3.12.8 Practical Tips . . . 404
3.12.9 For Reviewers . . . 405
3.12.10 See also — labs in this chapter . . . 405
3.13 The Cumulative Distribution Function . . . 405
3.13.1 Introduction . . . 405
3.13.2 Prerequisites . . . 405
3.13.3 Theory . . . 405
3.13.4 Assumptions . . . 406
3.13.5 R Implementation . . . 406
3.13.6 Output & Results . . . 406
3.13.7 Interpretation . . . 407
3.13.8 Practical Tips . . . 407
3.13.9 For Reviewers . . . 407
3.13.10 See also — labs in this chapter . . . 407
3.14 Discrete vs Continuous Variables . . . 407
3.14.1 Introduction . . . 407
3.14.2 Prerequisites . . . 408
3.14.3 Theory . . . 408
3.14.4 Assumptions . . . 408
3.14.5 R Implementation . . . 408

3.14.6 Output & Results . . . 409
3.14.7 Interpretation . . . 409
3.14.8 Practical Tips . . . 409
3.14.9 For Reviewers . . . 410
3.14.10 See also — labs in this chapter . . . 410
3.15 Expectation of a Random Variable . . . 410
3.15.1 Introduction . . . 410
3.15.2 Prerequisites . . . 410
3.15.3 Theory . . . 410
3.15.4 Assumptions . . . 411
3.15.5 R Implementation . . . 411
3.15.6 Output & Results . . . 412
3.15.7 Interpretation . . . 412
3.15.8 Practical Tips . . . 412
3.15.9 For Reviewers . . . 413
3.15.10 See also — labs in this chapter . . . 413
3.16 The Exponential Distribution . . . 413
3.16.1 Introduction . . . 413
3.16.2 Prerequisites . . . 413
3.16.3 Theory . . . 413
3.16.4 Assumptions . . . 414
3.16.5 R Implementation . . . 414
3.16.6 Output & Results . . . 414
3.16.7 Interpretation . . . 415
3.16.8 Practical Tips . . . 415
3.16.9 For Reviewers . . . 415
3.16.10 See also — labs in this chapter . . . 415
3.17 The F Distribution . . . 415
3.17.1 Introduction . . . 415
3.17.2 Prerequisites . . . 416
3.17.3 Theory . . . 416
3.17.4 Assumptions . . . 416
3.17.5 R Implementation . . . 416
3.17.6 Output & Results . . . 417
3.17.7 Interpretation . . . 417
3.17.8 Practical Tips . . . 417
3.17.9 For Reviewers . . . 418
3.17.10 See also — labs in this chapter . . . 418
3.18 The Gamma Distribution . . . 418
3.18.1 Introduction . . . 418
3.18.2 Prerequisites . . . 418
3.18.3 Theory . . . 418
3.18.4 Assumptions . . . 419
3.18.5 R Implementation . . . 419
3.18.6 Output & Results . . . 420
3.18.7 Interpretation . . . 420

3.18.8 Practical Tips . . . 420
3.18.9 For Reviewers . . . 420
3.18.10 See also — labs in this chapter . . . 420
3.19 The Geometric Distribution . . . 421
3.19.1 Introduction . . . 421
3.19.2 Prerequisites . . . 421
3.19.3 Theory . . . 421
3.19.4 Assumptions . . . 421
3.19.5 R Implementation . . . 421
3.19.6 Output & Results . . . 422
3.19.7 Interpretation . . . 422
3.19.8 Practical Tips . . . 423
3.19.9 For Reviewers . . . 423
3.19.10 See also — labs in this chapter . . . 423
3.20 The Hazard Function . . . 423
3.20.1 Introduction . . . 423
3.20.2 Prerequisites . . . 423
3.20.3 Theory . . . 423
3.20.4 Assumptions . . . 424
3.20.5 R Implementation . . . 424
3.20.6 Output & Results . . . 425
3.20.7 Interpretation . . . 425
3.20.8 Practical Tips . . . 425
3.20.9 For Reviewers . . . 425
3.20.10 See also — labs in this chapter . . . 426
3.21 The Hypergeometric Distribution . . . 426
3.21.1 Introduction . . . 426
3.21.2 Prerequisites . . . 426
3.21.3 Theory . . . 426
3.21.4 Assumptions . . . 427
3.21.5 R Implementation . . . 427
3.21.6 Output & Results . . . 427
3.21.7 Interpretation . . . 428
3.21.8 Practical Tips . . . 428
3.21.9 For Reviewers . . . 428
3.21.10 See also — labs in this chapter . . . 428
3.22 Independence . . . 428
3.22.1 Introduction . . . 428
3.22.2 Prerequisites . . . 429
3.22.3 Theory . . . 429
3.22.4 Assumptions . . . 429
3.22.5 R Implementation . . . 429
3.22.6 Output & Results . . . 430
3.22.7 Interpretation . . . 430
3.22.8 Practical Tips . . . 430
3.22.9 For Reviewers . . . 431

3.22.10 See also — labs in this chapter 431
3.23 Joint Distributions 431
3.23.1 Introduction 431
3.23.2 Prerequisites 431
3.23.3 Theory 431
3.23.4 Assumptions 432
3.23.5 R Implementation 432
3.23.6 Output & Results 432
3.23.7 Interpretation 433
3.23.8 Practical Tips 433
3.23.9 For Reviewers 433
3.23.10 See also — labs in this chapter 433
3.24 Kolmogorov's Axioms of Probability 434
3.24.1 Introduction 434
3.24.2 Prerequisites 434
3.24.3 Theory 434
3.24.4 Assumptions 434
3.24.5 R Implementation 435
3.24.6 Output & Results 435
3.24.7 Interpretation 436
3.24.8 Practical Tips 436
3.24.9 For Reviewers 436
3.24.10 See also — labs in this chapter 436
3.25 Law of Total Probability 436
3.25.1 Introduction 436
3.25.2 Prerequisites 437
3.25.3 Theory 437
3.25.4 Assumptions 437
3.25.5 R Implementation 437
3.25.6 Output & Results 438
3.25.7 Interpretation 438
3.25.8 Practical Tips 438
3.25.9 For Reviewers 439
3.25.10 See also — labs in this chapter 439
3.26 The Log-Normal Distribution 439
3.26.1 Introduction 439
3.26.2 Prerequisites 439
3.26.3 Theory 439
3.26.4 Assumptions 440
3.26.5 R Implementation 440
3.26.6 Output & Results 440
3.26.7 Interpretation 440
3.26.8 Practical Tips 441
3.26.9 For Reviewers 441
3.26.10 See also — labs in this chapter 441
3.27 Marginal Distributions 441

3.27.1 Introduction . . . 441
3.27.2 Prerequisites . . . 441
3.27.3 Theory . . . 441
3.27.4 Assumptions . . . 442
3.27.5 R Implementation . . . 442
3.27.6 Output & Results . . . 443
3.27.7 Interpretation . . . 443
3.27.8 Practical Tips . . . 443
3.27.9 For Reviewers . . . 443
3.27.10 See also — labs in this chapter . . . 444
3.28 The Multinomial Distribution . . . 444
3.28.1 Introduction . . . 444
3.28.2 Prerequisites . . . 444
3.28.3 Theory . . . 444
3.28.4 Assumptions . . . 445
3.28.5 R Implementation . . . 445
3.28.6 Output & Results . . . 445
3.28.7 Interpretation . . . 446
3.28.8 Practical Tips . . . 446
3.28.9 For Reviewers . . . 446
3.28.10 See also — labs in this chapter . . . 446
3.29 The Multivariate Normal Distribution . . . 447
3.29.1 Introduction . . . 447
3.29.2 Prerequisites . . . 447
3.29.3 Theory . . . 447
3.29.4 Assumptions . . . 447
3.29.5 R Implementation . . . 447
3.29.6 Output & Results . . . 448
3.29.7 Interpretation . . . 448
3.29.8 Practical Tips . . . 449
3.29.9 For Reviewers . . . 449
3.29.10 See also — labs in this chapter . . . 449
3.30 The Negative Binomial Distribution . . . 449
3.30.1 Introduction . . . 449
3.30.2 Prerequisites . . . 449
3.30.3 Theory . . . 450
3.30.4 Assumptions . . . 450
3.30.5 R Implementation . . . 450
3.30.6 Output & Results . . . 451
3.30.7 Interpretation . . . 451
3.30.8 Practical Tips . . . 451
3.30.9 For Reviewers . . . 452
3.30.10 See also — labs in this chapter . . . 452
3.31 The Normal Distribution . . . 452
3.31.1 Introduction . . . 452
3.31.2 Prerequisites . . . 452

3.31.3 Theory . . . 452
3.31.4 Assumptions . . . 453
3.31.5 R Implementation . . . 453
3.31.6 Output & Results . . . 454
3.31.7 Interpretation . . . 454
3.31.8 Practical Tips . . . 454
3.31.9 For Reviewers . . . 454
3.31.10 See also — labs in this chapter . . . 454
3.32 The Poisson Distribution . . . 455
3.32.1 Introduction . . . 455
3.32.2 Prerequisites . . . 455
3.32.3 Theory . . . 455
3.32.4 Assumptions . . . 455
3.32.5 R Implementation . . . 455
3.32.6 Output & Results . . . 456
3.32.7 Interpretation . . . 457
3.32.8 Practical Tips . . . 457
3.32.9 For Reviewers . . . 457
3.32.10 See also — labs in this chapter . . . 457
3.33 The Probability Density Function . . . 457
3.33.1 Introduction . . . 457
3.33.2 Prerequisites . . . 458
3.33.3 Theory . . . 458
3.33.4 Assumptions . . . 458
3.33.5 R Implementation . . . 458
3.33.6 Output & Results . . . 459
3.33.7 Interpretation . . . 459
3.33.8 Practical Tips . . . 459
3.33.9 For Reviewers . . . 460
3.33.10 See also — labs in this chapter . . . 460
3.34 The Probability Mass Function . . . 460
3.34.1 Introduction . . . 460
3.34.2 Prerequisites . . . 460
3.34.3 Theory . . . 460
3.34.4 Assumptions . . . 461
3.34.5 R Implementation . . . 461
3.34.6 Output & Results . . . 462
3.34.7 Interpretation . . . 462
3.34.8 Practical Tips . . . 462
3.34.9 For Reviewers . . . 463
3.34.10 See also — labs in this chapter . . . 463
3.35 Random Variables . . . 463
3.35.1 Introduction . . . 463
3.35.2 Prerequisites . . . 463
3.35.3 Theory . . . 463
3.35.4 Assumptions . . . 464

3.35.5 R Implementation . . . 464
3.35.6 Output & Results . . . 465
3.35.7 Interpretation . . . 465
3.35.8 Practical Tips . . . 465
3.35.9 For Reviewers . . . 465
3.35.10 See also — labs in this chapter . . . 466
3.36 Sample Space and Events . . . 466
3.36.1 Introduction . . . 466
3.36.2 Prerequisites . . . 466
3.36.3 Theory . . . 466
3.36.4 Assumptions . . . 467
3.36.5 R Implementation . . . 467
3.36.6 Output & Results . . . 467
3.36.7 Interpretation . . . 468
3.36.8 Practical Tips . . . 468
3.36.9 For Reviewers . . . 468
3.36.10 See also — labs in this chapter . . . 468
3.37 The Survival Function . . . 469
3.37.1 Introduction . . . 469
3.37.2 Prerequisites . . . 469
3.37.3 Theory . . . 469
3.37.4 Assumptions . . . 469
3.37.5 R Implementation . . . 470
3.37.6 Output & Results . . . 470
3.37.7 Interpretation . . . 471
3.37.8 Practical Tips . . . 471
3.37.9 For Reviewers . . . 471
3.37.10 See also — labs in this chapter . . . 471
3.38 Student's t-Distribution . . . 471
3.38.1 Introduction . . . 471
3.38.2 Prerequisites . . . 472
3.38.3 Theory . . . 472
3.38.4 Assumptions . . . 472
3.38.5 R Implementation . . . 472
3.38.6 Output & Results . . . 473
3.38.7 Interpretation . . . 473
3.38.8 Practical Tips . . . 473
3.38.9 For Reviewers . . . 474
3.38.10 See also — labs in this chapter . . . 474
3.39 Transformations of Random Variables . . . 474
3.39.1 Introduction . . . 474
3.39.2 Prerequisites . . . 474
3.39.3 Theory . . . 474
3.39.4 Assumptions . . . 475
3.39.5 R Implementation . . . 475
3.39.6 Output & Results . . . 476

3.39.7 Interpretation . . . 476
3.39.8 Practical Tips . . . 476
3.39.9 For Reviewers . . . 476
3.39.10 See also — labs in this chapter . . . 476
3.40 The Uniform Distribution . . . 477
3.40.1 Introduction . . . 477
3.40.2 Prerequisites . . . 477
3.40.3 Theory . . . 477
3.40.4 Assumptions . . . 477
3.40.5 R Implementation . . . 477
3.40.6 Output & Results . . . 478
3.40.7 Interpretation . . . 478
3.40.8 Practical Tips . . . 478
3.40.9 For Reviewers . . . 478
3.40.10 See also — labs in this chapter . . . 479
3.41 Variance and Covariance . . . 479
3.41.1 Introduction . . . 479
3.41.2 Prerequisites . . . 479
3.41.3 Theory . . . 479
3.41.4 Assumptions . . . 479
3.41.5 R Implementation . . . 480
3.41.6 Output & Results . . . 480
3.41.7 Interpretation . . . 481
3.41.8 Practical Tips . . . 481
3.41.9 For Reviewers . . . 481
3.41.10 See also — labs in this chapter . . . 481
3.42 The Weibull Distribution . . . 481
3.42.1 Introduction . . . 481
3.42.2 Prerequisites . . . 482
3.42.3 Theory . . . 482
3.42.4 Assumptions . . . 482
3.42.5 R Implementation . . . 482
3.42.6 Output & Results . . . 483
3.42.7 Interpretation . . . 483
3.42.8 Practical Tips . . . 483
3.42.9 For Reviewers . . . 484
3.42.10 See also — labs in this chapter . . . 484
3.43 Week 2, Session 5 — Continuous distributions and Q-Q plots 484
3.43.1 Learning objectives . . . 484
3.43.2 Prerequisites . . . 484
3.43.3 Background . . . 484
3.43.4 Setup . . . 485
3.43.5 1. Hypothesis . . . 485
3.43.6 2. Visualise . . . 485
3.43.7 3. Assumptions . . . 486
3.43.8 4. Conduct . . . 486

3.43.9 5. Concluding statement . . . 487
3.43.10 Common pitfalls . . . 487
3.43.11 Further reading . . . 487
3.43.12 Session info . . . 487
3.43.13 See also — chapter index . . . 487
3.44 Week 2, Session 4 — Discrete distributions: Bernoulli, binomial, Poisson . . . 488
3.44.1 Learning objectives . . . 488
3.44.2 Prerequisites . . . 488
3.44.3 Background . . . 488
3.44.4 Setup . . . 488
3.44.5 1. Hypothesis . . . 489
3.44.6 2. Visualise . . . 489
3.44.7 3. Assumptions . . . 489
3.44.8 4. Conduct . . . 489
3.44.9 5. Concluding statement . . . 490
3.44.10 Common pitfalls . . . 490
3.44.11 Further reading . . . 491
3.44.12 Session info . . . 491
3.44.13 See also — chapter index . . . 491
3.45 Week 2, Session 2 — Probability, conditional probability, Bayes' theorem . . . 491
3.45.1 Learning objectives . . . 491
3.45.2 Prerequisites . . . 491
3.45.3 Background . . . 491
3.45.4 Setup . . . 492
3.45.5 1. Hypothesis . . . 492
3.45.6 2. Visualise . . . 492
3.45.7 3. Assumptions . . . 493
3.45.8 4. Conduct . . . 493
3.45.9 5. Concluding statement . . . 494
3.45.10 Common pitfalls . . . 494
3.45.11 Further reading . . . 494
3.45.12 Session info . . . 494
3.45.13 See also — chapter index . . . 494

4 Inferential Statistics 495
4.1 Method pages . . . 495
4.2 Labs . . . 496
4.3 Anderson-Darling Test . . . 497
4.3.1 Introduction . . . 497
4.3.2 Prerequisites . . . 497
4.3.3 Theory . . . 497
4.3.4 Assumptions . . . 497
4.3.5 R Implementation . . . 497
4.3.6 Output & Results . . . 498

4.3.7 Interpretation . . . 498
4.3.8 Practical Tips . . . 498
4.3.9 For Reviewers . . . 499
4.3.10 See also — labs in this chapter . . . 499
4.4 Bartlett's Test . . . 499
4.4.1 Introduction . . . 499
4.4.2 Prerequisites . . . 499
4.4.3 Theory . . . 499
4.4.4 Assumptions . . . 500
4.4.5 R Implementation . . . 500
4.4.6 Output & Results . . . 500
4.4.7 Interpretation . . . 501
4.4.8 Practical Tips . . . 501
4.4.9 For Reviewers . . . 501
4.4.10 See also — labs in this chapter . . . 501
4.5 Benjamini-Hochberg FDR . . . 502
4.5.1 Introduction . . . 502
4.5.2 Prerequisites . . . 502
4.5.3 Theory . . . 502
4.5.4 Assumptions . . . 502
4.5.5 R Implementation . . . 502
4.5.6 Output & Results . . . 503
4.5.7 Interpretation . . . 503
4.5.8 Practical Tips . . . 503
4.5.9 For Reviewers . . . 504
4.5.10 See also — labs in this chapter . . . 504
4.6 Benjamini-Yekutieli FDR . . . 504
4.6.1 Introduction . . . 504
4.6.2 Prerequisites . . . 504
4.6.3 Theory . . . 504
4.6.4 Assumptions . . . 505
4.6.5 R Implementation . . . 505
4.6.6 Output & Results . . . 505
4.6.7 Interpretation . . . 506
4.6.8 Practical Tips . . . 506
4.6.9 For Reviewers . . . 506
4.6.10 See also — labs in this chapter . . . 506
4.7 Bonferroni Correction . . . 507
4.7.1 Introduction . . . 507
4.7.2 Prerequisites . . . 507
4.7.3 Theory . . . 507
4.7.4 Assumptions . . . 507
4.7.5 R Implementation . . . 507
4.7.6 Output & Results . . . 508
4.7.7 Interpretation . . . 509
4.7.8 Practical Tips . . . 509

4.7.9 For Reviewers . . . 509
4.7.10 See also — labs in this chapter . . . 509
4.8 Bootstrap Confidence Intervals . . . 509
4.8.1 Introduction . . . 509
4.8.2 Prerequisites . . . 510
4.8.3 Theory . . . 510
4.8.4 Assumptions . . . 510
4.8.5 R Implementation . . . 510
4.8.6 Output & Results . . . 511
4.8.7 Interpretation . . . 511
4.8.8 Practical Tips . . . 511
4.8.9 For Reviewers . . . 511
4.8.10 See also — labs in this chapter . . . 511
4.9 The Bootstrap: Introduction . . . 512
4.9.1 Introduction . . . 512
4.9.2 Prerequisites . . . 512
4.9.3 Theory . . . 512
4.9.4 Assumptions . . . 512
4.9.5 R Implementation . . . 513
4.9.6 Output & Results . . . 513
4.9.7 Interpretation . . . 513
4.9.8 Practical Tips . . . 514
4.9.9 For Reviewers . . . 514
4.9.10 See also — labs in this chapter . . . 514
4.10 Chi-Squared Goodness-of-Fit Test . . . 514
4.10.1 Introduction . . . 514
4.10.2 Prerequisites . . . 514
4.10.3 Theory . . . 515
4.10.4 Assumptions . . . 515
4.10.5 R Implementation . . . 515
4.10.6 Output & Results . . . 515
4.10.7 Interpretation . . . 516
4.10.8 Practical Tips . . . 516
4.10.9 For Reviewers . . . 516
4.10.10 See also — labs in this chapter . . . 516
4.11 Chi-Squared Test of Independence . . . 517
4.11.1 Introduction . . . 517
4.11.2 Prerequisites . . . 517
4.11.3 Theory . . . 517
4.11.4 Assumptions . . . 517
4.11.5 R Implementation . . . 517
4.11.6 Output & Results . . . 518
4.11.7 Interpretation . . . 518
4.11.8 Practical Tips . . . 518
4.11.9 For Reviewers . . . 519
4.11.10 See also — labs in this chapter . . . 519

4.12 Cochran-Mantel-Haenszel Test . . . 519
4.12.1 Introduction . . . 519
4.12.2 Prerequisites . . . 519
4.12.3 Theory . . . 519
4.12.4 Assumptions . . . 520
4.12.5 R Implementation . . . 520
4.12.6 Output & Results . . . 520
4.12.7 Interpretation . . . 521
4.12.8 Practical Tips . . . 521
4.12.9 For Reviewers . . . 521
4.12.10 See also — labs in this chapter . . . 521
4.13 Effect Sizes: Overview . . . 522
4.13.1 Introduction . . . 522
4.13.2 Prerequisites . . . 522
4.13.3 Theory . . . 522
4.13.4 Assumptions . . . 523
4.13.5 R Implementation . . . 523
4.13.6 Output & Results . . . 523
4.13.7 Interpretation . . . 524
4.13.8 Practical Tips . . . 524
4.13.9 For Reviewers . . . 524
4.13.10 See also — labs in this chapter . . . 524
4.14 Equivalence Testing with TOST . . . 525
4.14.1 Introduction . . . 525
4.14.2 Prerequisites . . . 525
4.14.3 Theory . . . 525
4.14.4 Assumptions . . . 525
4.14.5 R Implementation . . . 525
4.14.6 Output & Results . . . 526
4.14.7 Interpretation . . . 526
4.14.8 Practical Tips . . . 526
4.14.9 For Reviewers . . . 526
4.14.10 See also — labs in this chapter . . . 527
4.15 Fisher's Exact Test . . . 527
4.15.1 Introduction . . . 527
4.15.2 Prerequisites . . . 527
4.15.3 Theory . . . 527
4.15.4 Assumptions . . . 528
4.15.5 R Implementation . . . 528
4.15.6 Output & Results . . . 528
4.15.7 Interpretation . . . 528
4.15.8 Practical Tips . . . 529
4.15.9 For Reviewers . . . 529
4.15.10 See also — labs in this chapter . . . 529
4.16 Friedman Test . . . 529
4.16.1 Introduction . . . 529

4.16.2 Prerequisites 530
4.16.3 Theory 530
4.16.4 Assumptions 530
4.16.5 R Implementation 530
4.16.6 Output & Results 531
4.16.7 Interpretation 531
4.16.8 Practical Tips 531
4.16.9 R Packages Used 532
4.16.10 For Reviewers 532
4.16.11 See also — labs in this chapter 532
4.17 Holm's Step-Down Correction 532
4.17.1 Introduction 532
4.17.2 Prerequisites 533
4.17.3 Theory 533
4.17.4 Assumptions 533
4.17.5 R Implementation 533
4.17.6 Output & Results 534
4.17.7 Interpretation 534
4.17.8 Practical Tips 534
4.17.9 For Reviewers 534
4.17.10 See also — labs in this chapter 535
4.18 The Jackknife 535
4.18.1 Introduction 535
4.18.2 Prerequisites 535
4.18.3 Theory 535
4.18.4 Assumptions 536
4.18.5 R Implementation 536
4.18.6 Output & Results 536
4.18.7 Interpretation 536
4.18.8 Practical Tips 536
4.18.9 For Reviewers 537
4.18.10 See also — labs in this chapter 537
4.19 Kendall's Tau 537
4.19.1 Introduction 537
4.19.2 Prerequisites 537
4.19.3 Theory 537
4.19.4 Assumptions 538
4.19.5 R Implementation 538
4.19.6 Output & Results 538
4.19.7 Interpretation 539
4.19.8 Practical Tips 539
4.19.9 For Reviewers 539
4.19.10 See also — labs in this chapter 539
4.20 Kolmogorov-Smirnov Test 540
4.20.1 Introduction 540
4.20.2 Prerequisites 540

4.20.3 Theory . . . 540
4.20.4 Assumptions . . . 540
4.20.5 R Implementation . . . 541
4.20.6 Output & Results . . . 541
4.20.7 Interpretation . . . 541
4.20.8 Practical Tips . . . 541
4.20.9 R Packages Used . . . 542
4.20.10 For Reviewers . . . 542
4.20.11 See also — labs in this chapter . . . 542
4.21 Kruskal-Wallis Test . . . 543
4.21.1 Introduction . . . 543
4.21.2 Prerequisites . . . 543
4.21.3 Theory . . . 543
4.21.4 Assumptions . . . 543
4.21.5 R Implementation . . . 543
4.21.6 Output & Results . . . 544
4.21.7 Interpretation . . . 544
4.21.8 Practical Tips . . . 544
4.21.9 For Reviewers . . . 545
4.21.10 See also — labs in this chapter . . . 545
4.22 Levene's Test of Variances . . . 545
4.22.1 Introduction . . . 545
4.22.2 Prerequisites . . . 545
4.22.3 Theory . . . 545
4.22.4 Assumptions . . . 546
4.22.5 R Implementation . . . 546
4.22.6 Output & Results . . . 546
4.22.7 Interpretation . . . 547
4.22.8 Practical Tips . . . 547
4.22.9 R Packages Used . . . 547
4.22.10 For Reviewers . . . 548
4.22.11 See also — labs in this chapter . . . 548
4.23 Mann-Whitney U Test . . . 548
4.23.1 Introduction . . . 548
4.23.2 Prerequisites . . . 548
4.23.3 Theory . . . 548
4.23.4 Assumptions . . . 549
4.23.5 R Implementation . . . 549
4.23.6 Output & Results . . . 549
4.23.7 Interpretation . . . 549
4.23.8 Practical Tips . . . 550
4.23.9 For Reviewers . . . 550
4.23.10 See also — labs in this chapter . . . 550
4.24 McNemar's Test . . . 550
4.24.1 Introduction . . . 550
4.24.2 Prerequisites . . . 550

4.24.3 Theory . . . 551
4.24.4 Assumptions . . . 551
4.24.5 R Implementation . . . 551
4.24.6 Output & Results . . . 552
4.24.7 Interpretation . . . 552
4.24.8 Practical Tips . . . 552
4.24.9 For Reviewers . . . 552
4.24.10 See also — labs in this chapter . . . 553
4.25 Mixed ANOVA . . . 553
4.25.1 Introduction . . . 553
4.25.2 Prerequisites . . . 553
4.25.3 Theory . . . 553
4.25.4 Assumptions . . . 553
4.25.5 R Implementation . . . 554
4.25.6 Output & Results . . . 554
4.25.7 Interpretation . . . 555
4.25.8 Practical Tips . . . 555
4.25.9 For Reviewers . . . 555
4.25.10 See also — labs in this chapter . . . 555
4.26 Multiple Comparisons: Overview . . . 555
4.26.1 Introduction . . . 555
4.26.2 Prerequisites . . . 556
4.26.3 Theory . . . 556
4.26.4 Assumptions . . . 556
4.26.5 R Implementation . . . 556
4.26.6 Output & Results . . . 556
4.26.7 Interpretation . . . 557
4.26.8 Practical Tips . . . 557
4.26.9 For Reviewers . . . 557
4.26.10 See also — labs in this chapter . . . 557
4.27 Null and Alternative Hypotheses . . . 558
4.27.1 Introduction . . . 558
4.27.2 Prerequisites . . . 558
4.27.3 Theory . . . 558
4.27.4 Assumptions . . . 558
4.27.5 R Implementation . . . 559
4.27.6 Output & Results . . . 559
4.27.7 Interpretation . . . 559
4.27.8 Practical Tips . . . 559
4.27.9 For Reviewers . . . 560
4.27.10 See also — labs in this chapter . . . 560
4.28 One-Proportion Test . . . 560
4.28.1 Introduction . . . 560
4.28.2 Prerequisites . . . 560
4.28.3 Theory . . . 560
4.28.4 Assumptions . . . 561

4.28.5 R Implementation . . . 561
4.28.6 Output & Results . . . 561
4.28.7 Interpretation . . . 562
4.28.8 Practical Tips . . . 562
4.28.9 For Reviewers . . . 562
4.28.10 See also — labs in this chapter . . . 562
4.29 One-Sample t-Test . . . 563
4.29.1 Introduction . . . 563
4.29.2 Prerequisites . . . 563
4.29.3 Theory . . . 563
4.29.4 Assumptions . . . 563
4.29.5 R Implementation . . . 563
4.29.6 Output & Results . . . 564
4.29.7 Interpretation . . . 564
4.29.8 Practical Tips . . . 564
4.29.9 For Reviewers . . . 565
4.29.10 See also — labs in this chapter . . . 565
4.30 One-Sample z-Test . . . 565
4.30.1 Introduction . . . 565
4.30.2 Prerequisites . . . 565
4.30.3 Theory . . . 565
4.30.4 Assumptions . . . 566
4.30.5 R Implementation . . . 566
4.30.6 Output & Results . . . 566
4.30.7 Interpretation . . . 567
4.30.8 Practical Tips . . . 567
4.30.9 For Reviewers . . . 567
4.30.10 See also — labs in this chapter . . . 567
4.31 One-Way ANOVA . . . 568
4.31.1 Introduction . . . 568
4.31.2 Prerequisites . . . 568
4.31.3 Theory . . . 568
4.31.4 Assumptions . . . 568
4.31.5 R Implementation . . . 568
4.31.6 Output & Results . . . 569
4.31.7 Interpretation . . . 570
4.31.8 Practical Tips . . . 570
4.31.9 For Reviewers . . . 570
4.31.10 See also — labs in this chapter . . . 570
4.32 P-Values Explained . . . 570
4.32.1 Introduction . . . 570
4.32.2 Prerequisites . . . 571
4.32.3 Theory . . . 571
4.32.4 Assumptions . . . 571
4.32.5 R Implementation . . . 571
4.32.6 Output & Results . . . 572

4.32.7 Interpretation . . . 572
4.32.8 Practical Tips . . . 572
4.32.9 For Reviewers . . . 573
4.32.10 See also — labs in this chapter . . . 573
4.33 Paired t-Test . . . 573
4.33.1 Introduction . . . 573
4.33.2 Prerequisites . . . 573
4.33.3 Theory . . . 573
4.33.4 Assumptions . . . 574
4.33.5 R Implementation . . . 574
4.33.6 Output & Results . . . 574
4.33.7 Interpretation . . . 575
4.33.8 Practical Tips . . . 575
4.33.9 For Reviewers . . . 575
4.33.10 See also — labs in this chapter . . . 575
4.34 Pearson Correlation Test . . . 575
4.34.1 Introduction . . . 575
4.34.2 Prerequisites . . . 576
4.34.3 Theory . . . 576
4.34.4 Assumptions . . . 576
4.34.5 R Implementation . . . 576
4.34.6 Output & Results . . . 577
4.34.7 Interpretation . . . 577
4.34.8 Practical Tips . . . 577
4.34.9 For Reviewers . . . 577
4.34.10 See also — labs in this chapter . . . 578
4.35 Permutation Tests . . . 578
4.35.1 Introduction . . . 578
4.35.2 Prerequisites . . . 578
4.35.3 Theory . . . 578
4.35.4 Assumptions . . . 578
4.35.5 R Implementation . . . 579
4.35.6 Output & Results . . . 579
4.35.7 Interpretation . . . 579
4.35.8 Practical Tips . . . 580
4.35.9 For Reviewers . . . 580
4.35.10 See also — labs in this chapter . . . 580
4.36 Post-Hoc Tests with Tukey HSD . . . 580
4.36.1 Introduction . . . 580
4.36.2 Prerequisites . . . 580
4.36.3 Theory . . . 581
4.36.4 Assumptions . . . 581
4.36.5 R Implementation . . . 581
4.36.6 Output & Results . . . 582
4.36.7 Interpretation . . . 582
4.36.8 Practical Tips . . . 582

4.36.9 For Reviewers . . . 582
4.36.10 See also — labs in this chapter . . . 582
4.37 Repeated-Measures ANOVA . . . 583
4.37.1 Introduction . . . 583
4.37.2 Prerequisites . . . 583
4.37.3 Theory . . . 583
4.37.4 Assumptions . . . 583
4.37.5 R Implementation . . . 584
4.37.6 Output & Results . . . 584
4.37.7 Interpretation . . . 585
4.37.8 Practical Tips . . . 585
4.37.9 For Reviewers . . . 585
4.37.10 See also — labs in this chapter . . . 585
4.38 The Runs Test . . . 586
4.38.1 Introduction . . . 586
4.38.2 Prerequisites . . . 586
4.38.3 Theory . . . 586
4.38.4 Assumptions . . . 586
4.38.5 R Implementation . . . 587
4.38.6 Output & Results . . . 587
4.38.7 Interpretation . . . 587
4.38.8 Practical Tips . . . 587
4.38.9 R Packages Used . . . 588
4.38.10 For Reviewers . . . 588
4.38.11 See also — labs in this chapter . . . 588
4.39 Shapiro-Wilk Normality Test . . . 589
4.39.1 Introduction . . . 589
4.39.2 Prerequisites . . . 589
4.39.3 Theory . . . 589
4.39.4 Assumptions . . . 589
4.39.5 R Implementation . . . 589
4.39.6 Output & Results . . . 590
4.39.7 Interpretation . . . 590
4.39.8 Practical Tips . . . 590
4.39.9 R Packages Used . . . 591
4.39.10 For Reviewers . . . 591
4.39.11 See also — labs in this chapter . . . 591
4.40 The Sign Test . . . 592
4.40.1 Introduction . . . 592
4.40.2 Prerequisites . . . 592
4.40.3 Theory . . . 592
4.40.4 Assumptions . . . 592
4.40.5 R Implementation . . . 592
4.40.6 Output & Results . . . 593
4.40.7 Interpretation . . . 593
4.40.8 Practical Tips . . . 593

4.40.9 For Reviewers . . . 593
4.40.10 See also — labs in this chapter . . . 594
4.41 Spearman Rank Correlation Test . . . 594
4.41.1 Introduction . . . 594
4.41.2 Prerequisites . . . 594
4.41.3 Theory . . . 594
4.41.4 Assumptions . . . 595
4.41.5 R Implementation . . . 595
4.41.6 Output & Results . . . 595
4.41.7 Interpretation . . . 595
4.41.8 Practical Tips . . . 596
4.41.9 For Reviewers . . . 596
4.41.10 See also — labs in this chapter . . . 596
4.42 Two-Proportion Test . . . 596
4.42.1 Introduction . . . 596
4.42.2 Prerequisites . . . 596
4.42.3 Theory . . . 597
4.42.4 Assumptions . . . 597
4.42.5 R Implementation . . . 597
4.42.6 Output & Results . . . 598
4.42.7 Interpretation . . . 598
4.42.8 Practical Tips . . . 598
4.42.9 For Reviewers . . . 598
4.42.10 See also — labs in this chapter . . . 599
4.43 Two-Sample t-Test . . . 599
4.43.1 Introduction . . . 599
4.43.2 Prerequisites . . . 599
4.43.3 Theory . . . 599
4.43.4 Assumptions . . . 600
4.43.5 R Implementation . . . 600
4.43.6 Output & Results . . . 601
4.43.7 Interpretation . . . 601
4.43.8 Practical Tips . . . 601
4.43.9 For Reviewers . . . 601
4.43.10 See also — labs in this chapter . . . 602
4.44 Two-Way ANOVA . . . 602
4.44.1 Introduction . . . 602
4.44.2 Prerequisites . . . 602
4.44.3 Theory . . . 602
4.44.4 Assumptions . . . 603
4.44.5 R Implementation . . . 603
4.44.6 Output & Results . . . 603
4.44.7 Interpretation . . . 604
4.44.8 Practical Tips . . . 604
4.44.9 For Reviewers . . . 604
4.44.10 See also — labs in this chapter . . . 604

4.45 Type I and Type II Errors . . . 605
4.45.1 Introduction . . . 605
4.45.2 Prerequisites . . . 605
4.45.3 Theory . . . 605
4.45.4 Assumptions . . . 605
4.45.5 R Implementation . . . 605
4.45.6 Output & Results . . . 606
4.45.7 Interpretation . . . 606
4.45.8 Practical Tips . . . 606
4.45.9 For Reviewers . . . 607
4.45.10 See also — labs in this chapter . . . 607
4.46 Welch's t-Test . . . 607
4.46.1 Introduction . . . 607
4.46.2 Prerequisites . . . 607
4.46.3 Theory . . . 607
4.46.4 Assumptions . . . 608
4.46.5 R Implementation . . . 608
4.46.6 Output & Results . . . 608
4.46.7 Interpretation . . . 609
4.46.8 Practical Tips . . . 609
4.46.9 For Reviewers . . . 609
4.46.10 See also — labs in this chapter . . . 609
4.47 Wilcoxon Signed-Rank Test . . . 610
4.47.1 Introduction . . . 610
4.47.2 Prerequisites . . . 610
4.47.3 Theory . . . 610
4.47.4 Assumptions . . . 610
4.47.5 R Implementation . . . 611
4.47.6 Output & Results . . . 611
4.47.7 Interpretation . . . 611
4.47.8 Practical Tips . . . 611
4.47.9 R Packages Used . . . 612
4.47.10 For Reviewers . . . 612
4.47.11 See also — labs in this chapter . . . 612
4.48 Week 3, Session 2 — Bootstrap and permutation tests . . . 612
4.48.1 Learning objectives . . . 613
4.48.2 Prerequisites . . . 613
4.48.3 Background . . . 613
4.48.4 Setup . . . 613
4.48.5 1. Hypothesis . . . 613
4.48.6 2. Visualise . . . 614
4.48.7 3. Assumptions . . . 614
4.48.8 4. Conduct . . . 614
4.48.9 5. Concluding statement . . . 615
4.48.10 Common pitfalls . . . 615
4.48.11 Further reading . . . 615

4.48.12 Session info . . . 616
4.48.13 See also — chapter index . . . 616
4.49 Week 3, Session 5 — Hypothesis testing, p-values, type I/II errors . . . 616
4.49.1 Learning objectives . . . 616
4.49.2 Prerequisites . . . 616
4.49.3 Background . . . 616
4.49.4 Setup . . . 617
4.49.5 1. Hypothesis . . . 617
4.49.6 2. Visualise . . . 617
4.49.7 3. Assumptions . . . 617
4.49.8 4. Conduct . . . 618
4.49.9 5. Concluding statement . . . 618
4.49.10 Common pitfalls . . . 619
4.49.11 Further reading . . . 619
4.49.12 Session info . . . 619
4.49.13 See also — chapter index . . . 619
4.50 Week 3, Session 3 — Maximum likelihood estimation . . . 619
4.50.1 Learning objectives . . . 619
4.50.2 Prerequisites . . . 619
4.50.3 Background . . . 620
4.50.4 Setup . . . 620
4.50.5 1. Hypothesis . . . 620
4.50.6 2. Visualise . . . 620
4.50.7 3. Assumptions . . . 621
4.50.8 4. Conduct . . . 621
4.50.9 5. Concluding statement . . . 622
4.50.10 Common pitfalls . . . 622
4.50.11 Further reading . . . 622
4.50.12 Session info . . . 622
4.50.13 See also — chapter index . . . 623
4.51 Week 4, Session 4 — Non-parametric tests . . . 623
4.51.1 Learning objectives . . . 623
4.51.2 Prerequisites . . . 623
4.51.3 Background . . . 623
4.51.4 Setup . . . 623
4.51.5 1. Hypothesis . . . 624
4.51.6 2. Visualise . . . 624
4.51.7 3. Assumptions . . . 624
4.51.8 4. Conduct . . . 625
4.51.9 5. Concluding statement . . . 625
4.51.10 Common pitfalls . . . 626
4.51.11 Further reading . . . 626
4.51.12 Session info . . . 626
4.51.13 See also — chapter index . . . 626

4.52 Week 3, Session 4 — One-sample t-test and one-proportion test . . . 626
4.52.1 Learning objectives . . . 626
4.52.2 Prerequisites . . . 626
4.52.3 Background . . . 626
4.52.4 Setup . . . 627
4.52.5 1. Hypothesis . . . 627
4.52.6 2. Visualise . . . 627
4.52.7 3. Assumptions . . . 628
4.52.8 4. Conduct . . . 628
4.52.9 5. Concluding statement . . . 628
4.52.10 Common pitfalls . . . 629
4.52.11 Further reading . . . 629
4.52.12 Session info . . . 629
4.52.13 See also — chapter index . . . 629
4.53 Week 4, Session 3 — Pearson and Spearman correlation . . . 629
4.53.1 Learning objectives . . . 629
4.53.2 Prerequisites . . . 629
4.53.3 Background . . . 630
4.53.4 Setup . . . 630
4.53.5 1. Hypothesis . . . 630
4.53.6 2. Visualise . . . 630
4.53.7 3. Assumptions . . . 631
4.53.8 4. Conduct . . . 631
4.53.9 5. Concluding statement . . . 632
4.53.10 Common pitfalls . . . 632
4.53.11 Further reading . . . 632
4.53.12 Session info . . . 632
4.53.13 See also — chapter index . . . 632
4.54 Week 3, Session 1 — Populations, samples, and the central limit theorem . . . 633
4.54.1 Learning objectives . . . 633
4.54.2 Prerequisites . . . 633
4.54.3 Background . . . 633
4.54.4 Setup . . . 633
4.54.5 1. Hypothesis . . . 634
4.54.6 2. Visualise . . . 634
4.54.7 3. Assumptions . . . 634
4.54.8 4. Conduct . . . 634
4.54.9 5. Concluding statement . . . 635
4.54.10 Common pitfalls . . . 635
4.54.11 Further reading . . . 636
4.54.12 Session info . . . 636
4.54.13 See also — chapter index . . . 636
4.55 Week 4, Session 2 — Two proportions, chi-square, risk and odds . . . 636

4.55.1 Learning objectives . . . 636
4.55.2 Prerequisites . . . 636
4.55.3 Background . . . 636
4.55.4 Setup . . . 637
4.55.5 1. Hypothesis . . . 637
4.55.6 2. Visualise . . . 637
4.55.7 3. Assumptions . . . 638
4.55.8 4. Conduct . . . 638
4.55.9 5. Concluding statement . . . 639
4.55.10 Common pitfalls . . . 639
4.55.11 Further reading . . . 639
4.55.12 Session info . . . 639
4.55.13 See also — chapter index . . . 639
4.56 Week 4, Session 1 — Two-sample and paired t-tests . . . 640
4.56.1 Learning objectives . . . 640
4.56.2 Prerequisites . . . 640
4.56.3 Background . . . 640
4.56.4 Setup . . . 640
4.56.5 1. Hypothesis . . . 640
4.56.6 2. Visualise . . . 641
4.56.7 3. Assumptions . . . 641
4.56.8 4. Conduct . . . 642
4.56.9 5. Concluding statement . . . 642
4.56.10 Common pitfalls . . . 643
4.56.11 Further reading . . . 643
4.56.12 Session info . . . 643
4.56.13 See also — chapter index . . . 643

5 Sample Size & Power 645
5.1 Method pages . . . 645
5.2 Labs . . . 646
5.3 Effect Size: Cohen's d . . . 646
5.3.1 Introduction . . . 646
5.3.2 Prerequisites . . . 646
5.3.3 Theory . . . 646
5.3.4 Assumptions . . . 647
5.3.5 R Implementation . . . 647
5.3.6 Output & Results . . . 647
5.3.7 Interpretation . . . 647
5.3.8 Practical Tips . . . 648
5.3.9 For Reviewers . . . 648
5.3.10 See also — labs in this chapter . . . 648
5.4 Effect Size: Cohen's h . . . 648
5.4.1 Introduction . . . 648
5.4.2 Prerequisites . . . 648
5.4.3 Theory . . . 649

5.4.4 Assumptions . . . 649
5.4.5 R Implementation . . . 649
5.4.6 Output & Results . . . 649
5.4.7 Interpretation . . . 649
5.4.8 Practical Tips . . . 650
5.4.9 R Packages Used . . . 650
5.4.10 For Reviewers . . . 650
5.4.11 See also — labs in this chapter . . . 651
5.5 Effect Size: Eta-Squared . . . 651
5.5.1 Introduction . . . 651
5.5.2 Prerequisites . . . 651
5.5.3 Theory . . . 651
5.5.4 Assumptions . . . 652
5.5.5 R Implementation . . . 652
5.5.6 Output & Results . . . 652
5.5.7 Interpretation . . . 653
5.5.8 Practical Tips . . . 653
5.5.9 R Packages Used . . . 653
5.5.10 For Reviewers . . . 653
5.5.11 See also — labs in this chapter . . . 654
5.6 Minimum Detectable Effect . . . 654
5.6.1 Introduction . . . 654
5.6.2 Prerequisites . . . 654
5.6.3 Theory . . . 654
5.6.4 Assumptions . . . 654
5.6.5 R Implementation . . . 655
5.6.6 Output & Results . . . 655
5.6.7 Interpretation . . . 655
5.6.8 Practical Tips . . . 655
5.6.9 R Packages Used . . . 656
5.6.10 For Reviewers . . . 656
5.6.11 See also — labs in this chapter . . . 656
5.7 Post-Hoc Power: A Controversy . . . 656
5.7.1 Introduction . . . 656
5.7.2 Prerequisites . . . 657
5.7.3 Theory . . . 657
5.7.4 Assumptions . . . 657
5.7.5 R Implementation . . . 657
5.7.6 Output & Results . . . 658
5.7.7 Interpretation . . . 658
5.7.8 Practical Tips . . . 658
5.7.9 For Reviewers . . . 658
5.7.10 See also — labs in this chapter . . . 659
5.8 Power for Agreement (Kappa) . . . 659
5.8.1 Introduction . . . 659
5.8.2 Prerequisites . . . 659

5.8.3 Theory . . . 659
5.8.4 Assumptions . . . 659
5.8.5 R Implementation . . . 660
5.8.6 Output & Results . . . 660
5.8.7 Interpretation . . . 660
5.8.8 Practical Tips . . . 660
5.8.9 R Packages Used . . . 661
5.8.10 For Reviewers . . . 661
5.8.11 See also — labs in this chapter . . . 661
5.9 Power Analysis: Introduction . . . 661
5.9.1 Introduction . . . 661
5.9.2 Prerequisites . . . 661
5.9.3 Theory . . . 662
5.9.4 Assumptions . . . 662
5.9.5 R Implementation . . . 662
5.9.6 Output & Results . . . 663
5.9.7 Interpretation . . . 663
5.9.8 Practical Tips . . . 663
5.9.9 For Reviewers . . . 664
5.9.10 See also — labs in this chapter . . . 664
5.10 Power for One-Way ANOVA . . . 664
5.10.1 Introduction . . . 664
5.10.2 Prerequisites . . . 664
5.10.3 Theory . . . 664
5.10.4 Assumptions . . . 665
5.10.5 R Implementation . . . 665
5.10.6 Output & Results . . . 665
5.10.7 Interpretation . . . 665
5.10.8 Practical Tips . . . 665
5.10.9 R Packages Used . . . 666
5.10.10 For Reviewers . . . 666
5.10.11 See also — labs in this chapter . . . 666
5.11 Power for Bland-Altman Studies . . . 667
5.11.1 Introduction . . . 667
5.11.2 Prerequisites . . . 667
5.11.3 Theory . . . 667
5.11.4 Assumptions . . . 667
5.11.5 R Implementation . . . 668
5.11.6 Output & Results . . . 668
5.11.7 Interpretation . . . 668
5.11.8 Practical Tips . . . 668
5.11.9 R Packages Used . . . 669
5.11.10 For Reviewers . . . 669
5.11.11 See also — labs in this chapter . . . 669
5.12 Power for Chi-Squared Tests . . . 669
5.12.1 Introduction . . . 669

5.12.2 Prerequisites 670
5.12.3 Theory 670
5.12.4 Assumptions 670
5.12.5 R Implementation 670
5.12.6 Output & Results 671
5.12.7 Interpretation 671
5.12.8 Practical Tips 671
5.12.9 R Packages Used 672
5.12.10 For Reviewers 672
5.12.11 See also — labs in this chapter 672
5.13 Power for Cluster-RCT 672
5.13.1 Introduction 672
5.13.2 Prerequisites 672
5.13.3 Theory 672
5.13.4 Assumptions 673
5.13.5 R Implementation 673
5.13.6 Output & Results 673
5.13.7 Interpretation 673
5.13.8 Practical Tips 673
5.13.9 For Reviewers 674
5.13.10 See also — labs in this chapter 674
5.14 Power for Correlation Tests 674
5.14.1 Introduction 674
5.14.2 Prerequisites 674
5.14.3 Theory 674
5.14.4 Assumptions 675
5.14.5 R Implementation 675
5.14.6 Output & Results 675
5.14.7 Interpretation 675
5.14.8 Practical Tips 676
5.14.9 R Packages Used 676
5.14.10 For Reviewers 676
5.14.11 See also — labs in this chapter 676
5.15 Power for Cox Regression 677
5.15.1 Introduction 677
5.15.2 Prerequisites 677
5.15.3 Theory 677
5.15.4 Assumptions 677
5.15.5 R Implementation 677
5.15.6 Output & Results 678
5.15.7 Interpretation 678
5.15.8 Practical Tips 678
5.15.9 For Reviewers 678
5.15.10 See also — labs in this chapter 678
5.16 Power for Crossover Trials 679
5.16.1 Introduction 679

5.16.2 Prerequisites . . . 679
5.16.3 Theory . . . 679
5.16.4 Assumptions . . . 679
5.16.5 R Implementation . . . 679
5.16.6 Output & Results . . . 680
5.16.7 Interpretation . . . 680
5.16.8 Practical Tips . . . 680
5.16.9 For Reviewers . . . 680
5.16.10 See also — labs in this chapter . . . 680
5.17 Power for Diagnostic Accuracy . . . 680
5.17.1 Introduction . . . 680
5.17.2 Prerequisites . . . 681
5.17.3 Theory . . . 681
5.17.4 Assumptions . . . 681
5.17.5 R Implementation . . . 681
5.17.6 Output & Results . . . 681
5.17.7 Interpretation . . . 682
5.17.8 Practical Tips . . . 682
5.17.9 For Reviewers . . . 682
5.17.10 See also — labs in this chapter . . . 682
5.18 Power for Equivalence (TOST) . . . 682
5.18.1 Introduction . . . 682
5.18.2 Prerequisites . . . 683
5.18.3 Theory . . . 683
5.18.4 Assumptions . . . 683
5.18.5 R Implementation . . . 683
5.18.6 Output & Results . . . 683
5.18.7 Interpretation . . . 684
5.18.8 Practical Tips . . . 684
5.18.9 R Packages Used . . . 684
5.18.10 For Reviewers . . . 685
5.18.11 See also — labs in this chapter . . . 685
5.19 Power for ICC . . . 685
5.19.1 Introduction . . . 685
5.19.2 Prerequisites . . . 685
5.19.3 Theory . . . 685
5.19.4 Assumptions . . . 686
5.19.5 R Implementation . . . 686
5.19.6 Output & Results . . . 686
5.19.7 Interpretation . . . 686
5.19.8 Practical Tips . . . 687
5.19.9 R Packages Used . . . 687
5.19.10 For Reviewers . . . 687
5.19.11 See also — labs in this chapter . . . 687
5.20 Power for Linear Regression . . . 688
5.20.1 Introduction . . . 688

5.20.2 Prerequisites . . . 688
5.20.3 Theory . . . 688
5.20.4 Assumptions . . . 688
5.20.5 R Implementation . . . 689
5.20.6 Output & Results . . . 689
5.20.7 Interpretation . . . 689
5.20.8 Practical Tips . . . 689
5.20.9 R Packages Used . . . 690
5.20.10 For Reviewers . . . 690
5.20.11 See also — labs in this chapter . . . 690
5.21 Power for Logistic Regression . . . 690
5.21.1 Introduction . . . 690
5.21.2 Prerequisites . . . 690
5.21.3 Theory . . . 690
5.21.4 Assumptions . . . 691
5.21.5 R Implementation . . . 691
5.21.6 Output & Results . . . 691
5.21.7 Interpretation . . . 691
5.21.8 Practical Tips . . . 692
5.21.9 For Reviewers . . . 692
5.21.10 See also — labs in this chapter . . . 692
5.22 Power for the Log-Rank Test . . . 692
5.22.1 Introduction . . . 692
5.22.2 Prerequisites . . . 692
5.22.3 Theory . . . 693
5.22.4 Assumptions . . . 693
5.22.5 R Implementation . . . 693
5.22.6 Output & Results . . . 693
5.22.7 Interpretation . . . 694
5.22.8 Practical Tips . . . 694
5.22.9 R Packages Used . . . 694
5.22.10 For Reviewers . . . 695
5.22.11 See also — labs in this chapter . . . 695
5.23 Power for McNemar's Test . . . 695
5.23.1 Introduction . . . 695
5.23.2 Prerequisites . . . 695
5.23.3 Theory . . . 695
5.23.4 Assumptions . . . 695
5.23.5 R Implementation . . . 696
5.23.6 Output & Results . . . 696
5.23.7 Interpretation . . . 696
5.23.8 Practical Tips . . . 696
5.23.9 Reporting . . . 696
5.23.10 For Reviewers . . . 697
5.23.11 See also — labs in this chapter . . . 697
5.24 Power for Non-Inferiority Trials . . . 697

5.24.1 Introduction . . . 697
5.24.2 Prerequisites . . . 697
5.24.3 Theory . . . 697
5.24.4 Assumptions . . . 698
5.24.5 R Implementation . . . 698
5.24.6 Output & Results . . . 698
5.24.7 Interpretation . . . 698
5.24.8 Practical Tips . . . 698
5.24.9 For Reviewers . . . 699
5.24.10 See also — labs in this chapter . . . 699
5.25 Power for One-Proportion Test . . . 699
5.25.1 Introduction . . . 699
5.25.2 Prerequisites . . . 699
5.25.3 Theory . . . 699
5.25.4 Assumptions . . . 700
5.25.5 R Implementation . . . 700
5.25.6 Output & Results . . . 700
5.25.7 Interpretation . . . 701
5.25.8 Practical Tips . . . 701
5.25.9 R Packages Used . . . 701
5.25.10 For Reviewers . . . 702
5.25.11 See also — labs in this chapter . . . 702
5.26 Power for One-Sample t-Test . . . 702
5.26.1 Introduction . . . 702
5.26.2 Prerequisites . . . 702
5.26.3 Theory . . . 702
5.26.4 Assumptions . . . 703
5.26.5 R Implementation . . . 703
5.26.6 Output & Results . . . 703
5.26.7 Interpretation . . . 703
5.26.8 Practical Tips . . . 704
5.26.9 R Packages Used . . . 704
5.26.10 For Reviewers . . . 704
5.26.11 See also — labs in this chapter . . . 704
5.27 Power for Paired t-Test . . . 705
5.27.1 Introduction . . . 705
5.27.2 Prerequisites . . . 705
5.27.3 Theory . . . 705
5.27.4 Assumptions . . . 705
5.27.5 R Implementation . . . 706
5.27.6 Output & Results . . . 706
5.27.7 Interpretation . . . 706
5.27.8 Practical Tips . . . 706
5.27.9 R Packages Used . . . 707
5.27.10 For Reviewers . . . 707
5.27.11 See also — labs in this chapter . . . 707

5.28 Power for Repeated-Measures ANOVA 707
5.28.1 Introduction 707
5.28.2 Prerequisites 708
5.28.3 Theory 708
5.28.4 Assumptions 708
5.28.5 R Implementation 708
5.28.6 Output & Results 709
5.28.7 Interpretation 709
5.28.8 Practical Tips 709
5.28.9 R Packages Used 710
5.28.10 For Reviewers 710
5.28.11 See also — labs in this chapter 710
5.29 Power for Stepped-Wedge Trials 710
5.29.1 Introduction 710
5.29.2 Prerequisites 710
5.29.3 Theory 710
5.29.4 Assumptions 711
5.29.5 R Implementation 711
5.29.6 Output & Results 711
5.29.7 Interpretation 711
5.29.8 Practical Tips 711
5.29.9 For Reviewers 712
5.29.10 See also — labs in this chapter 712
5.30 Power for Two-Proportion Test 712
5.30.1 Introduction 712
5.30.2 Prerequisites 712
5.30.3 Theory 712
5.30.4 Assumptions 713
5.30.5 R Implementation 713
5.30.6 Output & Results 713
5.30.7 Interpretation 713
5.30.8 Practical Tips 714
5.30.9 R Packages Used 714
5.30.10 For Reviewers 714
5.30.11 See also — labs in this chapter 714
5.31 Sample Size for a Two-Sample t-Test 715
5.31.1 Introduction 715
5.31.2 Prerequisites 715
5.31.3 Theory 715
5.31.4 Assumptions 715
5.31.5 R Implementation 716
5.31.6 Output & Results 716
5.31.7 Interpretation 716
5.31.8 Practical Tips 717
5.31.9 For Reviewers 717
5.31.10 See also — labs in this chapter 717

5.32 Sample Size Sensitivity Analysis . . . 717
5.32.1 Introduction . . . 717
5.32.2 Prerequisites . . . 717
5.32.3 Theory . . . 717
5.32.4 Assumptions . . . 718
5.32.5 R Implementation . . . 718
5.32.6 Output & Results . . . 718
5.32.7 Interpretation . . . 718
5.32.8 Practical Tips . . . 719
5.32.9 Reporting . . . 719
5.32.10 For Reviewers . . . 719
5.32.11 See also — labs in this chapter . . . 719
5.33 Week 1, Session 5 — Power: closed-form and simulation . 719
5.33.1 Learning objectives . . . 720
5.33.2 Prerequisites . . . 720
5.33.3 Background . . . 720
5.33.4 Setup . . . 720
5.33.5 1. Hypothesis . . . 720
5.33.6 2. Visualise . . . 721
5.33.7 3. Assumptions . . . 721
5.33.8 4. Conduct . . . 721
5.33.9 5. Concluding statement . . . 722
5.33.10 Common pitfalls . . . 722
5.33.11 Further reading . . . 722
5.33.12 Session info . . . 722
5.33.13 See also — chapter index . . . 722
5.34 Week 4, Session 5 — Sample size, power, and Quarto reporting 722
5.34.1 Learning objectives . . . 723
5.34.2 Prerequisites . . . 723
5.34.3 Background . . . 723
5.34.4 Setup . . . 723
5.34.5 1. Goal . . . 724
5.34.6 2. Approach . . . 724
5.34.7 3. Execution . . . 724
5.34.8 4. Check . . . 725
5.34.9 5. Report . . . 725
5.34.10 Common pitfalls . . . 725
5.34.11 Further reading . . . 726
5.34.12 Session info . . . 726
5.34.13 See also — chapter index . . . 726

6 Data Visualisation 727
6.1 Method pages . . . 727
6.2 Labs . . . 728
6.3 Aesthetics and Geoms . . . 728
6.3.1 Introduction . . . 728

6.3.2 Prerequisites . . . 728
6.3.3 Theory . . . 729
6.3.4 Assumptions . . . 729
6.3.5 R Implementation . . . 729
6.3.6 Output & Results . . . 730
6.3.7 Interpretation . . . 730
6.3.8 Practical Tips . . . 730
6.3.9 R Packages Used . . . 731
6.3.10 For Reviewers . . . 731
6.3.11 See also — labs in this chapter . . . 731
6.4 Annotations and Labels . . . 731
6.4.1 Introduction . . . 731
6.4.2 Prerequisites . . . 731
6.4.3 Theory . . . 731
6.4.4 Assumptions . . . 732
6.4.5 R Implementation . . . 732
6.4.6 Output & Results . . . 732
6.4.7 Interpretation . . . 733
6.4.8 Practical Tips . . . 733
6.4.9 R Packages Used . . . 733
6.4.10 For Reviewers . . . 733
6.4.11 See also — labs in this chapter . . . 734
6.5 Bar Charts . . . 734
6.5.1 Introduction . . . 734
6.5.2 Prerequisites . . . 734
6.5.3 Theory . . . 734
6.5.4 Assumptions . . . 734
6.5.5 R Implementation . . . 735
6.5.6 Output & Results . . . 735
6.5.7 Interpretation . . . 735
6.5.8 Practical Tips . . . 735
6.5.9 R Packages Used . . . 736
6.5.10 For Reviewers . . . 736
6.5.11 See also — labs in this chapter . . . 736
6.6 Bland-Altman Plots . . . 736
6.6.1 Introduction . . . 736
6.6.2 Prerequisites . . . 736
6.6.3 Theory . . . 737
6.6.4 Assumptions . . . 737
6.6.5 R Implementation . . . 737
6.6.6 Output & Results . . . 738
6.6.7 Interpretation . . . 738
6.6.8 Practical Tips . . . 738
6.6.9 R Packages Used . . . 738
6.6.10 For Reviewers . . . 738
6.6.11 See also — labs in this chapter . . . 739

6.7 Boxplots . . . 739
6.7.1 Introduction . . . 739
6.7.2 Prerequisites . . . 739
6.7.3 Theory . . . 739
6.7.4 Assumptions . . . 740
6.7.5 R Implementation . . . 740
6.7.6 Output & Results . . . 740
6.7.7 Interpretation . . . 740
6.7.8 Practical Tips . . . 741
6.7.9 R Packages Used . . . 741
6.7.10 For Reviewers . . . 741
6.7.11 See also — labs in this chapter . . . 741
6.8 Bubble Plots . . . 741
6.8.1 Introduction . . . 741
6.8.2 Prerequisites . . . 742
6.8.3 Theory . . . 742
6.8.4 Assumptions . . . 742
6.8.5 R Implementation . . . 742
6.8.6 Output & Results . . . 742
6.8.7 Interpretation . . . 742
6.8.8 Practical Tips . . . 743
6.8.9 R Packages Used . . . 743
6.8.10 For Reviewers . . . 743
6.8.11 See also — labs in this chapter . . . 743
6.9 Colour-Blind-Safe Plots . . . 743
6.9.1 Introduction . . . 743
6.9.2 Prerequisites . . . 744
6.9.3 Theory . . . 744
6.9.4 Assumptions . . . 744
6.9.5 R Implementation . . . 744
6.9.6 Output & Results . . . 745
6.9.7 Interpretation . . . 745
6.9.8 Practical Tips . . . 745
6.9.9 R Packages Used . . . 746
6.9.10 For Reviewers . . . 746
6.9.11 See also — labs in this chapter . . . 746
6.10 Colour Palettes . . . 746
6.10.1 Introduction . . . 746
6.10.2 Prerequisites . . . 746
6.10.3 Theory . . . 746
6.10.4 Assumptions . . . 747
6.10.5 R Implementation . . . 747
6.10.6 Output & Results . . . 747
6.10.7 Interpretation . . . 748
6.10.8 Practical Tips . . . 748
6.10.9 R Packages Used . . . 748

6.10.10 For Reviewers . . . 748
6.10.11 See also — labs in this chapter . . . 749
6.11 Contour Plots . . . 749
6.11.1 Introduction . . . 749
6.11.2 Prerequisites . . . 749
6.11.3 Theory . . . 749
6.11.4 Assumptions . . . 749
6.11.5 R Implementation . . . 750
6.11.6 Output & Results . . . 750
6.11.7 Interpretation . . . 750
6.11.8 Practical Tips . . . 750
6.11.9 R Packages Used . . . 751
6.11.10 For Reviewers . . . 751
6.11.11 See also — labs in this chapter . . . 751
6.12 Correlation Heatmaps . . . 751
6.12.1 Introduction . . . 751
6.12.2 Prerequisites . . . 752
6.12.3 Theory . . . 752
6.12.4 Assumptions . . . 752
6.12.5 R Implementation . . . 752
6.12.6 Output & Results . . . 753
6.12.7 Interpretation . . . 753
6.12.8 Practical Tips . . . 753
6.12.9 R Packages Used . . . 753
6.12.10 For Reviewers . . . 753
6.12.11 See also — labs in this chapter . . . 754
6.13 Density Plots . . . 754
6.13.1 Introduction . . . 754
6.13.2 Prerequisites . . . 754
6.13.3 Theory . . . 754
6.13.4 Assumptions . . . 755
6.13.5 R Implementation . . . 755
6.13.6 Output & Results . . . 755
6.13.7 Interpretation . . . 756
6.13.8 Practical Tips . . . 756
6.13.9 R Packages Used . . . 756
6.13.10 For Reviewers . . . 756
6.13.11 See also — labs in this chapter . . . 756
6.14 Dot Plots . . . 757
6.14.1 Introduction . . . 757
6.14.2 Prerequisites . . . 757
6.14.3 Theory . . . 757
6.14.4 Assumptions . . . 757
6.14.5 R Implementation . . . 757
6.14.6 Output & Results . . . 758
6.14.7 Interpretation . . . 758

6.14.8 Practical Tips . . . 758
6.14.9 R Packages Used . . . 759
6.14.10 For Reviewers . . . 759
6.14.11 See also — labs in this chapter . . . 759
6.15 Facets and Panels . . . 759
6.15.1 Introduction . . . 759
6.15.2 Prerequisites . . . 759
6.15.3 Theory . . . 759
6.15.4 Assumptions . . . 760
6.15.5 R Implementation . . . 760
6.15.6 Output & Results . . . 760
6.15.7 Interpretation . . . 761
6.15.8 Practical Tips . . . 761
6.15.9 R Packages Used . . . 761
6.15.10 For Reviewers . . . 761
6.15.11 See also — labs in this chapter . . . 761
6.16 Forest Plots (Visualisation) . . . 762
6.16.1 Introduction . . . 762
6.16.2 Prerequisites . . . 762
6.16.3 Theory . . . 762
6.16.4 Assumptions . . . 762
6.16.5 R Implementation . . . 762
6.16.6 Output & Results . . . 763
6.16.7 Interpretation . . . 763
6.16.8 Practical Tips . . . 763
6.16.9 R Packages Used . . . 764
6.16.10 For Reviewers . . . 764
6.16.11 See also — labs in this chapter . . . 764
6.17 Funnel Plots (Visualisation) . . . 764
6.17.1 Introduction . . . 764
6.17.2 Prerequisites . . . 764
6.17.3 Theory . . . 765
6.17.4 Assumptions . . . 765
6.17.5 R Implementation . . . 765
6.17.6 Output & Results . . . 765
6.17.7 Interpretation . . . 766
6.17.8 Practical Tips . . . 766
6.17.9 R Packages Used . . . 766
6.17.10 For Reviewers . . . 766
6.17.11 See also — labs in this chapter . . . 766
6.18 ggplot2 Themes . . . 767
6.18.1 Introduction . . . 767
6.18.2 Prerequisites . . . 767
6.18.3 Theory . . . 767
6.18.4 Assumptions . . . 767
6.18.5 R Implementation . . . 767

6.18.6 Output & Results . . . 768
6.18.7 Interpretation . . . 768
6.18.8 Practical Tips . . . 768
6.18.9 R Packages Used . . . 769
6.18.10 For Reviewers . . . 769
6.18.11 See also — labs in this chapter . . . 769
6.19 Saving and Exporting Figures . . . 769
6.19.1 Introduction . . . 769
6.19.2 Prerequisites . . . 770
6.19.3 Theory . . . 770
6.19.4 Assumptions . . . 770
6.19.5 R Implementation . . . 770
6.19.6 Output & Results . . . 771
6.19.7 Interpretation . . . 771
6.19.8 Practical Tips . . . 771
6.19.9 R Packages Used . . . 771
6.19.10 For Reviewers . . . 772
6.19.11 See also — labs in this chapter . . . 772
6.20 The Grammar of Graphics . . . 772
6.20.1 Introduction . . . 772
6.20.2 Prerequisites . . . 772
6.20.3 Theory . . . 772
6.20.4 R Implementation . . . 773
6.20.5 Output & Results . . . 773
6.20.6 Interpretation . . . 774
6.20.7 Practical Tips . . . 774
6.20.8 For Reviewers . . . 774
6.20.9 See also — labs in this chapter . . . 774
6.21 Heatmaps . . . 775
6.21.1 Introduction . . . 775
6.21.2 Prerequisites . . . 775
6.21.3 Theory . . . 775
6.21.4 Assumptions . . . 775
6.21.5 R Implementation . . . 775
6.21.6 Output & Results . . . 776
6.21.7 Interpretation . . . 776
6.21.8 Practical Tips . . . 776
6.21.9 R Packages Used . . . 777
6.21.10 For Reviewers . . . 777
6.21.11 See also — labs in this chapter . . . 777
6.22 Hexbin Plots . . . 777
6.22.1 Introduction . . . 777
6.22.2 Prerequisites . . . 777
6.22.3 Theory . . . 777
6.22.4 Assumptions . . . 778
6.22.5 R Implementation . . . 778

6.22.6 Output & Results . . . 778
6.22.7 Interpretation . . . 778
6.22.8 Practical Tips . . . 779
6.22.9 R Packages Used . . . 779
6.22.10 For Reviewers . . . 779
6.22.11 See also — labs in this chapter . . . 779
6.23 Histograms . . . 779
6.23.1 Introduction . . . 779
6.23.2 Prerequisites . . . 780
6.23.3 Theory . . . 780
6.23.4 Assumptions . . . 780
6.23.5 R Implementation . . . 780
6.23.6 Output & Results . . . 781
6.23.7 Interpretation . . . 781
6.23.8 Practical Tips . . . 781
6.23.9 R Packages Used . . . 782
6.23.10 For Reviewers . . . 782
6.23.11 See also — labs in this chapter . . . 782
6.24 Interactive Plots with ggiraph . . . 782
6.24.1 Introduction . . . 782
6.24.2 Prerequisites . . . 782
6.24.3 Theory . . . 782
6.24.4 Assumptions . . . 783
6.24.5 R Implementation . . . 783
6.24.6 Output & Results . . . 783
6.24.7 Interpretation . . . 783
6.24.8 Practical Tips . . . 783
6.24.9 R Packages Used . . . 784
6.24.10 For Reviewers . . . 784
6.24.11 See also — labs in this chapter . . . 784
6.25 Interactive Plots with plotly . . . 784
6.25.1 Introduction . . . 784
6.25.2 Prerequisites . . . 785
6.25.3 Theory . . . 785
6.25.4 Assumptions . . . 785
6.25.5 R Implementation . . . 785
6.25.6 Output & Results . . . 785
6.25.7 Interpretation . . . 786
6.25.8 Practical Tips . . . 786
6.25.9 R Packages Used . . . 786
6.25.10 For Reviewers . . . 786
6.25.11 See also — labs in this chapter . . . 787
6.26 Line Plots . . . 787
6.26.1 Introduction . . . 787
6.26.2 Prerequisites . . . 787
6.26.3 Theory . . . 787

6.26.4 Assumptions . . . 787
6.26.5 R Implementation . . . 788
6.26.6 Output & Results . . . 788
6.26.7 Interpretation . . . 788
6.26.8 Practical Tips . . . 789
6.26.9 R Packages Used . . . 789
6.26.10 For Reviewers . . . 789
6.26.11 See also — labs in this chapter . . . 789
6.27 Pairs Plots . . . 789
6.27.1 Introduction . . . 789
6.27.2 Prerequisites . . . 790
6.27.3 Theory . . . 790
6.27.4 Assumptions . . . 790
6.27.5 R Implementation . . . 790
6.27.6 Output & Results . . . 790
6.27.7 Interpretation . . . 791
6.27.8 Practical Tips . . . 791
6.27.9 R Packages Used . . . 791
6.27.10 For Reviewers . . . 791
6.27.11 See also — labs in this chapter . . . 791
6.28 Patchwork: Multi-Plot Composition . . . 792
6.28.1 Introduction . . . 792
6.28.2 Prerequisites . . . 792
6.28.3 Theory . . . 792
6.28.4 Assumptions . . . 792
6.28.5 R Implementation . . . 793
6.28.6 Output & Results . . . 793
6.28.7 Interpretation . . . 793
6.28.8 Practical Tips . . . 793
6.28.9 R Packages Used . . . 794
6.28.10 For Reviewers . . . 794
6.28.11 See also — labs in this chapter . . . 794
6.29 Raincloud Plots . . . 794
6.29.1 Introduction . . . 794
6.29.2 Prerequisites . . . 794
6.29.3 Theory . . . 795
6.29.4 Assumptions . . . 795
6.29.5 R Implementation . . . 795
6.29.6 Output & Results . . . 795
6.29.7 Interpretation . . . 795
6.29.8 Practical Tips . . . 796
6.29.9 R Packages Used . . . 796
6.29.10 For Reviewers . . . 796
6.29.11 See also — labs in this chapter . . . 796
6.30 Ridge Plots . . . 796
6.30.1 Introduction . . . 796

6.30.2 Prerequisites . . . 797
6.30.3 Theory . . . 797
6.30.4 Assumptions . . . 797
6.30.5 R Implementation . . . 797
6.30.6 Output & Results . . . 798
6.30.7 Interpretation . . . 798
6.30.8 Practical Tips . . . 798
6.30.9 R Packages Used . . . 798
6.30.10 For Reviewers . . . 798
6.30.11 See also — labs in this chapter . . . 799
6.31 ROC Curves . . . 799
6.31.1 Introduction . . . 799
6.31.2 Prerequisites . . . 799
6.31.3 Theory . . . 799
6.31.4 Assumptions . . . 799
6.31.5 R Implementation . . . 799
6.31.6 Output & Results . . . 800
6.31.7 Interpretation . . . 800
6.31.8 Practical Tips . . . 800
6.31.9 R Packages Used . . . 801
6.31.10 For Reviewers . . . 801
6.31.11 See also — labs in this chapter . . . 801
6.32 Scales and Coordinates . . . 801
6.32.1 Introduction . . . 801
6.32.2 Prerequisites . . . 801
6.32.3 Theory . . . 801
6.32.4 Assumptions . . . 802
6.32.5 R Implementation . . . 802
6.32.6 Output & Results . . . 803
6.32.7 Interpretation . . . 803
6.32.8 Practical Tips . . . 803
6.32.9 R Packages Used . . . 804
6.32.10 For Reviewers . . . 804
6.32.11 See also — labs in this chapter . . . 804
6.33 Scatter Plots . . . 804
6.33.1 Introduction . . . 804
6.33.2 Prerequisites . . . 804
6.33.3 Theory . . . 804
6.33.4 Assumptions . . . 805
6.33.5 R Implementation . . . 805
6.33.6 Output & Results . . . 806
6.33.7 Interpretation . . . 806
6.33.8 Practical Tips . . . 806
6.33.9 R Packages Used . . . 806
6.33.10 For Reviewers . . . 806
6.33.11 See also — labs in this chapter . . . 807

6.34 Stacked and Dodged Bars . . . 807
6.34.1 Introduction . . . 807
6.34.2 Prerequisites . . . 807
6.34.3 Theory . . . 807
6.34.4 Assumptions . . . 807
6.34.5 R Implementation . . . 807
6.34.6 Output & Results . . . 808
6.34.7 Interpretation . . . 808
6.34.8 Practical Tips . . . 808
6.34.9 R Packages Used . . . 809
6.34.10 For Reviewers . . . 809
6.34.11 See also — labs in this chapter . . . 809
6.35 Survival Curves . . . 809
6.35.1 Introduction . . . 809
6.35.2 Prerequisites . . . 810
6.35.3 Theory . . . 810
6.35.4 Assumptions . . . 810
6.35.5 R Implementation . . . 810
6.35.6 Output & Results . . . 811
6.35.7 Interpretation . . . 811
6.35.8 Practical Tips . . . 811
6.35.9 R Packages Used . . . 812
6.35.10 For Reviewers . . . 812
6.35.11 See also — labs in this chapter . . . 812
6.36 Time Series Plots . . . 812
6.36.1 Introduction . . . 812
6.36.2 Prerequisites . . . 812
6.36.3 Theory . . . 812
6.36.4 Assumptions . . . 813
6.36.5 R Implementation . . . 813
6.36.6 Output & Results . . . 813
6.36.7 Interpretation . . . 814
6.36.8 Practical Tips . . . 814
6.36.9 R Packages Used . . . 814
6.36.10 For Reviewers . . . 814
6.36.11 See also — labs in this chapter . . . 814
6.37 Violin Plots . . . 815
6.37.1 Introduction . . . 815
6.37.2 Prerequisites . . . 815
6.37.3 Theory . . . 815
6.37.4 Assumptions . . . 815
6.37.5 R Implementation . . . 815
6.37.6 Output & Results . . . 816
6.37.7 Interpretation . . . 816
6.37.8 Practical Tips . . . 816
6.37.9 R Packages Used . . . 817

6.37.10 For Reviewers . . . 817
6.37.11 See also — labs in this chapter . . . 817
6.38 Week 1, Session 5 — ggplot2 grammar and multi-panel layouts 817
6.38.1 Learning objectives . . . 817
6.38.2 Prerequisites . . . 817
6.38.3 Background . . . 817
6.38.4 Setup . . . 818
6.38.5 1. Goal . . . 818
6.38.6 2. Approach . . . 818
6.38.7 3. Execution . . . 818
6.38.8 4. Check . . . 819
6.38.9 5. Report . . . 819
6.38.10 Common pitfalls . . . 819
6.38.11 Further reading . . . 820
6.38.12 Session info . . . 820
6.38.13 See also — chapter index . . . 820

7 Regression & Modelling 821
7.1 Method pages . . . 821
7.2 Labs . . . 822
7.3 Added-Variable Plots . . . 823
7.3.1 Introduction . . . 823
7.3.2 Prerequisites . . . 823
7.3.3 Theory . . . 823
7.3.4 Assumptions . . . 824
7.3.5 R Implementation . . . 824
7.3.6 Output & Results . . . 824
7.3.7 Interpretation . . . 824
7.3.8 Practical Tips . . . 824
7.3.9 R Packages Used . . . 825
7.3.10 For Reviewers . . . 825
7.3.11 See also — labs in this chapter . . . 825
7.4 Best Subset Selection . . . 826
7.4.1 Introduction . . . 826
7.4.2 Prerequisites . . . 826
7.4.3 Theory . . . 826
7.4.4 Assumptions . . . 826
7.4.5 R Implementation . . . 826
7.4.6 Output & Results . . . 827
7.4.7 Interpretation . . . 827
7.4.8 Practical Tips . . . 827
7.4.9 R Packages Used . . . 828
7.4.10 For Reviewers . . . 828
7.4.11 See also — labs in this chapter . . . 828
7.5 Beta Regression . . . 828
7.5.1 Introduction . . . 828

7.5.2 Prerequisites . . . 829
7.5.3 Theory . . . 829
7.5.4 Assumptions . . . 829
7.5.5 R Implementation . . . 829
7.5.6 Output & Results . . . 830
7.5.7 Interpretation . . . 830
7.5.8 Practical Tips . . . 830
7.5.9 R Packages Used . . . 830
7.5.10 For Reviewers . . . 830
7.5.11 See also — labs in this chapter . . . 831
7.6 Centring and Scaling Predictors . . . 831
7.6.1 Introduction . . . 831
7.6.2 Prerequisites . . . 831
7.6.3 Theory . . . 831
7.6.4 Assumptions . . . 832
7.6.5 R Implementation . . . 832
7.6.6 Output & Results . . . 832
7.6.7 Interpretation . . . 832
7.6.8 Practical Tips . . . 833
7.6.9 R Packages Used . . . 833
7.6.10 For Reviewers . . . 833
7.6.11 See also — labs in this chapter . . . 833
7.7 Contrasts in R . . . 834
7.7.1 Introduction . . . 834
7.7.2 Prerequisites . . . 834
7.7.3 Theory . . . 834
7.7.4 Assumptions . . . 835
7.7.5 R Implementation . . . 835
7.7.6 Output & Results . . . 835
7.7.7 Interpretation . . . 835
7.7.8 Practical Tips . . . 836
7.7.9 R Packages Used . . . 836
7.7.10 For Reviewers . . . 836
7.7.11 See also — labs in this chapter . . . 836
7.8 Cook's Distance . . . 837
7.8.1 Introduction . . . 837
7.8.2 Prerequisites . . . 837
7.8.3 Theory . . . 837
7.8.4 Assumptions . . . 838
7.8.5 R Implementation . . . 838
7.8.6 Output & Results . . . 838
7.8.7 Interpretation . . . 838
7.8.8 Practical Tips . . . 838
7.8.9 R Packages Used . . . 839
7.8.10 For Reviewers . . . 839
7.8.11 See also — labs in this chapter . . . 839

7.9 Dirichlet Regression . . . 840
7.9.1 Introduction . . . 840
7.9.2 Prerequisites . . . 840
7.9.3 Theory . . . 840
7.9.4 Assumptions . . . 840
7.9.5 R Implementation . . . 840
7.9.6 Output & Results . . . 841
7.9.7 Interpretation . . . 841
7.9.8 Practical Tips . . . 841
7.9.9 R Packages Used . . . 842
7.9.10 For Reviewers . . . 842
7.9.11 See also — labs in this chapter . . . 842
7.10 Dummy Coding . . . 842
7.10.1 Introduction . . . 842
7.10.2 Prerequisites . . . 843
7.10.3 Theory . . . 843
7.10.4 Assumptions . . . 843
7.10.5 R Implementation . . . 843
7.10.6 Output & Results . . . 844
7.10.7 Interpretation . . . 844
7.10.8 Practical Tips . . . 844
7.10.9 R Packages Used . . . 844
7.10.10 For Reviewers . . . 845
7.10.11 See also — labs in this chapter . . . 845
7.11 Effect (Deviation) Coding . . . 845
7.11.1 Introduction . . . 845
7.11.2 Prerequisites . . . 845
7.11.3 Theory . . . 846
7.11.4 Assumptions . . . 846
7.11.5 R Implementation . . . 846
7.11.6 Output & Results . . . 846
7.11.7 Interpretation . . . 847
7.11.8 Practical Tips . . . 847
7.11.9 R Packages Used . . . 847
7.11.10 For Reviewers . . . 847
7.11.11 See also — labs in this chapter . . . 848
7.12 Elastic Net . . . 848
7.12.1 Introduction . . . 848
7.12.2 Prerequisites . . . 848
7.12.3 Theory . . . 848
7.12.4 Assumptions . . . 849
7.12.5 R Implementation . . . 849
7.12.6 Output & Results . . . 849
7.12.7 Interpretation . . . 850
7.12.8 Practical Tips . . . 850
7.12.9 R Packages Used . . . 850

7.12.10 For Reviewers . . . 850
7.12.11 See also — labs in this chapter . . . 850
7.13 GAMs: Introduction . . . 851
7.13.1 Introduction . . . 851
7.13.2 Prerequisites . . . 851
7.13.3 Theory . . . 851
7.13.4 Assumptions . . . 852
7.13.5 R Implementation . . . 852
7.13.6 Output & Results . . . 852
7.13.7 Interpretation . . . 852
7.13.8 Practical Tips . . . 852
7.13.9 R Packages Used . . . 853
7.13.10 For Reviewers . . . 853
7.13.11 See also — labs in this chapter . . . 853
7.14 GAMMs: Additive Mixed Models . . . 854
7.14.1 Introduction . . . 854
7.14.2 Prerequisites . . . 854
7.14.3 Theory . . . 854
7.14.4 Assumptions . . . 854
7.14.5 R Implementation . . . 854
7.14.6 Output & Results . . . 855
7.14.7 Interpretation . . . 855
7.14.8 Practical Tips . . . 855
7.14.9 R Packages Used . . . 856
7.14.10 For Reviewers . . . 856
7.14.11 See also — labs in this chapter . . . 856
7.15 Generalised Estimating Equations . . . 856
7.15.1 Introduction . . . 856
7.15.2 Prerequisites . . . 857
7.15.3 Theory . . . 857
7.15.4 Assumptions . . . 857
7.15.5 R Implementation . . . 857
7.15.6 Output & Results . . . 858
7.15.7 Interpretation . . . 858
7.15.8 Practical Tips . . . 858
7.15.9 R Packages Used . . . 858
7.15.10 For Reviewers . . . 858
7.15.11 See also — labs in this chapter . . . 859
7.16 glmer: Generalised Mixed Models . . . 859
7.16.1 Introduction . . . 859
7.16.2 Prerequisites . . . 859
7.16.3 Theory . . . 859
7.16.4 Assumptions . . . 860
7.16.5 R Implementation . . . 860
7.16.6 Output & Results . . . 860
7.16.7 Interpretation . . . 861

7.16.8 Practical Tips . . . 861
7.16.9 R Packages Used . . . 861
7.16.10 For Reviewers . . . 861
7.16.11 See also — labs in this chapter . . . 861
7.17 Hurdle Models . . . 862
7.17.1 Introduction . . . 862
7.17.2 Prerequisites . . . 862
7.17.3 Theory . . . 862
7.17.4 Assumptions . . . 863
7.17.5 R Implementation . . . 863
7.17.6 Output & Results . . . 863
7.17.7 Interpretation . . . 863
7.17.8 Practical Tips . . . 863
7.17.9 R Packages Used . . . 864
7.17.10 For Reviewers . . . 864
7.17.11 See also — labs in this chapter . . . 864
7.18 Interactions in Regression . . . 865
7.18.1 Introduction . . . 865
7.18.2 Prerequisites . . . 865
7.18.3 Theory . . . 865
7.18.4 Assumptions . . . 865
7.18.5 R Implementation . . . 866
7.18.6 Output & Results . . . 866
7.18.7 Interpretation . . . 866
7.18.8 Practical Tips . . . 866
7.18.9 R Packages Used . . . 867
7.18.10 For Reviewers . . . 867
7.18.11 See also — labs in this chapter . . . 867
7.19 Lasso Regression . . . 868
7.19.1 Introduction . . . 868
7.19.2 Prerequisites . . . 868
7.19.3 Theory . . . 868
7.19.4 Assumptions . . . 868
7.19.5 R Implementation . . . 868
7.19.6 Output & Results . . . 869
7.19.7 Interpretation . . . 869
7.19.8 Practical Tips . . . 869
7.19.9 R Packages Used . . . 869
7.19.10 For Reviewers . . . 870
7.19.11 See also — labs in this chapter . . . 870
7.20 Leverage and Influence . . . 870
7.20.1 Introduction . . . 870
7.20.2 Prerequisites . . . 871
7.20.3 Theory . . . 871
7.20.4 Assumptions . . . 871
7.20.5 R Implementation . . . 871

7.20.6 Output & Results . . . 872
7.20.7 Interpretation . . . 872
7.20.8 Practical Tips . . . 872
7.20.9 R Packages Used . . . 872
7.20.10 For Reviewers . . . 872
7.20.11 See also — labs in this chapter . . . 873
7.21 Worked lme4::lmer Examples . . . 873
7.21.1 Introduction . . . 873
7.21.2 Prerequisites . . . 873
7.21.3 Theory . . . 873
7.21.4 Assumptions . . . 874
7.21.5 R Implementation . . . 874
7.21.6 Output & Results . . . 875
7.21.7 Interpretation . . . 875
7.21.8 Practical Tips . . . 875
7.21.9 R Packages Used . . . 876
7.21.10 For Reviewers . . . 876
7.21.11 See also — labs in this chapter . . . 876
7.22 Logistic Regression Diagnostics . . . 876
7.22.1 Introduction . . . 876
7.22.2 Prerequisites . . . 877
7.22.3 Theory . . . 877
7.22.4 Assumptions . . . 877
7.22.5 R Implementation . . . 877
7.22.6 Output & Results . . . 878
7.22.7 Interpretation . . . 878
7.22.8 Practical Tips . . . 878
7.22.9 R Packages Used . . . 878
7.22.10 For Reviewers . . . 878
7.22.11 See also — labs in this chapter . . . 879
7.23 Logistic Regression . . . 879
7.23.1 Introduction . . . 879
7.23.2 Prerequisites . . . 879
7.23.3 Theory . . . 879
7.23.4 Assumptions . . . 880
7.23.5 R Implementation . . . 880
7.23.6 Output & Results . . . 880
7.23.7 Interpretation . . . 880
7.23.8 Practical Tips . . . 881
7.23.9 R Packages Used . . . 881
7.23.10 For Reviewers . . . 881
7.23.11 See also — labs in this chapter . . . 881
7.24 Mixed-Effects Models: Introduction . . . 882
7.24.1 Introduction . . . 882
7.24.2 Prerequisites . . . 882
7.24.3 Theory . . . 882

7.24.4 Assumptions . . . 882
7.24.5 R Implementation . . . 883
7.24.6 Output & Results . . . 883
7.24.7 Interpretation . . . 883
7.24.8 Practical Tips . . . 883
7.24.9 For Reviewers . . . 883
7.24.10 See also — labs in this chapter . . . 884
7.25 Model Selection with AIC and BIC . . . 884
7.25.1 Introduction . . . 884
7.25.2 Prerequisites . . . 884
7.25.3 Theory . . . 884
7.25.4 Assumptions . . . 885
7.25.5 R Implementation . . . 885
7.25.6 Output & Results . . . 885
7.25.7 Interpretation . . . 885
7.25.8 Practical Tips . . . 886
7.25.9 R Packages Used . . . 886
7.25.10 For Reviewers . . . 886
7.25.11 See also — labs in this chapter . . . 886
7.26 Multicollinearity and VIF . . . 887
7.26.1 Introduction . . . 887
7.26.2 Prerequisites . . . 887
7.26.3 Theory . . . 887
7.26.4 Assumptions . . . 888
7.26.5 R Implementation . . . 888
7.26.6 Output & Results . . . 888
7.26.7 Interpretation . . . 888
7.26.8 Practical Tips . . . 888
7.26.9 R Packages Used . . . 889
7.26.10 For Reviewers . . . 889
7.26.11 See also — labs in this chapter . . . 889
7.27 Multinomial Logistic Regression . . . 890
7.27.1 Introduction . . . 890
7.27.2 Prerequisites . . . 890
7.27.3 Theory . . . 890
7.27.4 Assumptions . . . 890
7.27.5 R Implementation . . . 890
7.27.6 Output & Results . . . 891
7.27.7 Interpretation . . . 891
7.27.8 Practical Tips . . . 891
7.27.9 R Packages Used . . . 892
7.27.10 For Reviewers . . . 892
7.27.11 See also — labs in this chapter . . . 892
7.28 Multiple Linear Regression . . . 892
7.28.1 Introduction . . . 892
7.28.2 Prerequisites . . . 893

7.28.3 Theory . . . 893
7.28.4 Assumptions . . . 893
7.28.5 R Implementation . . . 893
7.28.6 Output & Results . . . 893
7.28.7 Interpretation . . . 893
7.28.8 Practical Tips . . . 894
7.28.9 For Reviewers . . . 894
7.28.10 See also — labs in this chapter . . . 894
7.29 Negative Binomial Regression . . . 894
7.29.1 Introduction . . . 894
7.29.2 Prerequisites . . . 895
7.29.3 Theory . . . 895
7.29.4 Assumptions . . . 895
7.29.5 R Implementation . . . 895
7.29.6 Output & Results . . . 896
7.29.7 Interpretation . . . 896
7.29.8 Practical Tips . . . 896
7.29.9 R Packages Used . . . 896
7.29.10 For Reviewers . . . 896
7.29.11 See also — labs in this chapter . . . 897
7.30 Nested vs Crossed Random Effects . . . 897
7.30.1 Introduction . . . 897
7.30.2 Prerequisites . . . 897
7.30.3 Theory . . . 897
7.30.4 Assumptions . . . 898
7.30.5 R Implementation . . . 898
7.30.6 Output & Results . . . 898
7.30.7 Interpretation . . . 899
7.30.8 Practical Tips . . . 899
7.30.9 R Packages Used . . . 899
7.30.10 For Reviewers . . . 899
7.30.11 See also — labs in this chapter . . . 899
7.31 Offsets in Regression . . . 900
7.31.1 Introduction . . . 900
7.31.2 Prerequisites . . . 900
7.31.3 Theory . . . 900
7.31.4 Assumptions . . . 901
7.31.5 R Implementation . . . 901
7.31.6 Output & Results . . . 901
7.31.7 Interpretation . . . 901
7.31.8 Practical Tips . . . 902
7.31.9 R Packages Used . . . 902
7.31.10 For Reviewers . . . 902
7.31.11 See also — labs in this chapter . . . 902
7.32 Ordinal Logistic Regression . . . 903
7.32.1 Introduction . . . 903

7.32.2 Prerequisites . . . 903
7.32.3 Theory . . . 903
7.32.4 Assumptions . . . 903
7.32.5 R Implementation . . . 904
7.32.6 Output & Results . . . 904
7.32.7 Interpretation . . . 904
7.32.8 Practical Tips . . . 904
7.32.9 R Packages Used . . . 905
7.32.10 For Reviewers . . . 905
7.32.11 See also — labs in this chapter . . . 905
7.33 Partial Correlation . . . 906
7.33.1 Introduction . . . 906
7.33.2 Prerequisites . . . 906
7.33.3 Theory . . . 906
7.33.4 Assumptions . . . 906
7.33.5 R Implementation . . . 907
7.33.6 Output & Results . . . 907
7.33.7 Interpretation . . . 907
7.33.8 Practical Tips . . . 907
7.33.9 R Packages Used . . . 908
7.33.10 For Reviewers . . . 908
7.33.11 See also — labs in this chapter . . . 908
7.34 Poisson Regression . . . 909
7.34.1 Introduction . . . 909
7.34.2 Prerequisites . . . 909
7.34.3 Theory . . . 909
7.34.4 Assumptions . . . 909
7.34.5 R Implementation . . . 909
7.34.6 Output & Results . . . 910
7.34.7 Interpretation . . . 910
7.34.8 Practical Tips . . . 910
7.34.9 R Packages Used . . . 911
7.34.10 For Reviewers . . . 911
7.34.11 See also — labs in this chapter . . . 911
7.35 Polynomial Regression . . . 912
7.35.1 Introduction . . . 912
7.35.2 Prerequisites . . . 912
7.35.3 Theory . . . 912
7.35.4 Assumptions . . . 912
7.35.5 R Implementation . . . 912
7.35.6 Output & Results . . . 913
7.35.7 Interpretation . . . 913
7.35.8 Practical Tips . . . 913
7.35.9 R Packages Used . . . 914
7.35.10 For Reviewers . . . 914
7.35.11 See also — labs in this chapter . . . 914

7.36 Probit Regression . . . 915
7.36.1 Introduction . . . 915
7.36.2 Prerequisites . . . 915
7.36.3 Theory . . . 915
7.36.4 Assumptions . . . 915
7.36.5 R Implementation . . . 915
7.36.6 Output & Results . . . 916
7.36.7 Interpretation . . . 916
7.36.8 Practical Tips . . . 916
7.36.9 R Packages Used . . . 917
7.36.10 For Reviewers . . . 917
7.36.11 See also — labs in this chapter . . . 917
7.37 Quantile Regression . . . 917
7.37.1 Introduction . . . 917
7.37.2 Prerequisites . . . 918
7.37.3 Theory . . . 918
7.37.4 Assumptions . . . 918
7.37.5 R Implementation . . . 918
7.37.6 Output & Results . . . 919
7.37.7 Interpretation . . . 919
7.37.8 Practical Tips . . . 919
7.37.9 R Packages Used . . . 919
7.37.10 For Reviewers . . . 919
7.37.11 See also — labs in this chapter . . . 920
7.38 Quasi-Poisson Regression . . . 920
7.38.1 Introduction . . . 920
7.38.2 Prerequisites . . . 920
7.38.3 Theory . . . 920
7.38.4 Assumptions . . . 921
7.38.5 R Implementation . . . 921
7.38.6 Output & Results . . . 921
7.38.7 Interpretation . . . 921
7.38.8 Practical Tips . . . 921
7.38.9 R Packages Used . . . 922
7.38.10 For Reviewers . . . 922
7.38.11 See also — labs in this chapter . . . 922
7.39 R-Squared and Adjusted R-Squared . . . 923
7.39.1 Introduction . . . 923
7.39.2 Prerequisites . . . 923
7.39.3 Theory . . . 923
7.39.4 Assumptions . . . 923
7.39.5 R Implementation . . . 924
7.39.6 Output & Results . . . 924
7.39.7 Interpretation . . . 924
7.39.8 Practical Tips . . . 924
7.39.9 R Packages Used . . . 925

7.39.10 For Reviewers . . . 925
7.39.11 See also — labs in this chapter . . . 925
7.40 Random-Intercept Models . . . 925
7.40.1 Introduction . . . 925
7.40.2 Prerequisites . . . 926
7.40.3 Theory . . . 926
7.40.4 Assumptions . . . 926
7.40.5 R Implementation . . . 926
7.40.6 Output & Results . . . 926
7.40.7 Interpretation . . . 927
7.40.8 Practical Tips . . . 927
7.40.9 R Packages Used . . . 927
7.40.10 For Reviewers . . . 927
7.40.11 See also — labs in this chapter . . . 928
7.41 Random-Slope Models . . . 928
7.41.1 Introduction . . . 928
7.41.2 Prerequisites . . . 928
7.41.3 Theory . . . 928
7.41.4 Assumptions . . . 929
7.41.5 R Implementation . . . 929
7.41.6 Output & Results . . . 929
7.41.7 Interpretation . . . 929
7.41.8 Practical Tips . . . 930
7.41.9 R Packages Used . . . 930
7.41.10 For Reviewers . . . 930
7.41.11 See also — labs in this chapter . . . 930
7.42 Linear Regression Assumptions . . . 931
7.42.1 Introduction . . . 931
7.42.2 Prerequisites . . . 931
7.42.3 Theory . . . 931
7.42.4 Assumptions . . . 931
7.42.5 R Implementation . . . 932
7.42.6 Output & Results . . . 932
7.42.7 Interpretation . . . 932
7.42.8 Practical Tips . . . 933
7.42.9 R Packages Used . . . 933
7.42.10 For Reviewers . . . 933
7.42.11 See also — labs in this chapter . . . 933
7.43 Residual Diagnostics . . . 934
7.43.1 Introduction . . . 934
7.43.2 Prerequisites . . . 934
7.43.3 Theory . . . 934
7.43.4 Assumptions . . . 935
7.43.5 R Implementation . . . 935
7.43.6 Output & Results . . . 935
7.43.7 Interpretation . . . 935

7.43.8 Practical Tips . . . 935
7.43.9 R Packages Used . . . 936
7.43.10 For Reviewers . . . 936
7.43.11 See also — labs in this chapter . . . 936
7.44 Ridge Regression . . . 937
7.44.1 Introduction . . . 937
7.44.2 Prerequisites . . . 937
7.44.3 Theory . . . 937
7.44.4 Assumptions . . . 937
7.44.5 R Implementation . . . 937
7.44.6 Output & Results . . . 938
7.44.7 Interpretation . . . 938
7.44.8 Practical Tips . . . 938
7.44.9 R Packages Used . . . 938
7.44.10 For Reviewers . . . 939
7.44.11 See also — labs in this chapter . . . 939
7.45 Robust Regression . . . 939
7.45.1 Introduction . . . 939
7.45.2 Prerequisites . . . 940
7.45.3 Theory . . . 940
7.45.4 Assumptions . . . 940
7.45.5 R Implementation . . . 940
7.45.6 Output & Results . . . 940
7.45.7 Interpretation . . . 941
7.45.8 Practical Tips . . . 941
7.45.9 R Packages Used . . . 941
7.45.10 For Reviewers . . . 941
7.45.11 See also — labs in this chapter . . . 941
7.46 ROC and AUC for Logistic Models . . . 942
7.46.1 Introduction . . . 942
7.46.2 Prerequisites . . . 942
7.46.3 Theory . . . 942
7.46.4 Assumptions . . . 943
7.46.5 R Implementation . . . 943
7.46.6 Output & Results . . . 943
7.46.7 Interpretation . . . 943
7.46.8 Practical Tips . . . 943
7.46.9 R Packages Used . . . 944
7.46.10 For Reviewers . . . 944
7.46.11 See also — labs in this chapter . . . 944
7.47 Simple Linear Regression . . . 945
7.47.1 Introduction . . . 945
7.47.2 Prerequisites . . . 945
7.47.3 Theory . . . 945
7.47.4 Assumptions . . . 945
7.47.5 R Implementation . . . 946

7.47.6 Output & Results . . . 946
7.47.7 Interpretation . . . 947
7.47.8 Practical Tips . . . 947
7.47.9 For Reviewers . . . 947
7.47.10 See also — labs in this chapter . . . 947
7.48 Spline Regression . . . 948
7.48.1 Introduction . . . 948
7.48.2 Prerequisites . . . 948
7.48.3 Theory . . . 948
7.48.4 Assumptions . . . 948
7.48.5 R Implementation . . . 949
7.48.6 Output & Results . . . 949
7.48.7 Interpretation . . . 949
7.48.8 Practical Tips . . . 949
7.48.9 R Packages Used . . . 950
7.48.10 For Reviewers . . . 950
7.48.11 See also — labs in this chapter . . . 950
7.49 Stepwise Regression: Pitfalls . . . 951
7.49.1 Introduction . . . 951
7.49.2 Prerequisites . . . 951
7.49.3 Theory . . . 951
7.49.4 Assumptions . . . 951
7.49.5 R Implementation . . . 951
7.49.6 Output & Results . . . 952
7.49.7 Interpretation . . . 952
7.49.8 Practical Tips . . . 952
7.49.9 R Packages Used . . . 953
7.49.10 For Reviewers . . . 953
7.49.11 See also — labs in this chapter . . . 953
7.50 Cox Regression as Regression . . . 953
7.50.1 Introduction . . . 953
7.50.2 Prerequisites . . . 954
7.50.3 Theory . . . 954
7.50.4 Assumptions . . . 954
7.50.5 R Implementation . . . 954
7.50.6 Output & Results . . . 954
7.50.7 Interpretation . . . 955
7.50.8 Practical Tips . . . 955
7.50.9 R Packages Used . . . 955
7.50.10 For Reviewers . . . 955
7.50.11 See also — labs in this chapter . . . 955
7.51 Tobit Regression . . . 956
7.51.1 Introduction . . . 956
7.51.2 Prerequisites . . . 956
7.51.3 Theory . . . 956
7.51.4 Assumptions . . . 957

7.51.5 R Implementation . . . 957
7.51.6 Output & Results . . . 957
7.51.7 Interpretation . . . 957
7.51.8 Practical Tips . . . 957
7.51.9 R Packages Used . . . 958
7.51.10 For Reviewers . . . 958
7.51.11 See also — labs in this chapter . . . 958
7.52 Truncated Regression . . . 959
7.52.1 Introduction . . . 959
7.52.2 Prerequisites . . . 959
7.52.3 Theory . . . 959
7.52.4 Assumptions . . . 959
7.52.5 R Implementation . . . 959
7.52.6 Output & Results . . . 960
7.52.7 Interpretation . . . 960
7.52.8 Practical Tips . . . 960
7.52.9 R Packages Used . . . 961
7.52.10 For Reviewers . . . 961
7.52.11 See also — labs in this chapter . . . 961
7.53 Zero-Inflated Models . . . 961
7.53.1 Introduction . . . 961
7.53.2 Prerequisites . . . 962
7.53.3 Theory . . . 962
7.53.4 Assumptions . . . 962
7.53.5 R Implementation . . . 962
7.53.6 Output & Results . . . 963
7.53.7 Interpretation . . . 963
7.53.8 Practical Tips . . . 963
7.53.9 R Packages Used . . . 963
7.53.10 For Reviewers . . . 964
7.53.11 See also — labs in this chapter . . . 964
7.54 Week 3, Session 2 — ANCOVA in RCTs (baseline adjustment) 964
7.54.1 Learning objectives . . . 964
7.54.2 Prerequisites . . . 965
7.54.3 Background . . . 965
7.54.4 Setup . . . 965
7.54.5 1. Hypothesis . . . 965
7.54.6 2. Visualise . . . 965
7.54.7 3. Assumptions . . . 966
7.54.8 4. Conduct . . . 966
7.54.9 5. Concluding statement . . . 966
7.54.10 Common pitfalls . . . 967
7.54.11 Further reading . . . 967
7.54.12 Session info . . . 967
7.54.13 See also — chapter index . . . 967

7.55 Week 3, Session 5 — Calibration, discrimination, ROC/AUC, Brier score . . . 967
7.55.1 Learning objectives . . . 967
7.55.2 Prerequisites . . . 967
7.55.3 Background . . . 967
7.55.4 Setup . . . 968
7.55.5 1. Hypothesis . . . 968
7.55.6 2. Visualise . . . 968
7.55.7 3. Assumptions . . . 969
7.55.8 4. Conduct . . . 969
7.55.9 5. Concluding statement . . . 969
7.55.10 Common pitfalls . . . 969
7.55.11 Further reading . . . 970
7.55.12 Session info . . . 970
7.55.13 See also — chapter index . . . 970
7.56 Week 1, Session 1 — Correlation vs regression; Model-I/II regression . . . 970
7.56.1 Learning objectives . . . 970
7.56.2 Prerequisites . . . 970
7.56.3 Background . . . 970
7.56.4 Setup . . . 971
7.56.5 1. Hypothesis . . . 971
7.56.6 2. Visualise . . . 971
7.56.7 3. Assumptions . . . 972
7.56.8 4. Conduct . . . 972
7.56.9 5. Concluding statement . . . 973
7.56.10 Common pitfalls . . . 973
7.56.11 Further reading . . . 973
7.56.12 Session info . . . 973
7.56.13 See also — chapter index . . . 973
7.57 Week 4, Session 4 — Decision curves, NRI, IDI . . . 973
7.57.1 Learning objectives . . . 973
7.57.2 Prerequisites . . . 974
7.57.3 Background . . . 974
7.57.4 Setup . . . 974
7.57.5 1. Goal . . . 974
7.57.6 2. Approach . . . 974
7.57.7 3. Execution — decision curve (hand-coded) . . . 975
7.57.8 4. Check — NRI and IDI . . . 975
7.57.9 5. Report . . . 976
7.57.10 Common pitfalls . . . 976
7.57.11 Further reading . . . 976
7.57.12 Session info . . . 976
7.57.13 See also — chapter index . . . 976
7.58 Week 1, Session 4 — Diagnostics: residuals, QQ, leverage, Cook's distance, VIF . . . 976

7.58.1 Learning objectives . . . 976
7.58.2 Prerequisites . . . 977
7.58.3 Background . . . 977
7.58.4 Setup . . . 977
7.58.5 1. Hypothesis . . . 977
7.58.6 2. Visualise . . . 977
7.58.7 3. Assumptions . . . 978
7.58.8 4. Conduct . . . 978
7.58.9 5. Concluding statement . . . 978
7.58.10 Common pitfalls . . . 979
7.58.11 Further reading . . . 979
7.58.12 Session info . . . 979
7.58.13 See also — chapter index . . . 979
7.59 Week 4, Session 1 — Dichotomisation, change scores, regression to the mean . . . 979
7.59.1 Learning objectives . . . 979
7.59.2 Prerequisites . . . 979
7.59.3 Background . . . 980
7.59.4 Setup . . . 980
7.59.5 1. Hypothesis . . . 980
7.59.6 2. Visualise . . . 980
7.59.7 3. Assumptions . . . 981
7.59.8 4. Conduct . . . 981
7.59.9 5. Concluding statement . . . 981
7.59.10 Common pitfalls . . . 982
7.59.11 Further reading . . . 982
7.59.12 Session info . . . 982
7.59.13 See also — chapter index . . . 982
7.60 Week 2, Session 4 — GAMs with mgcv . . . 982
7.60.1 Learning objectives . . . 982
7.60.2 Prerequisites . . . 982
7.60.3 Background . . . 983
7.60.4 Setup . . . 983
7.60.5 1. Hypothesis . . . 983
7.60.6 2. Visualise . . . 983
7.60.7 3. Assumptions . . . 984
7.60.8 4. Conduct . . . 984
7.60.9 5. Concluding statement . . . 984
7.60.10 Common pitfalls . . . 984
7.60.11 Further reading . . . 984
7.60.12 Session info . . . 984
7.60.13 See also — chapter index . . . 985
7.61 Week 3, Session 1 — Logistic regression (binomial GLM) . . . 985
7.61.1 Learning objectives . . . 985
7.61.2 Prerequisites . . . 985
7.61.3 Background . . . 985

7.61.4 Setup . . . 985
7.61.5 1. Hypothesis . . . 986
7.61.6 2. Visualise . . . 986
7.61.7 3. Assumptions . . . 986
7.61.8 4. Conduct . . . 986
7.61.9 5. Concluding statement . . . 987
7.61.10 Common pitfalls . . . 987
7.61.11 Further reading . . . 987
7.61.12 Session info . . . 987
7.61.13 See also — chapter index . . . 987
7.62 Week 1, Session 3 — Multiple regression: confounding, interaction, centring . . . 987
7.62.1 Learning objectives . . . 987
7.62.2 Prerequisites . . . 988
7.62.3 Background . . . 988
7.62.4 Setup . . . 988
7.62.5 1. Hypothesis . . . 988
7.62.6 2. Visualise . . . 988
7.62.7 3. Assumptions . . . 989
7.62.8 4. Conduct . . . 989
7.62.9 5. Concluding statement . . . 990
7.62.10 Common pitfalls . . . 990
7.62.11 Further reading . . . 990
7.62.12 Session info . . . 990
7.62.13 See also — chapter index . . . 990
7.63 Week 2, Session 5 — Non-linear regression with nls . . . 990
7.63.1 Learning objectives . . . 990
7.63.2 Prerequisites . . . 991
7.63.3 Background . . . 991
7.63.4 Setup . . . 991
7.63.5 1. Hypothesis . . . 991
7.63.6 2. Visualise . . . 991
7.63.7 3. Assumptions . . . 992
7.63.8 4. Conduct . . . 992
7.63.9 5. Concluding statement . . . 992
7.63.10 Common pitfalls . . . 992
7.63.11 Further reading . . . 992
7.63.12 Session info . . . 993
7.63.13 See also — chapter index . . . 993
7.64 Week 2, Session 1 — One-way ANOVA and contrasts (emmeans) . . . 993
7.64.1 Learning objectives . . . 993
7.64.2 Prerequisites . . . 993
7.64.3 Background . . . 993
7.64.4 Setup . . . 994
7.64.5 1. Hypothesis . . . 994

7.64.6 2. Visualise . . . 994
7.64.7 3. Assumptions . . . 994
7.64.8 4. Conduct . . . 994
7.64.9 5. Concluding statement . . . 995
7.64.10 Common pitfalls . . . 995
7.64.11 Further reading . . . 995
7.64.12 Session info . . . 995
7.64.13 See also — chapter index . . . 995
7.65 Week 3, Session 3 — Ordinal and multinomial regression . . . 995
7.65.1 Learning objectives . . . 995
7.65.2 Prerequisites . . . 996
7.65.3 Background . . . 996
7.65.4 Setup . . . 996
7.65.5 1. Hypothesis . . . 996
7.65.6 2. Visualise . . . 996
7.65.7 3. Assumptions . . . 997
7.65.8 4. Conduct . . . 997
7.65.9 5. Concluding statement . . . 998
7.65.10 Common pitfalls . . . 998
7.65.11 Further reading . . . 998
7.65.12 Session info . . . 998
7.65.13 See also — chapter index . . . 998
7.66 Week 3, Session 4 — Poisson and negative-binomial regression 998
7.66.1 Learning objectives . . . 998
7.66.2 Prerequisites . . . 999
7.66.3 Background . . . 999
7.66.4 Setup . . . 999
7.66.5 1. Hypothesis . . . 999
7.66.6 2. Visualise . . . 999
7.66.7 3. Assumptions . . . 1000
7.66.8 4. Conduct . . . 1000
7.66.9 5. Concluding statement . . . 1000
7.66.10 Common pitfalls . . . 1000
7.66.11 Further reading . . . 1000
7.66.12 Session info . . . 1001
7.66.13 See also — chapter index . . . 1001
7.67 Week 2, Session 3 — RCBD and repeated measures → mixed models . . . 1001
7.67.1 Learning objectives . . . 1001
7.67.2 Prerequisites . . . 1001
7.67.3 Background . . . 1001
7.67.4 Setup . . . 1002
7.67.5 1. Hypothesis . . . 1002
7.67.6 2. Visualise . . . 1002
7.67.7 3. Assumptions . . . 1002
7.67.8 4. Conduct . . . 1003

7.67.9 5. Concluding statement . . . 1003
7.67.10 Common pitfalls . . . 1003
7.67.11 Further reading . . . 1003
7.67.12 Session info . . . 1003
7.67.13 See also — chapter index . . . 1003
7.68 Week 1, Session 5 — Robust regression, weighted LS, HC (sandwich) standard errors . . . 1004
7.68.1 Learning objectives . . . 1004
7.68.2 Prerequisites . . . 1004
7.68.3 Background . . . 1004
7.68.4 Setup . . . 1004
7.68.5 1. Hypothesis . . . 1005
7.68.6 2. Visualise . . . 1005
7.68.7 3. Assumptions . . . 1005
7.68.8 4. Conduct . . . 1005
7.68.9 5. Concluding statement . . . 1006
7.68.10 Common pitfalls . . . 1006
7.68.11 Further reading . . . 1006
7.68.12 Session info . . . 1006
7.68.13 See also — chapter index . . . 1006
7.69 Week 1, Session 2 — Simple linear regression . . . 1007
7.69.1 Learning objectives . . . 1007
7.69.2 Prerequisites . . . 1007
7.69.3 Background . . . 1007
7.69.4 Setup . . . 1007
7.69.5 1. Hypothesis . . . 1008
7.69.6 2. Visualise . . . 1008
7.69.7 3. Assumptions . . . 1008
7.69.8 4. Conduct . . . 1008
7.69.9 5. Concluding statement . . . 1008
7.69.10 Common pitfalls . . . 1009
7.69.11 Further reading . . . 1009
7.69.12 Session info . . . 1009
7.69.13 See also — chapter index . . . 1009
7.70 Week 2, Session 2 — Two-way / factorial ANOVA with interaction . . . 1009
7.70.1 Learning objectives . . . 1009
7.70.2 Prerequisites . . . 1009
7.70.3 Background . . . 1009
7.70.4 Setup . . . 1010
7.70.5 1. Hypothesis . . . 1010
7.70.6 2. Visualise . . . 1010
7.70.7 3. Assumptions . . . 1011
7.70.8 4. Conduct . . . 1011
7.70.9 5. Concluding statement . . . 1011
7.70.10 Common pitfalls . . . 1011

7.70.11 Further reading . . . 1011
7.70.12 Session info . . . 1011
7.70.13 See also — chapter index . . . 1011

8 Multivariate Methods 1013
8.1 Method pages . . . 1013
8.2 Labs . . . 1014
8.3 Bray-Curtis Dissimilarity . . . 1014
8.3.1 Introduction . . . 1014
8.3.2 Prerequisites . . . 1014
8.3.3 Theory . . . 1015
8.3.4 Assumptions . . . 1015
8.3.5 R Implementation . . . 1015
8.3.6 Output & Results . . . 1015
8.3.7 Interpretation . . . 1015
8.3.8 Practical Tips . . . 1016
8.3.9 R Packages Used . . . 1016
8.3.10 For Reviewers . . . 1016
8.3.11 See also — labs in this chapter . . . 1016
8.4 Canonical Correlation Analysis . . . 1017
8.4.1 Introduction . . . 1017
8.4.2 Prerequisites . . . 1017
8.4.3 Theory . . . 1017
8.4.4 Assumptions . . . 1017
8.4.5 R Implementation . . . 1017
8.4.6 Output & Results . . . 1018
8.4.7 Interpretation . . . 1018
8.4.8 Practical Tips . . . 1018
8.4.9 R Packages Used . . . 1019
8.4.10 For Reviewers . . . 1019
8.4.11 See also — labs in this chapter . . . 1019
8.5 Cluster Validation: Silhouette . . . 1019
8.5.1 Introduction . . . 1019
8.5.2 Prerequisites . . . 1019
8.5.3 Theory . . . 1019
8.5.4 Assumptions . . . 1020
8.5.5 R Implementation . . . 1020
8.5.6 Output & Results . . . 1020
8.5.7 Interpretation . . . 1021
8.5.8 Practical Tips . . . 1021
8.5.9 R Packages Used . . . 1021
8.5.10 For Reviewers . . . 1021
8.5.11 See also — labs in this chapter . . . 1022
8.6 Correspondence Analysis . . . 1022
8.6.1 Introduction . . . 1022
8.6.2 Prerequisites . . . 1022

8.6.3 Theory . . . 1022
8.6.4 Assumptions . . . 1022
8.6.5 R Implementation . . . 1023
8.6.6 Output & Results . . . 1023
8.6.7 Interpretation . . . 1023
8.6.8 Practical Tips . . . 1023
8.6.9 R Packages Used . . . 1024
8.6.10 For Reviewers . . . 1024
8.6.11 See also — labs in this chapter . . . 1024
8.7 DBSCAN Clustering . . . 1024
8.7.1 Introduction . . . 1024
8.7.2 Prerequisites . . . 1024
8.7.3 Theory . . . 1025
8.7.4 Assumptions . . . 1025
8.7.5 R Implementation . . . 1025
8.7.6 Output & Results . . . 1025
8.7.7 Interpretation . . . 1026
8.7.8 Practical Tips . . . 1026
8.7.9 R Packages Used . . . 1026
8.7.10 For Reviewers . . . 1026
8.7.11 See also — labs in this chapter . . . 1026
8.8 Distance and Dissimilarity Measures . . . 1027
8.8.1 Introduction . . . 1027
8.8.2 Prerequisites . . . 1027
8.8.3 Theory . . . 1027
8.8.4 Assumptions . . . 1027
8.8.5 R Implementation . . . 1028
8.8.6 Output & Results . . . 1028
8.8.7 Interpretation . . . 1028
8.8.8 Practical Tips . . . 1028
8.8.9 R Packages Used . . . 1029
8.8.10 For Reviewers . . . 1029
8.8.11 See also — labs in this chapter . . . 1029
8.9 The Elbow Method . . . 1029
8.9.1 Introduction . . . 1029
8.9.2 Prerequisites . . . 1029
8.9.3 Theory . . . 1030
8.9.4 Assumptions . . . 1030
8.9.5 R Implementation . . . 1030
8.9.6 Output & Results . . . 1030
8.9.7 Interpretation . . . 1030
8.9.8 Practical Tips . . . 1031
8.9.9 R Packages Used . . . 1031
8.9.10 For Reviewers . . . 1031
8.9.11 See also — labs in this chapter . . . 1031
8.10 Confirmatory Factor Analysis . . . 1031

8.10.1 Introduction . . . 1031
8.10.2 Prerequisites . . . 1032
8.10.3 Theory . . . 1032
8.10.4 Assumptions . . . 1032
8.10.5 R Implementation . . . 1032
8.10.6 Output & Results . . . 1033
8.10.7 Interpretation . . . 1033
8.10.8 Practical Tips . . . 1033
8.10.9 R Packages Used . . . 1034
8.10.10 For Reviewers . . . 1034
8.10.11 See also — labs in this chapter . . . 1034
8.11 Exploratory Factor Analysis . . . 1034
8.11.1 Introduction . . . 1034
8.11.2 Prerequisites . . . 1034
8.11.3 Theory . . . 1034
8.11.4 Assumptions . . . 1035
8.11.5 R Implementation . . . 1035
8.11.6 Output & Results . . . 1036
8.11.7 Interpretation . . . 1036
8.11.8 Practical Tips . . . 1036
8.11.9 R Packages Used . . . 1036
8.11.10 For Reviewers . . . 1036
8.11.11 See also — labs in this chapter . . . 1037
8.12 Factor Rotation . . . 1037
8.12.1 Introduction . . . 1037
8.12.2 Prerequisites . . . 1037
8.12.3 Theory . . . 1037
8.12.4 Assumptions . . . 1038
8.12.5 R Implementation . . . 1038
8.12.6 Output & Results . . . 1038
8.12.7 Interpretation . . . 1038
8.12.8 Practical Tips . . . 1039
8.12.9 R Packages Used . . . 1039
8.12.10 For Reviewers . . . 1039
8.12.11 See also — labs in this chapter . . . 1039
8.13 The Gap Statistic . . . 1039
8.13.1 Introduction . . . 1039
8.13.2 Prerequisites . . . 1040
8.13.3 Theory . . . 1040
8.13.4 Assumptions . . . 1040
8.13.5 R Implementation . . . 1040
8.13.6 Output & Results . . . 1041
8.13.7 Interpretation . . . 1041
8.13.8 Practical Tips . . . 1041
8.13.9 R Packages Used . . . 1041
8.13.10 For Reviewers . . . 1041

8.13.11 See also — labs in this chapter 1042
8.14 Gaussian Mixture Models 1042
8.14.1 Introduction 1042
8.14.2 Prerequisites 1042
8.14.3 Theory 1042
8.14.4 Assumptions 1042
8.14.5 R Implementation 1043
8.14.6 Output & Results 1043
8.14.7 Interpretation 1043
8.14.8 Practical Tips 1043
8.14.9 R Packages Used 1044
8.14.10 For Reviewers 1044
8.14.11 See also — labs in this chapter 1044
8.15 Hierarchical Clustering 1044
8.15.1 Introduction 1044
8.15.2 Prerequisites 1045
8.15.3 Theory 1045
8.15.4 Assumptions 1045
8.15.5 R Implementation 1045
8.15.6 Output & Results 1046
8.15.7 Interpretation 1046
8.15.8 Practical Tips 1046
8.15.9 R Packages Used 1046
8.15.10 For Reviewers 1046
8.15.11 See also — labs in this chapter 1047
8.16 Independent Component Analysis 1047
8.16.1 Introduction 1047
8.16.2 Prerequisites 1047
8.16.3 Theory 1047
8.16.4 Assumptions 1048
8.16.5 R Implementation 1048
8.16.6 Output & Results 1048
8.16.7 Interpretation 1048
8.16.8 Practical Tips 1049
8.16.9 R Packages Used 1049
8.16.10 For Reviewers 1049
8.16.11 See also — labs in this chapter 1049
8.17 Jaccard and Dice Dissimilarity 1050
8.17.1 Introduction 1050
8.17.2 Prerequisites 1050
8.17.3 Theory 1050
8.17.4 Assumptions 1050
8.17.5 R Implementation 1050
8.17.6 Output & Results 1051
8.17.7 Interpretation 1051
8.17.8 Practical Tips 1051

8.17.9 R Packages Used . . . 1052
8.17.10 For Reviewers . . . 1052
8.17.11 See also — labs in this chapter . . . 1052
8.18 Kernel PCA . . . 1052
8.18.1 Introduction . . . 1052
8.18.2 Prerequisites . . . 1052
8.18.3 Theory . . . 1052
8.18.4 Assumptions . . . 1053
8.18.5 R Implementation . . . 1053
8.18.6 Output & Results . . . 1053
8.18.7 Interpretation . . . 1053
8.18.8 Practical Tips . . . 1054
8.18.9 R Packages Used . . . 1054
8.18.10 For Reviewers . . . 1054
8.18.11 See also — labs in this chapter . . . 1054
8.19 k-Means Clustering . . . 1055
8.19.1 Introduction . . . 1055
8.19.2 Prerequisites . . . 1055
8.19.3 Theory . . . 1055
8.19.4 Assumptions . . . 1055
8.19.5 R Implementation . . . 1056
8.19.6 Output & Results . . . 1056
8.19.7 Interpretation . . . 1056
8.19.8 Practical Tips . . . 1056
8.19.9 R Packages Used . . . 1057
8.19.10 For Reviewers . . . 1057
8.19.11 See also — labs in this chapter . . . 1057
8.20 Linear Discriminant Analysis . . . 1057
8.20.1 Introduction . . . 1057
8.20.2 Prerequisites . . . 1057
8.20.3 Theory . . . 1058
8.20.4 Assumptions . . . 1058
8.20.5 R Implementation . . . 1058
8.20.6 Output & Results . . . 1058
8.20.7 Interpretation . . . 1059
8.20.8 Practical Tips . . . 1059
8.20.9 R Packages Used . . . 1059
8.20.10 For Reviewers . . . 1059
8.20.11 See also — labs in this chapter . . . 1059
8.21 Mahalanobis Distance . . . 1060
8.21.1 Introduction . . . 1060
8.21.2 Prerequisites . . . 1060
8.21.3 Theory . . . 1060
8.21.4 Assumptions . . . 1060
8.21.5 R Implementation . . . 1060
8.21.6 Output & Results . . . 1061

8.21.7 Interpretation 1061
8.21.8 Practical Tips 1061
8.21.9 R Packages Used 1062
8.21.10 For Reviewers 1062
8.21.11 See also — labs in this chapter 1062
8.22 MANOVA 1062
8.22.1 Introduction 1062
8.22.2 Prerequisites 1062
8.22.3 Theory 1062
8.22.4 Assumptions 1063
8.22.5 R Implementation 1063
8.22.6 Output & Results 1063
8.22.7 Interpretation 1063
8.22.8 Practical Tips 1064
8.22.9 R Packages Used 1064
8.22.10 For Reviewers 1064
8.22.11 See also — labs in this chapter 1064
8.23 Metric Multidimensional Scaling 1065
8.23.1 Introduction 1065
8.23.2 Prerequisites 1065
8.23.3 Theory 1065
8.23.4 Assumptions 1065
8.23.5 R Implementation 1065
8.23.6 Output & Results 1066
8.23.7 Interpretation 1066
8.23.8 Practical Tips 1066
8.23.9 R Packages Used 1066
8.23.10 For Reviewers 1067
8.23.11 See also — labs in this chapter 1067
8.24 Non-Metric Multidimensional Scaling 1067
8.24.1 Introduction 1067
8.24.2 Prerequisites 1067
8.24.3 Theory 1067
8.24.4 Assumptions 1068
8.24.5 R Implementation 1068
8.24.6 Output & Results 1068
8.24.7 Interpretation 1068
8.24.8 Practical Tips 1068
8.24.9 R Packages Used 1069
8.24.10 For Reviewers 1069
8.24.11 See also — labs in this chapter 1069
8.25 Multiple Correspondence Analysis 1069
8.25.1 Introduction 1069
8.25.2 Prerequisites 1070
8.25.3 Theory 1070
8.25.4 Assumptions 1070

8.25.5 R Implementation . . . 1070
8.25.6 Output & Results . . . 1070
8.25.7 Interpretation . . . 1071
8.25.8 Practical Tips . . . 1071
8.25.9 R Packages Used . . . 1071
8.25.10 For Reviewers . . . 1071
8.25.11 See also — labs in this chapter . . . 1071
8.26 Partitioning Around Medoids . . . 1072
8.26.1 Introduction . . . 1072
8.26.2 Prerequisites . . . 1072
8.26.3 Theory . . . 1072
8.26.4 Assumptions . . . 1072
8.26.5 R Implementation . . . 1072
8.26.6 Output & Results . . . 1073
8.26.7 Interpretation . . . 1073
8.26.8 Practical Tips . . . 1073
8.26.9 R Packages Used . . . 1074
8.26.10 For Reviewers . . . 1074
8.26.11 See also — labs in this chapter . . . 1074
8.27 PLS Discriminant Analysis . . . 1074
8.27.1 Introduction . . . 1074
8.27.2 Prerequisites . . . 1074
8.27.3 Theory . . . 1074
8.27.4 Assumptions . . . 1075
8.27.5 R Implementation . . . 1075
8.27.6 Output & Results . . . 1075
8.27.7 Interpretation . . . 1075
8.27.8 Practical Tips . . . 1076
8.27.9 R Packages Used . . . 1076
8.27.10 For Reviewers . . . 1076
8.27.11 See also — labs in this chapter . . . 1076
8.28 Partial Least Squares Regression . . . 1077
8.28.1 Introduction . . . 1077
8.28.2 Prerequisites . . . 1077
8.28.3 Theory . . . 1077
8.28.4 Assumptions . . . 1077
8.28.5 R Implementation . . . 1077
8.28.6 Output & Results . . . 1078
8.28.7 Interpretation . . . 1078
8.28.8 Practical Tips . . . 1078
8.28.9 R Packages Used . . . 1078
8.28.10 For Reviewers . . . 1079
8.28.11 See also — labs in this chapter . . . 1079
8.29 Principal Component Analysis . . . 1079
8.29.1 Introduction . . . 1079
8.29.2 Prerequisites . . . 1079

8.29.3 Theory . . . 1079
8.29.4 Assumptions . . . 1080
8.29.5 R Implementation . . . 1080
8.29.6 Output & Results . . . 1081
8.29.7 Interpretation . . . 1081
8.29.8 Practical Tips . . . 1081
8.29.9 For Reviewers . . . 1081
8.29.10 See also — labs in this chapter . . . 1081
8.30 Quadratic Discriminant Analysis . . . 1082
8.30.1 Introduction . . . 1082
8.30.2 Prerequisites . . . 1082
8.30.3 Theory . . . 1082
8.30.4 Assumptions . . . 1082
8.30.5 R Implementation . . . 1082
8.30.6 Output & Results . . . 1083
8.30.7 Interpretation . . . 1083
8.30.8 Practical Tips . . . 1083
8.30.9 R Packages Used . . . 1083
8.30.10 For Reviewers . . . 1084
8.30.11 See also — labs in this chapter . . . 1084
8.31 Structural Equation Models . . . 1084
8.31.1 Introduction . . . 1084
8.31.2 Prerequisites . . . 1084
8.31.3 Theory . . . 1084
8.31.4 Assumptions . . . 1085
8.31.5 R Implementation . . . 1085
8.31.6 Output & Results . . . 1085
8.31.7 Interpretation . . . 1086
8.31.8 Practical Tips . . . 1086
8.31.9 R Packages Used . . . 1086
8.31.10 For Reviewers . . . 1086
8.31.11 See also — labs in this chapter . . . 1086
8.32 UMAP and t-SNE: Overview . . . 1087
8.32.1 Introduction . . . 1087
8.32.2 Prerequisites . . . 1087
8.32.3 Theory . . . 1087
8.32.4 Assumptions . . . 1088
8.32.5 R Implementation . . . 1088
8.32.6 Output & Results . . . 1088
8.32.7 Interpretation . . . 1088
8.32.8 Practical Tips . . . 1088
8.32.9 R Packages Used . . . 1089
8.32.10 For Reviewers . . . 1089
8.32.11 See also — labs in this chapter . . . 1089
8.33 Week 1, Session 4 — Clustering: k-means, hierarchical, model-based . . . 1089

8.33.1 Learning objectives . . . 1089
8.33.2 Prerequisites . . . 1090
8.33.3 Background . . . 1090
8.33.4 Setup . . . 1090
8.33.5 1. Goal . . . 1090
8.33.6 2. Approach . . . 1090
8.33.7 3. Execution . . . 1091
8.33.8 4. Check . . . 1091
8.33.9 5. Report . . . 1091
8.33.10 Common pitfalls . . . 1092
8.33.11 Further reading . . . 1092
8.33.12 Session info . . . 1092
8.33.13 See also — chapter index . . . 1092
8.34 Week 1, Session 3 — PCA, factor analysis, CCA, LDA . . . 1092
8.34.1 Learning objectives . . . 1092
8.34.2 Prerequisites . . . 1092
8.34.3 Background . . . 1093
8.34.4 Setup . . . 1093
8.34.5 1. Goal . . . 1093
8.34.6 2. Approach . . . 1093
8.34.7 3. Execution . . . 1094
8.34.8 4. Check . . . 1094
8.34.9 5. Report . . . 1094
8.34.10 Common pitfalls . . . 1094
8.34.11 Further reading . . . 1095
8.34.12 Session info . . . 1095
8.34.13 See also — chapter index . . . 1095
8.35 Week 1, Session 5 — UMAP and t-SNE . . . 1095
8.35.1 Learning objectives . . . 1095
8.35.2 Prerequisites . . . 1095
8.35.3 Background . . . 1095
8.35.4 Setup . . . 1096
8.35.5 1. Goal . . . 1096
8.35.6 2. Approach . . . 1096
8.35.7 3. Execution . . . 1096
8.35.8 4. Check . . . 1097
8.35.9 5. Report . . . 1097
8.35.10 Common pitfalls . . . 1097
8.35.11 Further reading . . . 1097
8.35.12 Session info . . . 1097
8.35.13 See also — chapter index . . . 1098

9 Time-Series Analysis 1099
9.1 Method pages . . . 1099
9.2 Labs . . . 1100
9.3 Interpreting ACF and PACF . . . 1100

9.3.1 Introduction . . . 1100
9.3.2 Prerequisites . . . 1100
9.3.3 Theory . . . 1100
9.3.4 Assumptions . . . 1101
9.3.5 R Implementation . . . 1101
9.3.6 Output & Results . . . 1101
9.3.7 Interpretation . . . 1102
9.3.8 Practical Tips . . . 1102
9.3.9 R Packages Used . . . 1102
9.3.10 For Reviewers . . . 1102
9.3.11 See also — labs in this chapter . . . 1102
9.4 Augmented Dickey-Fuller Test . . . 1103
9.4.1 Introduction . . . 1103
9.4.2 Prerequisites . . . 1103
9.4.3 Theory . . . 1103
9.4.4 Assumptions . . . 1103
9.4.5 R Implementation . . . 1103
9.4.6 Output & Results . . . 1104
9.4.7 Interpretation . . . 1104
9.4.8 Practical Tips . . . 1104
9.4.9 R Packages Used . . . 1104
9.4.10 For Reviewers . . . 1105
9.4.11 See also — labs in this chapter . . . 1105
9.5 ARIMA Models . . . 1105
9.5.1 Introduction . . . 1105
9.5.2 Prerequisites . . . 1105
9.5.3 Theory . . . 1105
9.5.4 Assumptions . . . 1106
9.5.5 R Implementation . . . 1106
9.5.6 Output & Results . . . 1107
9.5.7 Interpretation . . . 1107
9.5.8 Practical Tips . . . 1107
9.5.9 For Reviewers . . . 1107
9.5.10 See also — labs in this chapter . . . 1107
9.6 Bayesian Changepoint Detection . . . 1107
9.6.1 Introduction . . . 1107
9.6.2 Prerequisites . . . 1108
9.6.3 Theory . . . 1108
9.6.4 Assumptions . . . 1108
9.6.5 R Implementation . . . 1108
9.6.6 Output & Results . . . 1109
9.6.7 Interpretation . . . 1109
9.6.8 Practical Tips . . . 1109
9.6.9 R Packages Used . . . 1109
9.6.10 For Reviewers . . . 1110
9.6.11 See also — labs in this chapter . . . 1110

9.7 Changepoint Detection . . . 1110
9.7.1 Introduction . . . 1110
9.7.2 Prerequisites . . . 1110
9.7.3 Theory . . . 1110
9.7.4 Assumptions . . . 1111
9.7.5 R Implementation . . . 1111
9.7.6 Output & Results . . . 1111
9.7.7 Interpretation . . . 1111
9.7.8 Practical Tips . . . 1111
9.7.9 R Packages Used . . . 1112
9.7.10 For Reviewers . . . 1112
9.7.11 See also — labs in this chapter . . . 1112
9.8 Diebold-Mariano Test . . . 1112
9.8.1 Introduction . . . 1112
9.8.2 Prerequisites . . . 1113
9.8.3 Theory . . . 1113
9.8.4 Assumptions . . . 1113
9.8.5 R Implementation . . . 1113
9.8.6 Output & Results . . . 1113
9.8.7 Interpretation . . . 1113
9.8.8 Practical Tips . . . 1114
9.8.9 R Packages Used . . . 1114
9.8.10 For Reviewers . . . 1114
9.8.11 See also — labs in this chapter . . . 1114
9.9 Differencing a Time Series . . . 1114
9.9.1 Introduction . . . 1114
9.9.2 Prerequisites . . . 1115
9.9.3 Theory . . . 1115
9.9.4 Assumptions . . . 1115
9.9.5 R Implementation . . . 1115
9.9.6 Output & Results . . . 1116
9.9.7 Interpretation . . . 1116
9.9.8 Practical Tips . . . 1116
9.9.9 R Packages Used . . . 1116
9.9.10 For Reviewers . . . 1117
9.9.11 See also — labs in this chapter . . . 1117
9.10 ETS Models . . . 1117
9.10.1 Introduction . . . 1117
9.10.2 Prerequisites . . . 1117
9.10.3 Theory . . . 1117
9.10.4 Assumptions . . . 1118
9.10.5 R Implementation . . . 1118
9.10.6 Output & Results . . . 1118
9.10.7 Interpretation . . . 1118
9.10.8 Practical Tips . . . 1118
9.10.9 R Packages Used . . . 1119

9.10.10 For Reviewers . . . 1119
9.10.11 See also — labs in this chapter . . . 1119
9.11 Exponential Smoothing . . . 1119
9.11.1 Introduction . . . 1119
9.11.2 Prerequisites . . . 1120
9.11.3 Theory . . . 1120
9.11.4 Assumptions . . . 1120
9.11.5 R Implementation . . . 1120
9.11.6 Output & Results . . . 1121
9.11.7 Interpretation . . . 1121
9.11.8 Practical Tips . . . 1121
9.11.9 R Packages Used . . . 1121
9.11.10 For Reviewers . . . 1121
9.11.11 See also — labs in this chapter . . . 1122
9.12 Forecast Accuracy Metrics . . . 1122
9.12.1 Introduction . . . 1122
9.12.2 Prerequisites . . . 1122
9.12.3 Theory . . . 1122
9.12.4 Assumptions . . . 1122
9.12.5 R Implementation . . . 1123
9.12.6 Output & Results . . . 1123
9.12.7 Interpretation . . . 1123
9.12.8 Practical Tips . . . 1123
9.12.9 R Packages Used . . . 1124
9.12.10 For Reviewers . . . 1124
9.12.11 See also — labs in this chapter . . . 1124
9.13 GARCH Models . . . 1124
9.13.1 Introduction . . . 1124
9.13.2 Prerequisites . . . 1124
9.13.3 Theory . . . 1124
9.13.4 Assumptions . . . 1125
9.13.5 R Implementation . . . 1125
9.13.6 Output & Results . . . 1125
9.13.7 Interpretation . . . 1126
9.13.8 Practical Tips . . . 1126
9.13.9 R Packages Used . . . 1126
9.13.10 For Reviewers . . . 1126
9.13.11 See also — labs in this chapter . . . 1127
9.14 Granger Causality . . . 1127
9.14.1 Introduction . . . 1127
9.14.2 Prerequisites . . . 1127
9.14.3 Theory . . . 1127
9.14.4 Assumptions . . . 1127
9.14.5 R Implementation . . . 1127
9.14.6 Output & Results . . . 1128
9.14.7 Interpretation . . . 1128

9.14.8 Practical Tips . . . 1128
9.14.9 R Packages Used . . . 1128
9.14.10 For Reviewers . . . 1129
9.14.11 See also — labs in this chapter . . . 1129
9.15 Holt-Winters Method . . . 1129
9.15.1 Introduction . . . 1129
9.15.2 Prerequisites . . . 1129
9.15.3 Theory . . . 1129
9.15.4 Assumptions . . . 1130
9.15.5 R Implementation . . . 1130
9.15.6 Output & Results . . . 1130
9.15.7 Interpretation . . . 1130
9.15.8 Practical Tips . . . 1130
9.15.9 R Packages Used . . . 1131
9.15.10 For Reviewers . . . 1131
9.15.11 See also — labs in this chapter . . . 1131
9.16 The Kalman Filter . . . 1131
9.16.1 Introduction . . . 1131
9.16.2 Prerequisites . . . 1131
9.16.3 Theory . . . 1131
9.16.4 Assumptions . . . 1132
9.16.5 R Implementation . . . 1132
9.16.6 Output & Results . . . 1132
9.16.7 Interpretation . . . 1133
9.16.8 Practical Tips . . . 1133
9.16.9 R Packages Used . . . 1133
9.16.10 For Reviewers . . . 1133
9.16.11 See also — labs in this chapter . . . 1134
9.17 KPSS Test . . . 1134
9.17.1 Introduction . . . 1134
9.17.2 Prerequisites . . . 1134
9.17.3 Theory . . . 1134
9.17.4 Assumptions . . . 1134
9.17.5 R Implementation . . . 1134
9.17.6 Output & Results . . . 1135
9.17.7 Interpretation . . . 1135
9.17.8 Practical Tips . . . 1135
9.17.9 R Packages Used . . . 1136
9.17.10 For Reviewers . . . 1136
9.17.11 See also — labs in this chapter . . . 1136
9.18 The Ljung-Box Test . . . 1136
9.18.1 Introduction . . . 1136
9.18.2 Prerequisites . . . 1136
9.18.3 Theory . . . 1136
9.18.4 Assumptions . . . 1137
9.18.5 R Implementation . . . 1137

9.18.6 Output & Results 1137
9.18.7 Interpretation 1137
9.18.8 Practical Tips 1137
9.18.9 R Packages Used 1138
9.18.10 For Reviewers 1138
9.18.11 See also — labs in this chapter 1138
9.19 Moving Averages 1138
9.19.1 Introduction 1138
9.19.2 Prerequisites 1138
9.19.3 Theory 1139
9.19.4 Assumptions 1139
9.19.5 R Implementation 1139
9.19.6 Output & Results 1139
9.19.7 Interpretation 1140
9.19.8 Practical Tips 1140
9.19.9 R Packages Used 1140
9.19.10 For Reviewers 1140
9.19.11 See also — labs in this chapter 1140
9.20 Vector Autoregression (VAR) 1141
9.20.1 Introduction 1141
9.20.2 Prerequisites 1141
9.20.3 Theory 1141
9.20.4 Assumptions 1141
9.20.5 R Implementation 1141
9.20.6 Output & Results 1142
9.20.7 Interpretation 1142
9.20.8 Practical Tips 1142
9.20.9 R Packages Used 1143
9.20.10 For Reviewers 1143
9.20.11 See also — labs in this chapter 1143
9.21 Phillips-Perron Test 1143
9.21.1 Introduction 1143
9.21.2 Prerequisites 1143
9.21.3 Theory 1143
9.21.4 Assumptions 1144
9.21.5 R Implementation 1144
9.21.6 Output & Results 1144
9.21.7 Interpretation 1144
9.21.8 Practical Tips 1144
9.21.9 R Packages Used 1145
9.21.10 For Reviewers 1145
9.21.11 See also — labs in this chapter 1145
9.22 Rolling-Origin Cross-Validation 1145
9.22.1 Introduction 1145
9.22.2 Prerequisites 1145
9.22.3 Theory 1146

9.22.4 Assumptions . . . 1146
9.22.5 R Implementation . . . 1146
9.22.6 Output & Results . . . 1146
9.22.7 Interpretation . . . 1147
9.22.8 Practical Tips . . . 1147
9.22.9 R Packages Used . . . 1147
9.22.10 For Reviewers . . . 1147
9.22.11 See also — labs in this chapter . . . 1147
9.23 Seasonal ARIMA (SARIMA) . . . 1148
9.23.1 Introduction . . . 1148
9.23.2 Prerequisites . . . 1148
9.23.3 Theory . . . 1148
9.23.4 Assumptions . . . 1148
9.23.5 R Implementation . . . 1148
9.23.6 Output & Results . . . 1149
9.23.7 Interpretation . . . 1149
9.23.8 Practical Tips . . . 1149
9.23.9 R Packages Used . . . 1149
9.23.10 For Reviewers . . . 1150
9.23.11 See also — labs in this chapter . . . 1150
9.24 Seasonal Decomposition (STL) . . . 1150
9.24.1 Introduction . . . 1150
9.24.2 Prerequisites . . . 1150
9.24.3 Theory . . . 1150
9.24.4 Assumptions . . . 1151
9.24.5 R Implementation . . . 1151
9.24.6 Output & Results . . . 1151
9.24.7 Interpretation . . . 1151
9.24.8 Practical Tips . . . 1152
9.24.9 R Packages Used . . . 1152
9.24.10 For Reviewers . . . 1152
9.24.11 See also — labs in this chapter . . . 1152
9.25 Spectral Analysis . . . 1152
9.25.1 Introduction . . . 1152
9.25.2 Prerequisites . . . 1153
9.25.3 Theory . . . 1153
9.25.4 Assumptions . . . 1153
9.25.5 R Implementation . . . 1153
9.25.6 Output & Results . . . 1154
9.25.7 Interpretation . . . 1154
9.25.8 Practical Tips . . . 1154
9.25.9 R Packages Used . . . 1154
9.25.10 For Reviewers . . . 1154
9.25.11 See also — labs in this chapter . . . 1155
9.26 State-Space Models . . . 1155
9.26.1 Introduction . . . 1155

9.26.2 Prerequisites 1155
9.26.3 Theory 1155
9.26.4 Assumptions 1155
9.26.5 R Implementation 1155
9.26.6 Output & Results 1156
9.26.7 Interpretation 1156
9.26.8 Practical Tips 1156
9.26.9 R Packages Used 1157
9.26.10 For Reviewers 1157
9.26.11 See also — labs in this chapter 1157
9.27 Stationarity Tests 1157
9.27.1 Introduction 1157
9.27.2 Prerequisites 1157
9.27.3 Theory 1158
9.27.4 Assumptions 1158
9.27.5 R Implementation 1158
9.27.6 Output & Results 1159
9.27.7 Interpretation 1159
9.27.8 Practical Tips 1159
9.27.9 R Packages Used 1159
9.27.10 For Reviewers 1159
9.27.11 See also — labs in this chapter 1160
9.28 Time Series: Introduction 1160
9.28.1 Introduction 1160
9.28.2 Prerequisites 1160
9.28.3 Theory 1160
9.28.4 Assumptions 1160
9.28.5 R Implementation 1161
9.28.6 Output & Results 1161
9.28.7 Interpretation 1161
9.28.8 Practical Tips 1161
9.28.9 R Packages Used 1162
9.28.10 For Reviewers 1162
9.28.11 See also — labs in this chapter 1162
9.29 VECM and Cointegration 1162
9.29.1 Introduction 1162
9.29.2 Prerequisites 1162
9.29.3 Theory 1162
9.29.4 Assumptions 1163
9.29.5 R Implementation 1163
9.29.6 Output & Results 1163
9.29.7 Interpretation 1163
9.29.8 Practical Tips 1164
9.29.9 R Packages Used 1164
9.29.10 For Reviewers 1164
9.29.11 See also — labs in this chapter 1164

9.30 Wavelet Analysis . . . 1164
9.30.1 Introduction . . . 1164
9.30.2 Prerequisites . . . 1165
9.30.3 Theory . . . 1165
9.30.4 Assumptions . . . 1165
9.30.5 R Implementation . . . 1165
9.30.6 Output & Results . . . 1166
9.30.7 Interpretation . . . 1166
9.30.8 Practical Tips . . . 1166
9.30.9 R Packages Used . . . 1166
9.30.10 For Reviewers . . . 1166
9.30.11 See also — labs in this chapter . . . 1167
9.31 White Noise Tests . . . 1167
9.31.1 Introduction . . . 1167
9.31.2 Prerequisites . . . 1167
9.31.3 Theory . . . 1167
9.31.4 Assumptions . . . 1168
9.31.5 R Implementation . . . 1168
9.31.6 Output & Results . . . 1168
9.31.7 Interpretation . . . 1168
9.31.8 Practical Tips . . . 1168
9.31.9 R Packages Used . . . 1169
9.31.10 For Reviewers . . . 1169
9.31.11 See also — labs in this chapter . . . 1169
9.32 X-13ARIMA-SEATS Decomposition . . . 1169
9.32.1 Introduction . . . 1169
9.32.2 Prerequisites . . . 1169
9.32.3 Theory . . . 1170
9.32.4 Assumptions . . . 1170
9.32.5 R Implementation . . . 1170
9.32.6 Output & Results . . . 1170
9.32.7 Interpretation . . . 1171
9.32.8 Practical Tips . . . 1171
9.32.9 R Packages Used . . . 1171
9.32.10 For Reviewers . . . 1171
9.32.11 See also — labs in this chapter . . . 1171
9.33 Week 2, Session 5 — Time series basics . . . 1172
9.33.1 Learning objectives . . . 1172
9.33.2 Prerequisites . . . 1172
9.33.3 Background . . . 1172
9.33.4 Setup . . . 1172
9.33.5 1. Hypothesis . . . 1173
9.33.6 2. Visualise . . . 1173
9.33.7 3. Assumptions . . . 1173
9.33.8 4. Conduct . . . 1173
9.33.9 5. Concluding statement . . . 1174

9.33.10 Common pitfalls . . . 1174
9.33.11 Further reading . . . 1174
9.33.12 Session info . . . 1174
9.33.13 See also — chapter index . . . 1174

10 Bayesian Statistics **1175**
10.1 Method pages . . . 1175
10.2 Labs . . . 1176
10.3 Bayes Factors . . . 1176
10.3.1 Introduction . . . 1176
10.3.2 Prerequisites . . . 1177
10.3.3 Theory . . . 1177
10.3.4 Assumptions . . . 1177
10.3.5 R Implementation . . . 1177
10.3.6 Output & Results . . . 1177
10.3.7 Interpretation . . . 1178
10.3.8 Practical Tips . . . 1178
10.3.9 R Packages Used . . . 1178
10.3.10 For Reviewers . . . 1178
10.3.11 See also — labs in this chapter . . . 1178
10.4 Bayesian Hypothesis Testing . . . 1179
10.4.1 Introduction . . . 1179
10.4.2 Prerequisites . . . 1179
10.4.3 Theory . . . 1179
10.4.4 Assumptions . . . 1179
10.4.5 R Implementation . . . 1179
10.4.6 Output & Results . . . 1180
10.4.7 Interpretation . . . 1180
10.4.8 Practical Tips . . . 1180
10.4.9 R Packages Used . . . 1180
10.4.10 For Reviewers . . . 1181
10.4.11 See also — labs in this chapter . . . 1181
10.5 Bayes' Theorem for Parameters . . . 1181
10.5.1 Introduction . . . 1181
10.5.2 Prerequisites . . . 1181
10.5.3 Theory . . . 1181
10.5.4 Assumptions . . . 1182
10.5.5 R Implementation . . . 1182
10.5.6 Output & Results . . . 1182
10.5.7 Interpretation . . . 1182
10.5.8 Practical Tips . . . 1183
10.5.9 R Packages Used . . . 1183
10.5.10 For Reviewers . . . 1183
10.5.11 See also — labs in this chapter . . . 1183
10.6 Bayesian ANOVA . . . 1183
10.6.1 Introduction . . . 1183

10.6.2 Prerequisites . . . 1184
10.6.3 Theory . . . 1184
10.6.4 Assumptions . . . 1184
10.6.5 R Implementation . . . 1184
10.6.6 Output & Results . . . 1185
10.6.7 Interpretation . . . 1185
10.6.8 Practical Tips . . . 1185
10.6.9 R Packages Used . . . 1185
10.6.10 For Reviewers . . . 1185
10.6.11 See also — labs in this chapter . . . 1186
10.7 Bayesian Hierarchical Models . . . 1186
10.7.1 Introduction . . . 1186
10.7.2 Prerequisites . . . 1186
10.7.3 Theory . . . 1186
10.7.4 Assumptions . . . 1187
10.7.5 R Implementation . . . 1187
10.7.6 Output & Results . . . 1187
10.7.7 Interpretation . . . 1187
10.7.8 Practical Tips . . . 1188
10.7.9 R Packages Used . . . 1188
10.7.10 For Reviewers . . . 1188
10.7.11 See also — labs in this chapter . . . 1188
10.8 Bayesian Mediation Analysis . . . 1188
10.8.1 Introduction . . . 1188
10.8.2 Prerequisites . . . 1189
10.8.3 Theory . . . 1189
10.8.4 Assumptions . . . 1189
10.8.5 R Implementation . . . 1189
10.8.6 Output & Results . . . 1190
10.8.7 Interpretation . . . 1190
10.8.8 Practical Tips . . . 1190
10.8.9 R Packages Used . . . 1191
10.8.10 For Reviewers . . . 1191
10.8.11 See also — labs in this chapter . . . 1191
10.9 Bayesian Meta-Analysis . . . 1191
10.9.1 Introduction . . . 1191
10.9.2 Prerequisites . . . 1191
10.9.3 Theory . . . 1191
10.9.4 Assumptions . . . 1192
10.9.5 R Implementation . . . 1192
10.9.6 Output & Results . . . 1192
10.9.7 Interpretation . . . 1192
10.9.8 Practical Tips . . . 1193
10.9.9 R Packages Used . . . 1193
10.9.10 For Reviewers . . . 1193
10.9.11 See also — labs in this chapter . . . 1193

10.10 Bayesian Mixed Models 1193
10.10.1 Introduction 1193
10.10.2 Prerequisites 1194
10.10.3 Theory 1194
10.10.4 Assumptions 1194
10.10.5 R Implementation 1194
10.10.6 Output & Results 1195
10.10.7 Interpretation 1195
10.10.8 Practical Tips 1195
10.10.9 R Packages Used 1196
10.10.10 For Reviewers 1196
10.10.11 See also — labs in this chapter 1196
10.11 The Beta-Binomial Model 1196
10.11.1 Introduction 1196
10.11.2 Prerequisites 1196
10.11.3 Theory 1196
10.11.4 Assumptions 1197
10.11.5 R Implementation 1197
10.11.6 Output & Results 1197
10.11.7 Interpretation 1198
10.11.8 Practical Tips 1198
10.11.9 R Packages Used 1198
10.11.10 For Reviewers 1198
10.11.11 See also — labs in this chapter 1198
10.12 brms Basics 1199
10.12.1 Introduction 1199
10.12.2 Prerequisites 1199
10.12.3 Theory 1199
10.12.4 Assumptions 1199
10.12.5 R Implementation 1199
10.12.6 Output & Results 1200
10.12.7 Interpretation 1200
10.12.8 Practical Tips 1200
10.12.9 R Packages Used 1201
10.12.10 For Reviewers 1201
10.12.11 See also — labs in this chapter 1201
10.13 Conjugate Priors 1201
10.13.1 Introduction 1201
10.13.2 Prerequisites 1201
10.13.3 Theory 1201
10.13.4 Assumptions 1202
10.13.5 R Implementation 1202
10.13.6 Output & Results 1203
10.13.7 Interpretation 1203
10.13.8 Practical Tips 1203
10.13.9 For Reviewers 1204

10.13.10 See also — labs in this chapter 1204
10.14 Credible Intervals 1204
10.14.1 Introduction 1204
10.14.2 Prerequisites 1204
10.14.3 Theory 1204
10.14.4 Assumptions 1205
10.14.5 R Implementation 1205
10.14.6 Output & Results 1205
10.14.7 Interpretation 1205
10.14.8 Practical Tips 1206
10.14.9 R Packages Used 1206
10.14.10 For Reviewers 1206
10.14.11 See also — labs in this chapter 1206
10.15 Divergences and Pairs Plots 1206
10.15.1 Introduction 1206
10.15.2 Prerequisites 1207
10.15.3 Theory 1207
10.15.4 Assumptions 1207
10.15.5 R Implementation 1207
10.15.6 Output & Results 1208
10.15.7 Interpretation 1208
10.15.8 Practical Tips 1208
10.15.9 R Packages Used 1208
10.15.10 For Reviewers 1208
10.15.11 See also — labs in this chapter 1209
10.16 The Gamma-Poisson Model 1209
10.16.1 Introduction 1209
10.16.2 Prerequisites 1209
10.16.3 Theory 1209
10.16.4 Assumptions 1209
10.16.5 R Implementation 1210
10.16.6 Output & Results 1210
10.16.7 Interpretation 1210
10.16.8 Practical Tips 1210
10.16.9 R Packages Used 1211
10.16.10 For Reviewers 1211
10.16.11 See also — labs in this chapter 1211
10.17 Gibbs Sampling 1211
10.17.1 Introduction 1211
10.17.2 Prerequisites 1211
10.17.3 Theory 1212
10.17.4 Assumptions 1212
10.17.5 R Implementation 1212
10.17.6 Output & Results 1213
10.17.7 Interpretation 1213
10.17.8 Practical Tips 1213

10.17.9 R Packages Used . . . 1213
10.17.10 For Reviewers . . . 1214
10.17.11 See also — labs in this chapter . . . 1214
10.18 Hamiltonian Monte Carlo . . . 1214
10.18.1 Introduction . . . 1214
10.18.2 Prerequisites . . . 1214
10.18.3 Theory . . . 1214
10.18.4 Assumptions . . . 1215
10.18.5 R Implementation . . . 1215
10.18.6 Output & Results . . . 1215
10.18.7 Interpretation . . . 1215
10.18.8 Practical Tips . . . 1216
10.18.9 R Packages Used . . . 1216
10.18.10 For Reviewers . . . 1216
10.18.11 See also — labs in this chapter . . . 1216
10.19 Highest Posterior Density Interval . . . 1216
10.19.1 Introduction . . . 1216
10.19.2 Prerequisites . . . 1217
10.19.3 Theory . . . 1217
10.19.4 Assumptions . . . 1217
10.19.5 R Implementation . . . 1217
10.19.6 Output & Results . . . 1217
10.19.7 Interpretation . . . 1218
10.19.8 Practical Tips . . . 1218
10.19.9 R Packages Used . . . 1218
10.19.10 For Reviewers . . . 1218
10.19.11 See also — labs in this chapter . . . 1219
10.20 Informative Priors . . . 1219
10.20.1 Introduction . . . 1219
10.20.2 Prerequisites . . . 1219
10.20.3 Theory . . . 1219
10.20.4 Assumptions . . . 1219
10.20.5 R Implementation . . . 1220
10.20.6 Output & Results . . . 1220
10.20.7 Interpretation . . . 1220
10.20.8 Practical Tips . . . 1221
10.20.9 R Packages Used . . . 1221
10.20.10 For Reviewers . . . 1221
10.20.11 See also — labs in this chapter . . . 1221
10.21 Jeffreys' Prior . . . 1221
10.21.1 Introduction . . . 1221
10.21.2 Prerequisites . . . 1222
10.21.3 Theory . . . 1222
10.21.4 Assumptions . . . 1222
10.21.5 R Implementation . . . 1222
10.21.6 Output & Results . . . 1223

10.21.7 Interpretation . . . 1223
10.21.8 Practical Tips . . . 1223
10.21.9 R Packages Used . . . 1224
10.21.10 For Reviewers . . . 1224
10.21.11 See also — labs in this chapter . . . 1224
10.22 Bayesian Linear Regression . . . 1224
10.22.1 Introduction . . . 1224
10.22.2 Prerequisites . . . 1224
10.22.3 Theory . . . 1224
10.22.4 Assumptions . . . 1225
10.22.5 R Implementation . . . 1225
10.22.6 Output & Results . . . 1225
10.22.7 Interpretation . . . 1225
10.22.8 Practical Tips . . . 1226
10.22.9 R Packages Used . . . 1226
10.22.10 For Reviewers . . . 1226
10.22.11 See also — labs in this chapter . . . 1226
10.23 Bayesian Logistic Regression . . . 1226
10.23.1 Introduction . . . 1226
10.23.2 Prerequisites . . . 1227
10.23.3 Theory . . . 1227
10.23.4 Assumptions . . . 1227
10.23.5 R Implementation . . . 1227
10.23.6 Output & Results . . . 1228
10.23.7 Interpretation . . . 1228
10.23.8 Practical Tips . . . 1228
10.23.9 R Packages Used . . . 1228
10.23.10 For Reviewers . . . 1229
10.23.11 See also — labs in this chapter . . . 1229
10.24 Leave-One-Out Cross-Validation . . . 1229
10.24.1 Introduction . . . 1229
10.24.2 Prerequisites . . . 1229
10.24.3 Theory . . . 1229
10.24.4 Assumptions . . . 1230
10.24.5 R Implementation . . . 1230
10.24.6 Output & Results . . . 1230
10.24.7 Interpretation . . . 1230
10.24.8 Practical Tips . . . 1231
10.24.9 R Packages Used . . . 1231
10.24.10 For Reviewers . . . 1231
10.24.11 See also — labs in this chapter . . . 1231
10.25 Metropolis-Hastings . . . 1231
10.25.1 Introduction . . . 1231
10.25.2 Prerequisites . . . 1232
10.25.3 Theory . . . 1232
10.25.4 Assumptions . . . 1232

10.25.5 R Implementation 1232
10.25.6 Output & Results 1233
10.25.7 Interpretation 1233
10.25.8 Practical Tips 1233
10.25.9 R Packages Used 1234
10.25.10 For Reviewers 1234
10.25.11 See also — labs in this chapter 1234
10.26 The Normal-Normal Model 1234
10.26.1 Introduction 1234
10.26.2 Prerequisites 1234
10.26.3 Theory 1234
10.26.4 Assumptions 1235
10.26.5 R Implementation 1235
10.26.6 Output & Results 1235
10.26.7 Interpretation 1236
10.26.8 Practical Tips 1236
10.26.9 R Packages Used 1236
10.26.10 For Reviewers 1236
10.26.11 See also — labs in this chapter 1236
10.27 Posterior Mean and Variance 1237
10.27.1 Introduction 1237
10.27.2 Prerequisites 1237
10.27.3 Theory 1237
10.27.4 Assumptions 1237
10.27.5 R Implementation 1238
10.27.6 Output & Results 1238
10.27.7 Interpretation 1238
10.27.8 Practical Tips 1238
10.27.9 R Packages Used 1239
10.27.10 For Reviewers 1239
10.27.11 See also — labs in this chapter 1239
10.28 Posterior Predictive Checks 1239
10.28.1 Introduction 1239
10.28.2 Prerequisites 1239
10.28.3 Theory 1239
10.28.4 Assumptions 1240
10.28.5 R Implementation 1240
10.28.6 Output & Results 1240
10.28.7 Interpretation 1241
10.28.8 Practical Tips 1241
10.28.9 R Packages Used 1241
10.28.10 For Reviewers 1241
10.28.11 See also — labs in this chapter 1241
10.29 The Posterior Predictive Distribution 1242
10.29.1 Introduction 1242
10.29.2 Prerequisites 1242

10.29.3 Theory . . . 1242
10.29.4 Assumptions . . . 1242
10.29.5 R Implementation . . . 1242
10.29.6 Output & Results . . . 1243
10.29.7 Interpretation . . . 1243
10.29.8 Practical Tips . . . 1243
10.29.9 R Packages Used . . . 1244
10.29.10 For Reviewers . . . 1244
10.29.11 See also — labs in this chapter . . . 1244
10.30 Prior Specification . . . 1244
10.30.1 Introduction . . . 1244
10.30.2 Prerequisites . . . 1244
10.30.3 Theory . . . 1244
10.30.4 Assumptions . . . 1245
10.30.5 R Implementation . . . 1245
10.30.6 Output & Results . . . 1245
10.30.7 Interpretation . . . 1245
10.30.8 Practical Tips . . . 1246
10.30.9 R Packages Used . . . 1246
10.30.10 For Reviewers . . . 1246
10.30.11 See also — labs in this chapter . . . 1246
10.31 R-hat and ESS Diagnostics . . . 1247
10.31.1 Introduction . . . 1247
10.31.2 Prerequisites . . . 1247
10.31.3 Theory . . . 1247
10.31.4 Assumptions . . . 1247
10.31.5 R Implementation . . . 1247
10.31.6 Output & Results . . . 1248
10.31.7 Interpretation . . . 1248
10.31.8 Practical Tips . . . 1248
10.31.9 R Packages Used . . . 1248
10.31.10 For Reviewers . . . 1249
10.31.11 See also — labs in this chapter . . . 1249
10.32 rstanarm . . . 1249
10.32.1 Introduction . . . 1249
10.32.2 Prerequisites . . . 1249
10.32.3 Theory . . . 1249
10.32.4 Assumptions . . . 1250
10.32.5 R Implementation . . . 1250
10.32.6 Output & Results . . . 1250
10.32.7 Interpretation . . . 1250
10.32.8 Practical Tips . . . 1250
10.32.9 R Packages Used . . . 1251
10.32.10 For Reviewers . . . 1251
10.32.11 See also — labs in this chapter . . . 1251
10.33 The Savage-Dickey Density Ratio . . . 1251

10.33.1 Introduction . . . 1251
10.33.2 Prerequisites . . . 1251
10.33.3 Theory . . . 1252
10.33.4 Assumptions . . . 1252
10.33.5 R Implementation . . . 1252
10.33.6 Output & Results . . . 1252
10.33.7 Interpretation . . . 1253
10.33.8 Practical Tips . . . 1253
10.33.9 R Packages Used . . . 1253
10.33.10 For Reviewers . . . 1253
10.33.11 See also — labs in this chapter . . . 1253
10.34 Stan: Introduction . . . 1254
10.34.1 Introduction . . . 1254
10.34.2 Prerequisites . . . 1254
10.34.3 Theory . . . 1254
10.34.4 Assumptions . . . 1254
10.34.5 R Implementation . . . 1255
10.34.6 Output & Results . . . 1255
10.34.7 Interpretation . . . 1255
10.34.8 Practical Tips . . . 1256
10.34.9 R Packages Used . . . 1256
10.34.10 For Reviewers . . . 1256
10.34.11 See also — labs in this chapter . . . 1256
10.35 tidybayes Workflow . . . 1256
10.35.1 Introduction . . . 1256
10.35.2 Prerequisites . . . 1257
10.35.3 Theory . . . 1257
10.35.4 Assumptions . . . 1257
10.35.5 R Implementation . . . 1257
10.35.6 Output & Results . . . 1258
10.35.7 Interpretation . . . 1258
10.35.8 Practical Tips . . . 1258
10.35.9 R Packages Used . . . 1259
10.35.10 For Reviewers . . . 1259
10.35.11 See also — labs in this chapter . . . 1259
10.36 WAIC . . . 1259
10.36.1 Introduction . . . 1259
10.36.2 Prerequisites . . . 1259
10.36.3 Theory . . . 1259
10.36.4 Assumptions . . . 1260
10.36.5 R Implementation . . . 1260
10.36.6 Output & Results . . . 1260
10.36.7 Interpretation . . . 1261
10.36.8 Practical Tips . . . 1261
10.36.9 R Packages Used . . . 1261
10.36.10 For Reviewers . . . 1261

10.36.11 See also — labs in this chapter 1261
10.37 Weakly Informative Priors 1262
10.37.1 Introduction 1262
10.37.2 Prerequisites 1262
10.37.3 Theory 1262
10.37.4 Assumptions 1262
10.37.5 R Implementation 1262
10.37.6 Output & Results 1263
10.37.7 Interpretation 1263
10.37.8 Practical Tips 1263
10.37.9 R Packages Used 1264
10.37.10 For Reviewers 1264
10.37.11 See also — labs in this chapter 1264
10.38 Week 3, Session 1 — Bayesian thinking; α control vs decision error 1264
10.38.1 Learning objectives 1264
10.38.2 Prerequisites 1264
10.38.3 Background 1264
10.38.4 Setup 1265
10.38.5 1. Hypothesis 1265
10.38.6 2. Visualise 1265
10.38.7 3. Assumptions 1266
10.38.8 4. Conduct 1266
10.38.9 5. Conclude 1266
10.38.10 Common pitfalls 1267
10.38.11 Further reading 1267
10.38.12 Session info 1267
10.38.13 See also — chapter index 1267
10.39 Week 3, Session 2 — brms / Stan, LOO, hierarchical models 1267
10.39.1 Learning objectives 1267
10.39.2 Prerequisites 1267
10.39.3 Background 1267
10.39.4 Setup 1268
10.39.5 1. Goal 1268
10.39.6 2. Approach 1268
10.39.7 3. Execution 1268
10.39.8 4. Check 1269
10.39.9 5. Report 1269
10.39.10 Common pitfalls 1270
10.39.11 Further reading 1270
10.39.12 Session info 1270
10.39.13 See also — chapter index 1270

11 Survival Analysis **1271**
11.1 Method pages 1271
11.2 Labs 1272

11.3 Accelerated Failure Time Models 1272
11.3.1 Introduction 1272
11.3.2 Prerequisites 1273
11.3.3 Theory 1273
11.3.4 Assumptions 1273
11.3.5 R Implementation 1273
11.3.6 Output & Results 1274
11.3.7 Interpretation 1274
11.3.8 Practical Tips 1274
11.3.9 R Packages Used 1275
11.3.10 For Reviewers 1275
11.3.11 See also — labs in this chapter 1275
11.4 The Brier Score for Survival 1275
11.4.1 Introduction 1275
11.4.2 Prerequisites 1275
11.4.3 Theory 1275
11.4.4 Assumptions 1276
11.4.5 R Implementation 1276
11.4.6 Output & Results 1276
11.4.7 Interpretation 1276
11.4.8 Practical Tips 1277
11.4.9 R Packages Used 1277
11.4.10 For Reviewers 1277
11.4.11 See also — labs in this chapter 1277
11.5 Censoring Types 1278
11.5.1 Introduction 1278
11.5.2 Prerequisites 1278
11.5.3 Theory 1278
11.5.4 Assumptions 1278
11.5.5 R Implementation 1279
11.5.6 Output & Results 1279
11.5.7 Interpretation 1279
11.5.8 Practical Tips 1279
11.5.9 R Packages Used 1280
11.5.10 For Reviewers 1280
11.5.11 See also — labs in this chapter 1280
11.6 Competing Risks: Cumulative Incidence 1280
11.6.1 Introduction 1280
11.6.2 Prerequisites 1280
11.6.3 Theory 1280
11.6.4 Assumptions 1281
11.6.5 R Implementation 1281
11.6.6 Output & Results 1281
11.6.7 Interpretation 1282
11.6.8 Practical Tips 1282
11.6.9 R Packages Used 1282

11.6.10 For Reviewers . . . 1282
11.6.11 See also — labs in this chapter . . . 1282
11.7 Concordance and the C-Index . . . 1283
11.7.1 Introduction . . . 1283
11.7.2 Prerequisites . . . 1283
11.7.3 Theory . . . 1283
11.7.4 Assumptions . . . 1283
11.7.5 R Implementation . . . 1284
11.7.6 Output & Results . . . 1284
11.7.7 Interpretation . . . 1284
11.7.8 Practical Tips . . . 1284
11.7.9 R Packages Used . . . 1285
11.7.10 For Reviewers . . . 1285
11.7.11 See also — labs in this chapter . . . 1285
11.8 Conditional Survival . . . 1285
11.8.1 Introduction . . . 1285
11.8.2 Prerequisites . . . 1285
11.8.3 Theory . . . 1285
11.8.4 Assumptions . . . 1286
11.8.5 R Implementation . . . 1286
11.8.6 Output & Results . . . 1286
11.8.7 Interpretation . . . 1286
11.8.8 Practical Tips . . . 1286
11.8.9 R Packages Used . . . 1287
11.8.10 For Reviewers . . . 1287
11.8.11 See also — labs in this chapter . . . 1287
11.9 Checking Cox Assumptions . . . 1287
11.9.1 Introduction . . . 1287
11.9.2 Prerequisites . . . 1288
11.9.3 Theory . . . 1288
11.9.4 Assumptions . . . 1288
11.9.5 R Implementation . . . 1288
11.9.6 Output & Results . . . 1289
11.9.7 Interpretation . . . 1289
11.9.8 Practical Tips . . . 1289
11.9.9 R Packages Used . . . 1290
11.9.10 For Reviewers . . . 1290
11.9.11 See also — labs in this chapter . . . 1290
11.10 Cox Proportional Hazards Regression . . . 1290
11.10.1 Introduction . . . 1290
11.10.2 Prerequisites . . . 1290
11.10.3 Theory . . . 1290
11.10.4 Assumptions . . . 1291
11.10.5 R Implementation . . . 1291
11.10.6 Output & Results . . . 1291
11.10.7 Interpretation . . . 1291

11.10.8 Practical Tips 1292
11.10.9 R Packages Used 1292
11.10.10 For Reviewers 1292
11.10.11 See also — labs in this chapter 1292
11.11 Fine-Gray Subdistribution Hazards 1293
11.11.1 Introduction 1293
11.11.2 Prerequisites 1293
11.11.3 Theory 1293
11.11.4 Assumptions 1293
11.11.5 R Implementation 1293
11.11.6 Output & Results 1294
11.11.7 Interpretation 1294
11.11.8 Practical Tips 1294
11.11.9 R Packages Used 1295
11.11.10 For Reviewers 1295
11.11.11 See also — labs in this chapter 1295
11.12 Royston-Parmar Flexible Parametric Models 1295
11.12.1 Introduction 1295
11.12.2 Prerequisites 1295
11.12.3 Theory 1295
11.12.4 Assumptions 1296
11.12.5 R Implementation 1296
11.12.6 Output & Results 1296
11.12.7 Interpretation 1297
11.12.8 Practical Tips 1297
11.12.9 R Packages Used 1297
11.12.10 For Reviewers 1297
11.12.11 See also — labs in this chapter 1298
11.13 Frailty Models 1298
11.13.1 Introduction 1298
11.13.2 Prerequisites 1298
11.13.3 Theory 1298
11.13.4 Assumptions 1298
11.13.5 R Implementation 1299
11.13.6 Output & Results 1299
11.13.7 Interpretation 1299
11.13.8 Practical Tips 1299
11.13.9 R Packages Used 1300
11.13.10 For Reviewers 1300
11.13.11 See also — labs in this chapter 1300
11.14 Hazard, Survival, and Cumulative Hazard 1300
11.14.1 Introduction 1300
11.14.2 Prerequisites 1300
11.14.3 Theory 1300
11.14.4 Assumptions 1301
11.14.5 R Implementation 1301

11.14.6 Output & Results . . . 1301
11.14.7 Interpretation . . . 1301
11.14.8 Practical Tips . . . 1302
11.14.9 R Packages Used . . . 1302
11.14.10 For Reviewers . . . 1302
11.14.11 See also — labs in this chapter . . . 1302
11.15 Interval-Censored Data . . . 1303
11.15.1 Introduction . . . 1303
11.15.2 Prerequisites . . . 1303
11.15.3 Theory . . . 1303
11.15.4 Assumptions . . . 1303
11.15.5 R Implementation . . . 1303
11.15.6 Output & Results . . . 1304
11.15.7 Interpretation . . . 1304
11.15.8 Practical Tips . . . 1304
11.15.9 R Packages Used . . . 1305
11.15.10 For Reviewers . . . 1305
11.15.11 See also — labs in this chapter . . . 1305
11.16 Joint Longitudinal-Survival Models . . . 1305
11.16.1 Introduction . . . 1305
11.16.2 Prerequisites . . . 1305
11.16.3 Theory . . . 1305
11.16.4 Assumptions . . . 1306
11.16.5 R Implementation . . . 1306
11.16.6 Output & Results . . . 1306
11.16.7 Interpretation . . . 1306
11.16.8 Practical Tips . . . 1307
11.16.9 R Packages Used . . . 1307
11.16.10 For Reviewers . . . 1307
11.16.11 See also — labs in this chapter . . . 1307
11.17 Kaplan-Meier Estimation . . . 1308
11.17.1 Introduction . . . 1308
11.17.2 Prerequisites . . . 1308
11.17.3 Theory . . . 1308
11.17.4 Assumptions . . . 1309
11.17.5 R Implementation . . . 1309
11.17.6 Output & Results . . . 1309
11.17.7 Interpretation . . . 1310
11.17.8 Practical Tips . . . 1310
11.17.9 For Reviewers . . . 1310
11.17.10 See also — labs in this chapter . . . 1310
11.18 Landmark Analysis . . . 1310
11.18.1 Introduction . . . 1310
11.18.2 Prerequisites . . . 1311
11.18.3 Theory . . . 1311
11.18.4 Assumptions . . . 1311

11.18.5 R Implementation 1311
11.18.6 Output & Results 1312
11.18.7 Interpretation 1312
11.18.8 Practical Tips 1312
11.18.9 R Packages Used 1313
11.18.10 For Reviewers 1313
11.18.11 See also — labs in this chapter 1313
11.19 Left Truncation 1313
11.19.1 Introduction 1313
11.19.2 Prerequisites 1313
11.19.3 Theory 1314
11.19.4 Assumptions 1314
11.19.5 R Implementation 1314
11.19.6 Output & Results 1314
11.19.7 Interpretation 1315
11.19.8 Practical Tips 1315
11.19.9 R Packages Used 1315
11.19.10 For Reviewers 1315
11.19.11 See also — labs in this chapter 1315
11.20 Log-Rank Test: Details 1316
11.20.1 Introduction 1316
11.20.2 Prerequisites 1316
11.20.3 Theory 1316
11.20.4 Assumptions 1316
11.20.5 R Implementation 1316
11.20.6 Output & Results 1317
11.20.7 Interpretation 1317
11.20.8 Practical Tips 1317
11.20.9 R Packages Used 1317
11.20.10 For Reviewers 1318
11.20.11 See also — labs in this chapter 1318
11.21 Multi-State Models 1318
11.21.1 Introduction 1318
11.21.2 Prerequisites 1318
11.21.3 Theory 1318
11.21.4 Assumptions 1319
11.21.5 R Implementation 1319
11.21.6 Output & Results 1319
11.21.7 Interpretation 1319
11.21.8 Practical Tips 1320
11.21.9 R Packages Used 1320
11.21.10 For Reviewers 1320
11.21.11 See also — labs in this chapter 1320
11.22 The Nelson-Aalen Estimator 1321
11.22.1 Introduction 1321
11.22.2 Prerequisites 1321

11.22.3 Theory 1321
11.22.4 Assumptions 1321
11.22.5 R Implementation 1322
11.22.6 Output & Results 1322
11.22.7 Interpretation 1322
11.22.8 Practical Tips 1322
11.22.9 R Packages Used 1323
11.22.10 For Reviewers 1323
11.22.11 See also — labs in this chapter 1323
11.23 Parametric Survival: Log-Normal 1323
11.23.1 Introduction 1323
11.23.2 Prerequisites 1323
11.23.3 Theory 1323
11.23.4 Assumptions 1324
11.23.5 R Implementation 1324
11.23.6 Output & Results 1324
11.23.7 Interpretation 1324
11.23.8 Practical Tips 1325
11.23.9 R Packages Used 1325
11.23.10 For Reviewers 1325
11.23.11 See also — labs in this chapter 1325
11.24 Parametric Survival: Weibull 1326
11.24.1 Introduction 1326
11.24.2 Prerequisites 1326
11.24.3 Theory 1326
11.24.4 Assumptions 1326
11.24.5 R Implementation 1326
11.24.6 Output & Results 1327
11.24.7 Interpretation 1327
11.24.8 Practical Tips 1327
11.24.9 R Packages Used 1328
11.24.10 For Reviewers 1328
11.24.11 See also — labs in this chapter 1328
11.25 Recurrent Events 1328
11.25.1 Introduction 1328
11.25.2 Prerequisites 1328
11.25.3 Theory 1328
11.25.4 Assumptions 1329
11.25.5 R Implementation 1329
11.25.6 Output & Results 1329
11.25.7 Interpretation 1330
11.25.8 Practical Tips 1330
11.25.9 R Packages Used 1330
11.25.10 For Reviewers 1330
11.25.11 See also — labs in this chapter 1331
11.26 Restricted Mean Survival Time (RMST) 1331

11.26.1 Introduction 1331
11.26.2 Prerequisites 1331
11.26.3 Theory 1331
11.26.4 Assumptions 1331
11.26.5 R Implementation 1332
11.26.6 Output & Results 1332
11.26.7 Interpretation 1332
11.26.8 Practical Tips 1332
11.26.9 R Packages Used 1333
11.26.10 For Reviewers 1333
11.26.11 See also — labs in this chapter 1333
11.27 Schoenfeld Residuals 1333
11.27.1 Introduction 1333
11.27.2 Prerequisites 1333
11.27.3 Theory 1333
11.27.4 Assumptions 1334
11.27.5 R Implementation 1334
11.27.6 Output & Results 1334
11.27.7 Interpretation 1334
11.27.8 Practical Tips 1335
11.27.9 R Packages Used 1335
11.27.10 For Reviewers 1335
11.27.11 See also — labs in this chapter 1335
11.28 Stratified Cox Models 1336
11.28.1 Introduction 1336
11.28.2 Prerequisites 1336
11.28.3 Theory 1336
11.28.4 Assumptions 1336
11.28.5 R Implementation 1336
11.28.6 Output & Results 1337
11.28.7 Interpretation 1337
11.28.8 Practical Tips 1337
11.28.9 R Packages Used 1337
11.28.10 For Reviewers 1338
11.28.11 See also — labs in this chapter 1338
11.29 Simulating Survival Data 1338
11.29.1 Introduction 1338
11.29.2 Prerequisites 1338
11.29.3 Theory 1338
11.29.4 Assumptions 1339
11.29.5 R Implementation 1339
11.29.6 Output & Results 1339
11.29.7 Interpretation 1339
11.29.8 Practical Tips 1340
11.29.9 R Packages Used 1340
11.29.10 For Reviewers 1340

11.29.11 See also — labs in this chapter 1340
11.30 Time-Dependent ROC 1341
11.30.1 Introduction 1341
11.30.2 Prerequisites 1341
11.30.3 Theory 1341
11.30.4 Assumptions 1341
11.30.5 R Implementation 1341
11.30.6 Output & Results 1342
11.30.7 Interpretation 1342
11.30.8 Practical Tips 1342
11.30.9 R Packages Used 1343
11.30.10 For Reviewers 1343
11.30.11 See also — labs in this chapter 1343
11.31 Time-Varying Covariates 1343
11.31.1 Introduction 1343
11.31.2 Prerequisites 1343
11.31.3 Theory 1343
11.31.4 Assumptions 1344
11.31.5 R Implementation 1344
11.31.6 Output & Results 1344
11.31.7 Interpretation 1344
11.31.8 Practical Tips 1345
11.31.9 R Packages Used 1345
11.31.10 For Reviewers 1345
11.31.11 See also — labs in this chapter 1345
11.32 Weighted Log-Rank Tests 1346
11.32.1 Introduction 1346
11.32.2 Prerequisites 1346
11.32.3 Theory 1346
11.32.4 Assumptions 1346
11.32.5 R Implementation 1347
11.32.6 Output & Results 1347
11.32.7 Interpretation 1347
11.32.8 Practical Tips 1347
11.32.9 R Packages Used 1348
11.32.10 For Reviewers 1348
11.32.11 See also — labs in this chapter 1348
11.33 Week 3, Session 2 — Competing risks and multistate models 1348
11.33.1 Learning objectives 1348
11.33.2 Prerequisites 1348
11.33.3 Background 1348
11.33.4 Setup 1349
11.33.5 1. Hypothesis 1349
11.33.6 2. Visualise 1349
11.33.7 3. Assumptions 1350
11.33.8 4. Conduct 1350

11.33.9 5. Concluding statement . . . 1351
11.33.10 Common pitfalls . . . 1351
11.33.11 Further reading . . . 1351
11.33.12 Session info . . . 1351
11.33.13 See also — chapter index . . . 1351
11.34 Week 3, Session 4 — Survival ML (random survival forest, DeepSurv) . . . 1352
11.34.1 Learning objectives . . . 1352
11.34.2 Prerequisites . . . 1352
11.34.3 Background . . . 1352
11.34.4 Setup . . . 1352
11.34.5 1. Goal . . . 1353
11.34.6 2. Approach . . . 1353
11.34.7 3. Execution . . . 1353
11.34.8 4. Check . . . 1353
11.34.9 5. Report . . . 1354
11.34.10 Common pitfalls . . . 1354
11.34.11 Further reading . . . 1354
11.34.12 Session info . . . 1355
11.34.13 See also — chapter index . . . 1355
11.35 Week 4, Session 3 — Survival primer: KM, log-rank, Cox PH 1355
11.35.1 Learning objectives . . . 1355
11.35.2 Prerequisites . . . 1355
11.35.3 Background . . . 1355
11.35.4 Setup . . . 1356
11.35.5 1. Hypothesis . . . 1356
11.35.6 2. Visualise . . . 1356
11.35.7 3. Assumptions . . . 1356
11.35.8 4. Conduct . . . 1356
11.35.9 5. Concluding statement . . . 1357
11.35.10 Common pitfalls . . . 1357
11.35.11 Further reading . . . 1357
11.35.12 Session info . . . 1357
11.35.13 See also — chapter index . . . 1357
11.36 Week 3, Session 5 — Time-dependent Brier, IPA, external validation . . . 1357
11.36.1 Learning objectives . . . 1358
11.36.2 Prerequisites . . . 1358
11.36.3 Background . . . 1358
11.36.4 Setup . . . 1358
11.36.5 1. Hypothesis . . . 1358
11.36.6 2. Visualise . . . 1359
11.36.7 3. Assumptions . . . 1359
11.36.8 4. Conduct . . . 1359
11.36.9 5. Concluding statement . . . 1360
11.36.10 Common pitfalls . . . 1360

11.36.11 Further reading . . . 1361
11.36.12 Session info . . . 1361
11.36.13 See also — chapter index . . . 1361
11.37 Week 3, Session 1 — Time-varying covariates and immortal-time bias . . . 1361
11.37.1 Learning objectives . . . 1361
11.37.2 Prerequisites . . . 1361
11.37.3 Background . . . 1361
11.37.4 Setup . . . 1362
11.37.5 1. Hypothesis . . . 1362
11.37.6 2. Visualise . . . 1362
11.37.7 3. Assumptions . . . 1363
11.37.8 4. Conduct . . . 1363
11.37.9 5. Concluding statement . . . 1364
11.37.10 Common pitfalls . . . 1364
11.37.11 Further reading . . . 1364
11.37.12 Session info . . . 1364
11.37.13 See also — chapter index . . . 1364

12 Bioinformatics **1365**
12.1 Method pages . . . 1365
12.2 Labs . . . 1366
12.3 Variant Annotation with VEP . . . 1367
12.3.1 Introduction . . . 1367
12.3.2 Prerequisites . . . 1367
12.3.3 Theory . . . 1367
12.3.4 Assumptions . . . 1367
12.3.5 R Implementation . . . 1367
12.3.6 Output & Results . . . 1368
12.3.7 Interpretation . . . 1368
12.3.8 Practical Tips . . . 1368
12.3.9 For Reviewers . . . 1368
12.3.10 See also — labs in this chapter . . . 1369
12.4 ATAC-Seq Analysis . . . 1369
12.4.1 Introduction . . . 1369
12.4.2 Prerequisites . . . 1369
12.4.3 Theory . . . 1369
12.4.4 Assumptions . . . 1369
12.4.5 R Implementation . . . 1370
12.4.6 Output & Results . . . 1370
12.4.7 Interpretation . . . 1370
12.4.8 Practical Tips . . . 1370
12.4.9 R Packages Used . . . 1371
12.4.10 For Reviewers . . . 1371
12.4.11 See also — labs in this chapter . . . 1371
12.5 Batch Correction with ComBat . . . 1371

12.5.1 Introduction . . . 1371
12.5.2 Prerequisites . . . 1372
12.5.3 Theory . . . 1372
12.5.4 Assumptions . . . 1372
12.5.5 R Implementation . . . 1372
12.5.6 Output & Results . . . 1373
12.5.7 Interpretation . . . 1373
12.5.8 Practical Tips . . . 1373
12.5.9 R Packages Used . . . 1373
12.5.10 For Reviewers . . . 1374
12.5.11 See also — labs in this chapter . . . 1374
12.6 Gene Annotation with biomaRt . . . 1374
12.6.1 Introduction . . . 1374
12.6.2 Prerequisites . . . 1374
12.6.3 Theory . . . 1374
12.6.4 Assumptions . . . 1375
12.6.5 R Implementation . . . 1375
12.6.6 Output & Results . . . 1375
12.6.7 Interpretation . . . 1376
12.6.8 Practical Tips . . . 1376
12.6.9 R Packages Used . . . 1376
12.6.10 For Reviewers . . . 1376
12.6.11 See also — labs in this chapter . . . 1376
12.7 Bulk RNA-seq Differential Expression with DESeq2 . . . 1377
12.7.1 Introduction . . . 1377
12.7.2 Prerequisites . . . 1377
12.7.3 Theory . . . 1377
12.7.4 Assumptions . . . 1378
12.7.5 R Implementation . . . 1378
12.7.6 Output & Results . . . 1379
12.7.7 Interpretation . . . 1379
12.7.8 Practical Tips . . . 1379
12.7.9 For Reviewers . . . 1379
12.7.10 See also — labs in this chapter . . . 1380
12.8 Alignment with BWA and Bowtie . . . 1380
12.8.1 Introduction . . . 1380
12.8.2 Prerequisites . . . 1380
12.8.3 Theory . . . 1380
12.8.4 Assumptions . . . 1380
12.8.5 R Implementation . . . 1381
12.8.6 Output & Results . . . 1381
12.8.7 Interpretation . . . 1381
12.8.8 Practical Tips . . . 1381
12.8.9 R Packages Used . . . 1382
12.8.10 For Reviewers . . . 1382
12.8.11 See also — labs in this chapter . . . 1382

12.9 ChIP-Seq Analysis 1382
12.9.1 Introduction 1382
12.9.2 Prerequisites 1382
12.9.3 Theory 1383
12.9.4 Assumptions 1383
12.9.5 R Implementation 1383
12.9.6 Output & Results 1384
12.9.7 Interpretation 1384
12.9.8 Practical Tips 1384
12.9.9 R Packages Used 1384
12.9.10 For Reviewers 1385
12.9.11 See also — labs in this chapter 1385
12.10 Copy Number Variation Analysis 1385
12.10.1 Introduction 1385
12.10.2 Prerequisites 1385
12.10.3 Theory 1385
12.10.4 Assumptions 1385
12.10.5 R Implementation 1385
12.10.6 Output & Results 1386
12.10.7 Interpretation 1386
12.10.8 Practical Tips 1386
12.10.9 For Reviewers 1387
12.10.10 See also — labs in this chapter 1387
12.11 Counting Reads with featureCounts 1387
12.11.1 Introduction 1387
12.11.2 Prerequisites 1387
12.11.3 Theory 1387
12.11.4 Assumptions 1388
12.11.5 R Implementation 1388
12.11.6 Output & Results 1388
12.11.7 Interpretation 1388
12.11.8 Practical Tips 1389
12.11.9 R Packages Used 1389
12.11.10 For Reviewers 1389
12.11.11 See also — labs in this chapter 1389
12.12 Drug-Target Interaction Mining 1389
12.12.1 Introduction 1389
12.12.2 Prerequisites 1390
12.12.3 Theory 1390
12.12.4 Assumptions 1390
12.12.5 R Implementation 1390
12.12.6 Output & Results 1391
12.12.7 Interpretation 1391
12.12.8 Practical Tips 1391
12.12.9 Reporting 1391
12.12.10 For Reviewers 1392

12.12.11 See also — labs in this chapter 1392
12.13 Differential Expression with edgeR 1392
12.13.1 Introduction 1392
12.13.2 Prerequisites 1392
12.13.3 Theory 1392
12.13.4 Assumptions 1393
12.13.5 R Implementation 1393
12.13.6 Output & Results 1393
12.13.7 Interpretation 1393
12.13.8 Practical Tips 1394
12.13.9 R Packages Used 1394
12.13.10 For Reviewers 1394
12.13.11 See also — labs in this chapter 1394
12.14 FASTQ Quality Control 1395
12.14.1 Introduction 1395
12.14.2 Prerequisites 1395
12.14.3 Theory 1395
12.14.4 Assumptions 1395
12.14.5 R Implementation 1395
12.14.6 Output & Results 1396
12.14.7 Interpretation 1396
12.14.8 Practical Tips 1396
12.14.9 R Packages Used 1397
12.14.10 For Reviewers 1397
12.14.11 See also — labs in this chapter 1397
12.15 Gene Ontology Enrichment 1397
12.15.1 Introduction 1397
12.15.2 Prerequisites 1397
12.15.3 Theory 1397
12.15.4 Assumptions 1398
12.15.5 R Implementation 1398
12.15.6 Output & Results 1398
12.15.7 Interpretation 1399
12.15.8 Practical Tips 1399
12.15.9 R Packages Used 1399
12.15.10 For Reviewers 1399
12.15.11 See also — labs in this chapter 1399
12.16 GSEA Preranked Analysis 1400
12.16.1 Introduction 1400
12.16.2 Prerequisites 1400
12.16.3 Theory 1400
12.16.4 Assumptions 1400
12.16.5 R Implementation 1400
12.16.6 Output & Results 1401
12.16.7 Interpretation 1401
12.16.8 Practical Tips 1401

12.16.9 For Reviewers . . . 1401
12.16.10 See also — labs in this chapter . . . 1401
12.17 GSVA Single-Sample Enrichment . . . 1402
12.17.1 Introduction . . . 1402
12.17.2 Prerequisites . . . 1402
12.17.3 Theory . . . 1402
12.17.4 Assumptions . . . 1402
12.17.5 R Implementation . . . 1402
12.17.6 Output & Results . . . 1403
12.17.7 Interpretation . . . 1403
12.17.8 Practical Tips . . . 1403
12.17.9 R Packages Used . . . 1404
12.17.10 For Reviewers . . . 1404
12.17.11 See also — labs in this chapter . . . 1404
12.18 Heatmaps for RNA-seq . . . 1404
12.18.1 Introduction . . . 1404
12.18.2 Prerequisites . . . 1404
12.18.3 Theory . . . 1404
12.18.4 Assumptions . . . 1405
12.18.5 R Implementation . . . 1405
12.18.6 Output & Results . . . 1405
12.18.7 Interpretation . . . 1406
12.18.8 Practical Tips . . . 1406
12.18.9 R Packages Used . . . 1406
12.18.10 For Reviewers . . . 1406
12.18.11 See also — labs in this chapter . . . 1407
12.19 Differential Expression with limma-voom . . . 1407
12.19.1 Introduction . . . 1407
12.19.2 Prerequisites . . . 1407
12.19.3 Theory . . . 1407
12.19.4 Assumptions . . . 1407
12.19.5 R Implementation . . . 1408
12.19.6 Output & Results . . . 1408
12.19.7 Interpretation . . . 1408
12.19.8 Practical Tips . . . 1408
12.19.9 R Packages Used . . . 1409
12.19.10 For Reviewers . . . 1409
12.19.11 See also — labs in this chapter . . . 1409
12.20 MA Plots . . . 1409
12.20.1 Introduction . . . 1409
12.20.2 Prerequisites . . . 1410
12.20.3 Theory . . . 1410
12.20.4 Assumptions . . . 1410
12.20.5 R Implementation . . . 1410
12.20.6 Output & Results . . . 1410
12.20.7 Interpretation . . . 1411

12.20.8 Practical Tips 1411
12.20.9 R Packages Used 1411
12.20.10 For Reviewers 1411
12.20.11 See also — labs in this chapter 1412
12.21 Metagenomic Profiling 1412
12.21.1 Introduction 1412
12.21.2 Prerequisites 1412
12.21.3 Theory 1412
12.21.4 Assumptions 1412
12.21.5 R Implementation 1412
12.21.6 Output & Results 1413
12.21.7 Interpretation 1413
12.21.8 Practical Tips 1413
12.21.9 For Reviewers 1414
12.21.10 See also — labs in this chapter 1414
12.22 DNA Methylation Analysis 1414
12.22.1 Introduction 1414
12.22.2 Prerequisites 1414
12.22.3 Theory 1414
12.22.4 Assumptions 1415
12.22.5 R Implementation 1415
12.22.6 Output & Results 1415
12.22.7 Interpretation 1415
12.22.8 Practical Tips 1416
12.22.9 R Packages Used 1416
12.22.10 For Reviewers 1416
12.22.11 See also — labs in this chapter 1416
12.23 Microbiome Analysis with DADA2 1417
12.23.1 Introduction 1417
12.23.2 Prerequisites 1417
12.23.3 Theory 1417
12.23.4 Assumptions 1417
12.23.5 R Implementation 1417
12.23.6 Output & Results 1418
12.23.7 Interpretation 1418
12.23.8 Practical Tips 1419
12.23.9 R Packages Used 1419
12.23.10 For Reviewers 1419
12.23.11 See also — labs in this chapter 1419
12.24 Microbiome Diversity Metrics 1419
12.24.1 Introduction 1419
12.24.2 Prerequisites 1420
12.24.3 Theory 1420
12.24.4 Assumptions 1420
12.24.5 R Implementation 1420
12.24.6 Output & Results 1421

12.24.7 Interpretation 1421
12.24.8 Practical Tips 1421
12.24.9 For Reviewers 1421
12.24.10 See also — labs in this chapter 1421
12.25 Multiple Sequence Alignment 1421
12.25.1 Introduction 1421
12.25.2 Prerequisites 1422
12.25.3 Theory 1422
12.25.4 Assumptions 1422
12.25.5 R Implementation 1422
12.25.6 Output & Results 1422
12.25.7 Interpretation 1423
12.25.8 Practical Tips 1423
12.25.9 For Reviewers 1423
12.25.10 See also — labs in this chapter 1423
12.26 Multi-Omics Integration 1423
12.26.1 Introduction 1423
12.26.2 Prerequisites 1423
12.26.3 Theory 1424
12.26.4 Assumptions 1424
12.26.5 R Implementation 1424
12.26.6 Output & Results 1425
12.26.7 Interpretation 1425
12.26.8 Practical Tips 1425
12.26.9 For Reviewers 1425
12.26.10 See also — labs in this chapter 1425
12.27 KEGG Pathway Enrichment 1425
12.27.1 Introduction 1425
12.27.2 Prerequisites 1426
12.27.3 Theory 1426
12.27.4 Assumptions 1426
12.27.5 R Implementation 1426
12.27.6 Output & Results 1427
12.27.7 Interpretation 1427
12.27.8 Practical Tips 1427
12.27.9 R Packages Used 1428
12.27.10 For Reviewers 1428
12.27.11 See also — labs in this chapter 1428
12.28 PCA of RNA-seq Samples 1428
12.28.1 Introduction 1428
12.28.2 Prerequisites 1428
12.28.3 Theory 1428
12.28.4 Assumptions 1429
12.28.5 R Implementation 1429
12.28.6 Output & Results 1429
12.28.7 Interpretation 1429

12.28.8 Practical Tips 1430
12.28.9 R Packages Used 1430
12.28.10 For Reviewers 1430
12.28.11 See also — labs in this chapter 1430
12.29 Phylogenetic Trees with ape 1430
12.29.1 Introduction 1430
12.29.2 Prerequisites 1431
12.29.3 Theory 1431
12.29.4 Assumptions 1431
12.29.5 R Implementation 1431
12.29.6 Output & Results 1432
12.29.7 Interpretation 1432
12.29.8 Practical Tips 1432
12.29.9 For Reviewers 1432
12.29.10 See also — labs in this chapter 1432
12.30 Population Genetics Basics 1433
12.30.1 Introduction 1433
12.30.2 Prerequisites 1433
12.30.3 Theory 1433
12.30.4 Assumptions 1433
12.30.5 R Implementation 1433
12.30.6 Output & Results 1434
12.30.7 Interpretation 1434
12.30.8 Practical Tips 1434
12.30.9 For Reviewers 1434
12.30.10 See also — labs in this chapter 1434
12.31 Protein Structure Prediction 1435
12.31.1 Introduction 1435
12.31.2 Prerequisites 1435
12.31.3 Theory 1435
12.31.4 Assumptions 1435
12.31.5 R Implementation 1435
12.31.6 Output & Results 1436
12.31.7 Interpretation 1436
12.31.8 Practical Tips 1436
12.31.9 For Reviewers 1436
12.31.10 See also — labs in this chapter 1436
12.32 Proteomics with MSstats 1437
12.32.1 Introduction 1437
12.32.2 Prerequisites 1437
12.32.3 Theory 1437
12.32.4 Assumptions 1437
12.32.5 R Implementation 1437
12.32.6 Output & Results 1438
12.32.7 Interpretation 1438
12.32.8 Practical Tips 1439

12.32.9 For Reviewers . . . 1439
12.32.10 See also — labs in this chapter . . . 1439
12.33 Read Trimming and Adapter Removal . . . 1439
12.33.1 Introduction . . . 1439
12.33.2 Prerequisites . . . 1439
12.33.3 Theory . . . 1440
12.33.4 Assumptions . . . 1440
12.33.5 R Implementation . . . 1440
12.33.6 Output & Results . . . 1440
12.33.7 Interpretation . . . 1440
12.33.8 Practical Tips . . . 1441
12.33.9 R Packages Used . . . 1441
12.33.10 For Reviewers . . . 1441
12.33.11 See also — labs in this chapter . . . 1441
12.34 RNA-seq Normalisation . . . 1442
12.34.1 Introduction . . . 1442
12.34.2 Prerequisites . . . 1442
12.34.3 Theory . . . 1442
12.34.4 Assumptions . . . 1442
12.34.5 R Implementation . . . 1443
12.34.6 Output & Results . . . 1443
12.34.7 Interpretation . . . 1443
12.34.8 Practical Tips . . . 1443
12.34.9 R Packages Used . . . 1444
12.34.10 For Reviewers . . . 1444
12.34.11 See also — labs in this chapter . . . 1444
12.35 Pseudoalignment with Salmon and Kallisto . . . 1444
12.35.1 Introduction . . . 1444
12.35.2 Prerequisites . . . 1445
12.35.3 Theory . . . 1445
12.35.4 Assumptions . . . 1445
12.35.5 R Implementation . . . 1445
12.35.6 Output & Results . . . 1446
12.35.7 Interpretation . . . 1446
12.35.8 Practical Tips . . . 1446
12.35.9 R Packages Used . . . 1446
12.35.10 For Reviewers . . . 1446
12.35.11 See also — labs in this chapter . . . 1447
12.36 Cell-Type Annotation . . . 1447
12.36.1 Introduction . . . 1447
12.36.2 Prerequisites . . . 1447
12.36.3 Theory . . . 1447
12.36.4 Assumptions . . . 1448
12.36.5 R Implementation . . . 1448
12.36.6 Output & Results . . . 1448
12.36.7 Interpretation . . . 1448

12.36.8 Practical Tips 1448
12.36.9 R Packages Used 1449
12.36.10 For Reviewers 1449
12.36.11 See also — labs in this chapter 1449
12.37 scRNA Clustering 1449
12.37.1 Introduction 1449
12.37.2 Prerequisites 1450
12.37.3 Theory 1450
12.37.4 Assumptions 1450
12.37.5 R Implementation 1450
12.37.6 Output & Results 1451
12.37.7 Interpretation 1451
12.37.8 Practical Tips 1451
12.37.9 R Packages Used 1452
12.37.10 For Reviewers 1452
12.37.11 See also — labs in this chapter 1452
12.38 Integration with Harmony 1452
12.38.1 Introduction 1452
12.38.2 Prerequisites 1452
12.38.3 Theory 1452
12.38.4 Assumptions 1453
12.38.5 R Implementation 1453
12.38.6 Output & Results 1453
12.38.7 Interpretation 1453
12.38.8 Practical Tips 1454
12.38.9 For Reviewers 1454
12.38.10 See also — labs in this chapter 1454
12.39 Finding Marker Genes 1454
12.39.1 Introduction 1454
12.39.2 Prerequisites 1454
12.39.3 Theory 1455
12.39.4 Assumptions 1455
12.39.5 R Implementation 1455
12.39.6 Output & Results 1456
12.39.7 Interpretation 1456
12.39.8 Practical Tips 1456
12.39.9 R Packages Used 1456
12.39.10 For Reviewers 1457
12.39.11 See also — labs in this chapter 1457
12.40 scRNA Normalisation 1457
12.40.1 Introduction 1457
12.40.2 Prerequisites 1457
12.40.3 Theory 1457
12.40.4 Assumptions 1458
12.40.5 R Implementation 1458
12.40.6 Output & Results 1458

12.40.7 Interpretation 1458
12.40.8 Practical Tips 1459
12.40.9 R Packages Used 1459
12.40.10 For Reviewers 1459
12.40.11 See also — labs in this chapter 1459
12.41 scRNA QC and Filtering 1459
12.41.1 Introduction 1459
12.41.2 Prerequisites 1460
12.41.3 Theory 1460
12.41.4 Assumptions 1460
12.41.5 R Implementation 1460
12.41.6 Output & Results 1461
12.41.7 Interpretation 1461
12.41.8 Practical Tips 1461
12.41.9 R Packages Used 1462
12.41.10 For Reviewers 1462
12.41.11 See also — labs in this chapter 1462
12.42 Trajectory Analysis 1462
12.42.1 Introduction 1462
12.42.2 Prerequisites 1462
12.42.3 Theory 1463
12.42.4 Assumptions 1463
12.42.5 R Implementation 1463
12.42.6 Output & Results 1464
12.42.7 Interpretation 1464
12.42.8 Practical Tips 1464
12.42.9 R Packages Used 1464
12.42.10 For Reviewers 1465
12.42.11 See also — labs in this chapter 1465
12.43 Sequence Alignment: Overview 1465
12.43.1 Introduction 1465
12.43.2 Prerequisites 1465
12.43.3 Theory 1465
12.43.4 Assumptions 1466
12.43.5 R Implementation 1466
12.43.6 Output & Results 1466
12.43.7 Interpretation 1467
12.43.8 Practical Tips 1467
12.43.9 R Packages Used 1467
12.43.10 For Reviewers 1467
12.43.11 See also — labs in this chapter 1467
12.44 Single-Cell with Seurat 1468
12.44.1 Introduction 1468
12.44.2 Prerequisites 1468
12.44.3 Theory 1468
12.44.4 Assumptions 1468

12.44.5 R Implementation 1469
12.44.6 Output & Results 1469
12.44.7 Interpretation 1469
12.44.8 Practical Tips 1469
12.44.9 R Packages Used 1470
12.44.10 For Reviewers 1470
12.44.11 See also — labs in this chapter 1470
12.45 Spatial Transcriptomics 1470
12.45.1 Introduction 1470
12.45.2 Prerequisites 1470
12.45.3 Theory 1471
12.45.4 Assumptions 1471
12.45.5 R Implementation 1471
12.45.6 Output & Results 1472
12.45.7 Interpretation 1472
12.45.8 Practical Tips 1472
12.45.9 For Reviewers 1472
12.45.10 See also — labs in this chapter 1472
12.46 STAR: Spliced Alignment 1472
12.46.1 Introduction 1472
12.46.2 Prerequisites 1473
12.46.3 Theory 1473
12.46.4 Assumptions 1473
12.46.5 R Implementation 1473
12.46.6 Output & Results 1474
12.46.7 Interpretation 1474
12.46.8 Practical Tips 1474
12.46.9 R Packages Used 1474
12.46.10 For Reviewers 1474
12.46.11 See also — labs in this chapter 1475
12.47 Structural Variant Detection 1475
12.47.1 Introduction 1475
12.47.2 Prerequisites 1475
12.47.3 Theory 1475
12.47.4 Assumptions 1475
12.47.5 R Implementation 1475
12.47.6 Output & Results 1476
12.47.7 Interpretation 1476
12.47.8 Practical Tips 1476
12.47.9 For Reviewers 1476
12.47.10 See also — labs in this chapter 1477
12.48 Surrogate Variable Analysis (SVA) 1477
12.48.1 Introduction 1477
12.48.2 Prerequisites 1477
12.48.3 Theory 1477
12.48.4 Assumptions 1477

12.48.5 R Implementation . . . 1477
12.48.6 Output & Results . . . 1478
12.48.7 Interpretation . . . 1478
12.48.8 Practical Tips . . . 1478
12.48.9 For Reviewers . . . 1478
12.48.10 See also — labs in this chapter . . . 1479
12.49 Transcript-to-Gene Summarisation . . . 1479
12.49.1 Introduction . . . 1479
12.49.2 Prerequisites . . . 1479
12.49.3 Theory . . . 1479
12.49.4 Assumptions . . . 1479
12.49.5 R Implementation . . . 1480
12.49.6 Output & Results . . . 1480
12.49.7 Interpretation . . . 1480
12.49.8 Practical Tips . . . 1480
12.49.9 R Packages Used . . . 1481
12.49.10 For Reviewers . . . 1481
12.49.11 See also — labs in this chapter . . . 1481
12.50 Variant Calling with GATK . . . 1481
12.50.1 Introduction . . . 1481
12.50.2 Prerequisites . . . 1481
12.50.3 Theory . . . 1481
12.50.4 Assumptions . . . 1482
12.50.5 R Implementation . . . 1482
12.50.6 Output & Results . . . 1482
12.50.7 Interpretation . . . 1482
12.50.8 Practical Tips . . . 1482
12.50.9 For Reviewers . . . 1483
12.50.10 See also — labs in this chapter . . . 1483
12.51 VCF Manipulation . . . 1483
12.51.1 Introduction . . . 1483
12.51.2 Prerequisites . . . 1483
12.51.3 Theory . . . 1483
12.51.4 Assumptions . . . 1484
12.51.5 R Implementation . . . 1484
12.51.6 Output & Results . . . 1484
12.51.7 Interpretation . . . 1485
12.51.8 Practical Tips . . . 1485
12.51.9 R Packages Used . . . 1485
12.51.10 For Reviewers . . . 1485
12.51.11 See also — labs in this chapter . . . 1485
12.52 Volcano Plots . . . 1486
12.52.1 Introduction . . . 1486
12.52.2 Prerequisites . . . 1486
12.52.3 Theory . . . 1486
12.52.4 Assumptions . . . 1486

12.52.5 R Implementation . . . 1486
12.52.6 Output & Results . . . 1487
12.52.7 Interpretation . . . 1487
12.52.8 Practical Tips . . . 1487
12.52.9 R Packages Used . . . 1488
12.52.10 For Reviewers . . . 1488
12.52.11 See also — labs in this chapter . . . 1488
12.53 Week 4, Session 2 — Enrichment analysis (GSEA / ORA) 1488
12.53.1 Learning objectives . . . 1488
12.53.2 Prerequisites . . . 1489
12.53.3 Background . . . 1489
12.53.4 Setup . . . 1489
12.53.5 1. Goal . . . 1489
12.53.6 2. Approach . . . 1489
12.53.7 3. Execution . . . 1490
12.53.8 4. Check . . . 1490
12.53.9 5. Report . . . 1491
12.53.10 Common pitfalls . . . 1491
12.53.11 Further reading . . . 1491
12.53.12 Session info . . . 1491
12.53.13 See also — chapter index . . . 1491
12.54 Week 4, Session 1 — RNA-seq with DESeq2 / edgeR . . . 1491
12.54.1 Learning objectives . . . 1492
12.54.2 Prerequisites . . . 1492
12.54.3 Background . . . 1492
12.54.4 Setup . . . 1492
12.54.5 1. Goal . . . 1492
12.54.6 2. Approach . . . 1492
12.54.7 3. Execution . . . 1493
12.54.8 4. Check . . . 1494
12.54.9 5. Report . . . 1494
12.54.10 Common pitfalls . . . 1494
12.54.11 Further reading . . . 1495
12.54.12 Session info . . . 1495
12.54.13 See also — chapter index . . . 1495
12.55 Week 4, Session 3 — scRNA-seq with Seurat . . . 1495
12.55.1 Learning objectives . . . 1495
12.55.2 Prerequisites . . . 1495
12.55.3 Background . . . 1495
12.55.4 Setup . . . 1496
12.55.5 1. Goal . . . 1496
12.55.6 2. Approach . . . 1496
12.55.7 3. Execution . . . 1496
12.55.8 4. Check . . . 1497
12.55.9 5. Report . . . 1497
12.55.10 Common pitfalls . . . 1498

12.55.11 Further reading . . . 1498
12.55.12 Session info . . . 1498
12.55.13 See also — chapter index . . . 1498

13 Machine Learning 1499
13.1 Method pages . . . 1499
13.2 Labs . . . 1500
13.3 Anomaly Detection . . . 1501
13.3.1 Introduction . . . 1501
13.3.2 Prerequisites . . . 1501
13.3.3 Theory . . . 1501
13.3.4 Assumptions . . . 1501
13.3.5 R Implementation . . . 1501
13.3.6 Output & Results . . . 1502
13.3.7 Interpretation . . . 1502
13.3.8 Practical Tips . . . 1502
13.3.9 For Reviewers . . . 1502
13.3.10 See also — labs in this chapter . . . 1502
13.4 Bagging . . . 1503
13.4.1 Introduction . . . 1503
13.4.2 Prerequisites . . . 1503
13.4.3 Theory . . . 1503
13.4.4 Assumptions . . . 1503
13.4.5 R Implementation . . . 1504
13.4.6 Output & Results . . . 1504
13.4.7 Interpretation . . . 1504
13.4.8 Practical Tips . . . 1504
13.4.9 R Packages Used . . . 1505
13.4.10 For Reviewers . . . 1505
13.4.11 See also — labs in this chapter . . . 1505
13.5 The Bias-Variance Tradeoff . . . 1505
13.5.1 Introduction . . . 1505
13.5.2 Prerequisites . . . 1505
13.5.3 Theory . . . 1505
13.5.4 Assumptions . . . 1506
13.5.5 R Implementation . . . 1506
13.5.6 Output & Results . . . 1506
13.5.7 Interpretation . . . 1506
13.5.8 Practical Tips . . . 1507
13.5.9 For Reviewers . . . 1507
13.5.10 See also — labs in this chapter . . . 1507
13.6 Calibration Plots . . . 1507
13.6.1 Introduction . . . 1507
13.6.2 Prerequisites . . . 1507
13.6.3 Theory . . . 1508
13.6.4 Assumptions . . . 1508

13.6.5 R Implementation . . . 1508
13.6.6 Output & Results . . . 1509
13.6.7 Interpretation . . . 1509
13.6.8 Practical Tips . . . 1509
13.6.9 For Reviewers . . . 1509
13.6.10 See also — labs in this chapter . . . 1509
13.7 CatBoost . . . 1510
13.7.1 Introduction . . . 1510
13.7.2 Prerequisites . . . 1510
13.7.3 Theory . . . 1510
13.7.4 Assumptions . . . 1510
13.7.5 R Implementation . . . 1510
13.7.6 Output & Results . . . 1511
13.7.7 Interpretation . . . 1511
13.7.8 Practical Tips . . . 1511
13.7.9 For Reviewers . . . 1511
13.7.10 See also — labs in this chapter . . . 1512
13.8 Class Weights for Imbalanced Data . . . 1512
13.8.1 Introduction . . . 1512
13.8.2 Prerequisites . . . 1512
13.8.3 Theory . . . 1512
13.8.4 Assumptions . . . 1513
13.8.5 R Implementation . . . 1513
13.8.6 Output & Results . . . 1513
13.8.7 Interpretation . . . 1514
13.8.8 Practical Tips . . . 1514
13.8.9 R Packages Used . . . 1514
13.8.10 For Reviewers . . . 1514
13.8.11 See also — labs in this chapter . . . 1514
13.9 DBSCAN as an ML Tool . . . 1515
13.9.1 Introduction . . . 1515
13.9.2 Prerequisites . . . 1515
13.9.3 Theory . . . 1515
13.9.4 Assumptions . . . 1515
13.9.5 R Implementation . . . 1516
13.9.6 Output & Results . . . 1516
13.9.7 Interpretation . . . 1516
13.9.8 Practical Tips . . . 1516
13.9.9 R Packages Used . . . 1517
13.9.10 For Reviewers . . . 1517
13.9.11 See also — labs in this chapter . . . 1517
13.10 k-Means as an ML Tool . . . 1517
13.10.1 Introduction . . . 1517
13.10.2 Prerequisites . . . 1518
13.10.3 Theory . . . 1518
13.10.4 Assumptions . . . 1518

13.10.5 R Implementation . . . 1518
13.10.6 Output & Results . . . 1519
13.10.7 Interpretation . . . 1519
13.10.8 Practical Tips . . . 1519
13.10.9 R Packages Used . . . 1519
13.10.10 For Reviewers . . . 1520
13.10.11 See also — labs in this chapter . . . 1520
13.11 Convolutional Neural Networks: Introduction . . . 1520
13.11.1 Introduction . . . 1520
13.11.2 Prerequisites . . . 1520
13.11.3 Theory . . . 1520
13.11.4 Assumptions . . . 1521
13.11.5 R Implementation . . . 1521
13.11.6 Output & Results . . . 1521
13.11.7 Interpretation . . . 1521
13.11.8 Practical Tips . . . 1522
13.11.9 For Reviewers . . . 1522
13.11.10 See also — labs in this chapter . . . 1522
13.12 Cross-Validation . . . 1522
13.12.1 Introduction . . . 1522
13.12.2 Prerequisites . . . 1522
13.12.3 Theory . . . 1523
13.12.4 Assumptions . . . 1523
13.12.5 R Implementation . . . 1524
13.12.6 Output & Results . . . 1524
13.12.7 Interpretation . . . 1525
13.12.8 Practical Tips . . . 1525
13.12.9 For Reviewers . . . 1525
13.12.10 See also — labs in this chapter . . . 1525
13.13 Decision Trees (CART) . . . 1526
13.13.1 Introduction . . . 1526
13.13.2 Prerequisites . . . 1526
13.13.3 Theory . . . 1526
13.13.4 Assumptions . . . 1526
13.13.5 R Implementation . . . 1527
13.13.6 Output & Results . . . 1527
13.13.7 Interpretation . . . 1527
13.13.8 Practical Tips . . . 1527
13.13.9 R Packages Used . . . 1528
13.13.10 For Reviewers . . . 1528
13.13.11 See also — labs in this chapter . . . 1528
13.14 Dropout and Early Stopping . . . 1528
13.14.1 Introduction . . . 1528
13.14.2 Prerequisites . . . 1529
13.14.3 Theory . . . 1529
13.14.4 Assumptions . . . 1529

13.14.5 R Implementation . . . 1529
13.14.6 Output & Results . . . 1530
13.14.7 Interpretation . . . 1530
13.14.8 Practical Tips . . . 1530
13.14.9 For Reviewers . . . 1530
13.14.10 See also — labs in this chapter . . . 1531
13.15 Extremely Randomised Trees . . . 1531
13.15.1 Introduction . . . 1531
13.15.2 Prerequisites . . . 1531
13.15.3 Theory . . . 1531
13.15.4 Assumptions . . . 1532
13.15.5 R Implementation . . . 1532
13.15.6 Output & Results . . . 1532
13.15.7 Interpretation . . . 1532
13.15.8 Practical Tips . . . 1533
13.15.9 R Packages Used . . . 1533
13.15.10 For Reviewers . . . 1533
13.15.11 See also — labs in this chapter . . . 1533
13.16 Feature Engineering . . . 1534
13.16.1 Introduction . . . 1534
13.16.2 Prerequisites . . . 1534
13.16.3 Theory . . . 1534
13.16.4 Assumptions . . . 1534
13.16.5 R Implementation . . . 1534
13.16.6 Output & Results . . . 1535
13.16.7 Interpretation . . . 1535
13.16.8 Practical Tips . . . 1535
13.16.9 For Reviewers . . . 1535
13.16.10 See also — labs in this chapter . . . 1535
13.17 Feature Selection . . . 1536
13.17.1 Introduction . . . 1536
13.17.2 Prerequisites . . . 1536
13.17.3 Theory . . . 1536
13.17.4 Assumptions . . . 1536
13.17.5 R Implementation . . . 1537
13.17.6 Output & Results . . . 1537
13.17.7 Interpretation . . . 1537
13.17.8 Practical Tips . . . 1537
13.17.9 R Packages Used . . . 1538
13.17.10 For Reviewers . . . 1538
13.17.11 See also — labs in this chapter . . . 1538
13.18 Feedforward Networks in torch . . . 1539
13.18.1 Introduction . . . 1539
13.18.2 Prerequisites . . . 1539
13.18.3 Theory . . . 1539
13.18.4 Assumptions . . . 1539

13.18.5 R Implementation . . . 1539
13.18.6 Output & Results . . . 1540
13.18.7 Interpretation . . . 1540
13.18.8 Practical Tips . . . 1541
13.18.9 R Packages Used . . . 1541
13.18.10 For Reviewers . . . 1541
13.18.11 See also — labs in this chapter . . . 1541
13.19 Gradient Boosting . . . 1542
13.19.1 Introduction . . . 1542
13.19.2 Prerequisites . . . 1542
13.19.3 Theory . . . 1542
13.19.4 Assumptions . . . 1542
13.19.5 R Implementation . . . 1542
13.19.6 Output & Results . . . 1543
13.19.7 Interpretation . . . 1543
13.19.8 Practical Tips . . . 1543
13.19.9 R Packages Used . . . 1544
13.19.10 For Reviewers . . . 1544
13.19.11 See also — labs in this chapter . . . 1544
13.20 Isolation Forest . . . 1544
13.20.1 Introduction . . . 1544
13.20.2 Prerequisites . . . 1545
13.20.3 Theory . . . 1545
13.20.4 Assumptions . . . 1545
13.20.5 R Implementation . . . 1545
13.20.6 Output & Results . . . 1546
13.20.7 Interpretation . . . 1546
13.20.8 Practical Tips . . . 1546
13.20.9 R Packages Used . . . 1546
13.20.10 For Reviewers . . . 1546
13.20.11 See also — labs in this chapter . . . 1547
13.21 Isotonic Regression Calibration . . . 1547
13.21.1 Introduction . . . 1547
13.21.2 Prerequisites . . . 1547
13.21.3 Theory . . . 1547
13.21.4 Assumptions . . . 1547
13.21.5 R Implementation . . . 1548
13.21.6 Output & Results . . . 1548
13.21.7 Interpretation . . . 1548
13.21.8 Practical Tips . . . 1548
13.21.9 R Packages Used . . . 1549
13.21.10 For Reviewers . . . 1549
13.21.11 See also — labs in this chapter . . . 1549
13.22 k-Nearest Neighbours Classification . . . 1549
13.22.1 Introduction . . . 1549
13.22.2 Prerequisites . . . 1550

13.22.3 Theory . . . 1550
13.22.4 Assumptions . . . 1550
13.22.5 R Implementation . . . 1550
13.22.6 Output & Results . . . 1551
13.22.7 Interpretation . . . 1551
13.22.8 Practical Tips . . . 1551
13.22.9 R Packages Used . . . 1551
13.22.10 For Reviewers . . . 1552
13.22.11 See also — labs in this chapter . . . 1552
13.23 k-Nearest Neighbours Regression . . . 1552
13.23.1 Introduction . . . 1552
13.23.2 Prerequisites . . . 1552
13.23.3 Theory . . . 1552
13.23.4 Assumptions . . . 1553
13.23.5 R Implementation . . . 1553
13.23.6 Output & Results . . . 1553
13.23.7 Interpretation . . . 1553
13.23.8 Practical Tips . . . 1554
13.23.9 R Packages Used . . . 1554
13.23.10 For Reviewers . . . 1554
13.23.11 See also — labs in this chapter . . . 1554
13.24 LightGBM . . . 1555
13.24.1 Introduction . . . 1555
13.24.2 Prerequisites . . . 1555
13.24.3 Theory . . . 1555
13.24.4 Assumptions . . . 1555
13.24.5 R Implementation . . . 1555
13.24.6 Output & Results . . . 1556
13.24.7 Interpretation . . . 1556
13.24.8 Practical Tips . . . 1556
13.24.9 R Packages Used . . . 1557
13.24.10 For Reviewers . . . 1557
13.24.11 See also — labs in this chapter . . . 1557
13.25 LIME Explanations . . . 1557
13.25.1 Introduction . . . 1557
13.25.2 Prerequisites . . . 1558
13.25.3 Theory . . . 1558
13.25.4 Assumptions . . . 1558
13.25.5 R Implementation . . . 1558
13.25.6 Output & Results . . . 1558
13.25.7 Interpretation . . . 1559
13.25.8 Practical Tips . . . 1559
13.25.9 For Reviewers . . . 1559
13.25.10 See also — labs in this chapter . . . 1559
13.26 Linear Discriminant as ML . . . 1559
13.26.1 Introduction . . . 1559

13.26.2 Prerequisites . . . 1560
13.26.3 Theory . . . 1560
13.26.4 Assumptions . . . 1560
13.26.5 R Implementation . . . 1560
13.26.6 Output & Results . . . 1561
13.26.7 Interpretation . . . 1561
13.26.8 Practical Tips . . . 1561
13.26.9 R Packages Used . . . 1562
13.26.10 For Reviewers . . . 1562
13.26.11 See also — labs in this chapter . . . 1562
13.27 Logistic Regression as ML . . . 1562
13.27.1 Introduction . . . 1562
13.27.2 Prerequisites . . . 1562
13.27.3 Theory . . . 1562
13.27.4 Assumptions . . . 1563
13.27.5 R Implementation . . . 1563
13.27.6 Output & Results . . . 1563
13.27.7 Interpretation . . . 1564
13.27.8 Practical Tips . . . 1564
13.27.9 R Packages Used . . . 1564
13.27.10 For Reviewers . . . 1564
13.27.11 See also — labs in this chapter . . . 1564
13.28 A Complete mlr3 Workflow . . . 1565
13.28.1 Introduction . . . 1565
13.28.2 Prerequisites . . . 1565
13.28.3 Theory . . . 1565
13.28.4 Assumptions . . . 1565
13.28.5 R Implementation . . . 1566
13.28.6 Output & Results . . . 1566
13.28.7 Interpretation . . . 1566
13.28.8 Practical Tips . . . 1567
13.28.9 R Packages Used . . . 1567
13.28.10 For Reviewers . . . 1567
13.28.11 See also — labs in this chapter . . . 1567
13.29 Naive Bayes . . . 1568
13.29.1 Introduction . . . 1568
13.29.2 Prerequisites . . . 1568
13.29.3 Theory . . . 1568
13.29.4 Assumptions . . . 1568
13.29.5 R Implementation . . . 1568
13.29.6 Output & Results . . . 1569
13.29.7 Interpretation . . . 1569
13.29.8 Practical Tips . . . 1569
13.29.9 R Packages Used . . . 1570
13.29.10 For Reviewers . . . 1570
13.29.11 See also — labs in this chapter . . . 1570

13.30 Nested Cross-Validation 1570
13.30.1 Introduction 1570
13.30.2 Prerequisites 1571
13.30.3 Theory 1571
13.30.4 Assumptions 1571
13.30.5 R Implementation 1571
13.30.6 Output & Results 1572
13.30.7 Interpretation 1572
13.30.8 Practical Tips 1572
13.30.9 R Packages Used 1573
13.30.10 For Reviewers 1573
13.30.11 See also — labs in this chapter 1573
13.31 Neural Networks: Introduction 1574
13.31.1 Introduction 1574
13.31.2 Prerequisites 1574
13.31.3 Theory 1574
13.31.4 Assumptions 1574
13.31.5 R Implementation 1575
13.31.6 Output & Results 1575
13.31.7 Interpretation 1575
13.31.8 Practical Tips 1575
13.31.9 R Packages Used 1576
13.31.10 For Reviewers 1576
13.31.11 See also — labs in this chapter 1576
13.32 Partial Dependence Plots 1576
13.32.1 Introduction 1576
13.32.2 Prerequisites 1577
13.32.3 Theory 1577
13.32.4 Assumptions 1577
13.32.5 R Implementation 1577
13.32.6 Output & Results 1578
13.32.7 Interpretation 1578
13.32.8 Practical Tips 1578
13.32.9 R Packages Used 1578
13.32.10 For Reviewers 1579
13.32.11 See also — labs in this chapter 1579
13.33 Platt Scaling 1579
13.33.1 Introduction 1579
13.33.2 Prerequisites 1579
13.33.3 Theory 1579
13.33.4 Assumptions 1580
13.33.5 R Implementation 1580
13.33.6 Output & Results 1580
13.33.7 Interpretation 1581
13.33.8 Practical Tips 1581
13.33.9 R Packages Used 1581

13.33.10 For Reviewers . . . 1581
13.33.11 See also — labs in this chapter . . . 1581
13.34 Random Forests . . . 1582
13.34.1 Introduction . . . 1582
13.34.2 Prerequisites . . . 1582
13.34.3 Theory . . . 1582
13.34.4 Assumptions . . . 1582
13.34.5 R Implementation . . . 1583
13.34.6 Output & Results . . . 1583
13.34.7 Interpretation . . . 1583
13.34.8 Practical Tips . . . 1583
13.34.9 R Packages Used . . . 1584
13.34.10 For Reviewers . . . 1584
13.34.11 See also — labs in this chapter . . . 1584
13.35 Preprocessing with recipes . . . 1584
13.35.1 Introduction . . . 1584
13.35.2 Prerequisites . . . 1585
13.35.3 Theory . . . 1585
13.35.4 Assumptions . . . 1585
13.35.5 R Implementation . . . 1585
13.35.6 Output & Results . . . 1586
13.35.7 Interpretation . . . 1586
13.35.8 Practical Tips . . . 1586
13.35.9 R Packages Used . . . 1586
13.35.10 For Reviewers . . . 1587
13.35.11 See also — labs in this chapter . . . 1587
13.36 Regularisation in ML . . . 1587
13.36.1 Introduction . . . 1587
13.36.2 Prerequisites . . . 1587
13.36.3 Theory . . . 1587
13.36.4 Assumptions . . . 1587
13.36.5 R Implementation . . . 1588
13.36.6 Output & Results . . . 1588
13.36.7 Interpretation . . . 1588
13.36.8 Practical Tips . . . 1588
13.36.9 For Reviewers . . . 1589
13.36.10 See also — labs in this chapter . . . 1589
13.37 RNNs and LSTMs: Introduction . . . 1589
13.37.1 Introduction . . . 1589
13.37.2 Prerequisites . . . 1589
13.37.3 Theory . . . 1589
13.37.4 Assumptions . . . 1590
13.37.5 R Implementation . . . 1590
13.37.6 Output & Results . . . 1590
13.37.7 Interpretation . . . 1590
13.37.8 Practical Tips . . . 1591

13.37.9 For Reviewers . . . 1591
13.37.10 See also — labs in this chapter . . . 1591
13.38 SHAP Values . . . 1591
13.38.1 Introduction . . . 1591
13.38.2 Prerequisites . . . 1592
13.38.3 Theory . . . 1592
13.38.4 Assumptions . . . 1592
13.38.5 R Implementation . . . 1592
13.38.6 Output & Results . . . 1593
13.38.7 Interpretation . . . 1593
13.38.8 Practical Tips . . . 1593
13.38.9 For Reviewers . . . 1593
13.38.10 See also — labs in this chapter . . . 1593
13.39 SMOTE for Imbalanced Classes . . . 1594
13.39.1 Introduction . . . 1594
13.39.2 Prerequisites . . . 1594
13.39.3 Theory . . . 1594
13.39.4 Assumptions . . . 1594
13.39.5 R Implementation . . . 1594
13.39.6 Output & Results . . . 1595
13.39.7 Interpretation . . . 1595
13.39.8 Practical Tips . . . 1595
13.39.9 For Reviewers . . . 1595
13.39.10 See also — labs in this chapter . . . 1595
13.40 Stacking Ensembles . . . 1596
13.40.1 Introduction . . . 1596
13.40.2 Prerequisites . . . 1596
13.40.3 Theory . . . 1596
13.40.4 Assumptions . . . 1596
13.40.5 R Implementation . . . 1596
13.40.6 Output & Results . . . 1597
13.40.7 Interpretation . . . 1597
13.40.8 Practical Tips . . . 1597
13.40.9 R Packages Used . . . 1598
13.40.10 For Reviewers . . . 1598
13.40.11 See also — labs in this chapter . . . 1598
13.41 Supervised Learning: Overview . . . 1598
13.41.1 Introduction . . . 1598
13.41.2 Prerequisites . . . 1599
13.41.3 Theory . . . 1599
13.41.4 Assumptions . . . 1599
13.41.5 R Implementation . . . 1599
13.41.6 Output & Results . . . 1600
13.41.7 Interpretation . . . 1600
13.41.8 Practical Tips . . . 1600
13.41.9 For Reviewers . . . 1600

13.41.10 See also — labs in this chapter . . . 1600
13.42 Kernel SVMs . . . 1601
13.42.1 Introduction . . . 1601
13.42.2 Prerequisites . . . 1601
13.42.3 Theory . . . 1601
13.42.4 Assumptions . . . 1601
13.42.5 R Implementation . . . 1601
13.42.6 Output & Results . . . 1602
13.42.7 Interpretation . . . 1602
13.42.8 Practical Tips . . . 1602
13.42.9 R Packages Used . . . 1603
13.42.10 For Reviewers . . . 1603
13.42.11 See also — labs in this chapter . . . 1603
13.43 Linear Support Vector Machines . . . 1603
13.43.1 Introduction . . . 1603
13.43.2 Prerequisites . . . 1603
13.43.3 Theory . . . 1603
13.43.4 Assumptions . . . 1604
13.43.5 R Implementation . . . 1604
13.43.6 Output & Results . . . 1605
13.43.7 Interpretation . . . 1605
13.43.8 Practical Tips . . . 1605
13.43.9 R Packages Used . . . 1605
13.43.10 For Reviewers . . . 1605
13.43.11 See also — labs in this chapter . . . 1606
13.44 A Complete tidymodels Workflow . . . 1606
13.44.1 Introduction . . . 1606
13.44.2 Prerequisites . . . 1606
13.44.3 Theory . . . 1606
13.44.4 Assumptions . . . 1607
13.44.5 R Implementation . . . 1607
13.44.6 Output & Results . . . 1607
13.44.7 Interpretation . . . 1608
13.44.8 Practical Tips . . . 1608
13.44.9 R Packages Used . . . 1608
13.44.10 For Reviewers . . . 1608
13.44.11 See also — labs in this chapter . . . 1608
13.45 Train-Test Splits . . . 1609
13.45.1 Introduction . . . 1609
13.45.2 Prerequisites . . . 1609
13.45.3 Theory . . . 1609
13.45.4 Assumptions . . . 1609
13.45.5 R Implementation . . . 1610
13.45.6 Output & Results . . . 1610
13.45.7 Interpretation . . . 1610
13.45.8 Practical Tips . . . 1610

13.45.9 R Packages Used . . . 1611
13.45.10 For Reviewers . . . 1611
13.45.11 See also — labs in this chapter . . . 1611
13.46 Transformers: Overview . . . 1611
13.46.1 Introduction . . . 1611
13.46.2 Prerequisites . . . 1612
13.46.3 Theory . . . 1612
13.46.4 Assumptions . . . 1612
13.46.5 R Implementation . . . 1612
13.46.6 Output & Results . . . 1613
13.46.7 Interpretation . . . 1613
13.46.8 Practical Tips . . . 1613
13.46.9 For Reviewers . . . 1613
13.46.10 See also — labs in this chapter . . . 1614
13.47 Variable Importance . . . 1614
13.47.1 Introduction . . . 1614
13.47.2 Prerequisites . . . 1614
13.47.3 Theory . . . 1614
13.47.4 Assumptions . . . 1614
13.47.5 R Implementation . . . 1615
13.47.6 Output & Results . . . 1615
13.47.7 Interpretation . . . 1615
13.47.8 Practical Tips . . . 1615
13.47.9 Reporting . . . 1616
13.47.10 For Reviewers . . . 1616
13.47.11 See also — labs in this chapter . . . 1616
13.48 XGBoost . . . 1616
13.48.1 Introduction . . . 1616
13.48.2 Prerequisites . . . 1617
13.48.3 Theory . . . 1617
13.48.4 Assumptions . . . 1617
13.48.5 R Implementation . . . 1617
13.48.6 Output & Results . . . 1618
13.48.7 Interpretation . . . 1618
13.48.8 Practical Tips . . . 1618
13.48.9 R Packages Used . . . 1619
13.48.10 For Reviewers . . . 1619
13.48.11 See also — labs in this chapter . . . 1619
13.49 Week 1, Session 1 — Cross-validation, nested CV, bootstrap .632+ . . . 1619
13.49.1 Learning objectives . . . 1619
13.49.2 Prerequisites . . . 1619
13.49.3 Background . . . 1620
13.49.4 Setup . . . 1620
13.49.5 1. Goal . . . 1620
13.49.6 2. Approach . . . 1620

13.49.7 3. Execution 1621
13.49.8 4. Check 1621
13.49.9 5. Report 1622
13.49.10 Common pitfalls 1622
13.49.11 Further reading 1622
13.49.12 Session info 1622
13.49.13 See also — chapter index 1623
13.50 Week 4, Session 4 — FDR, knockoffs, replication crisis . . 1623
13.50.1 Learning objectives 1623
13.50.2 Prerequisites 1623
13.50.3 Background 1623
13.50.4 Setup 1624
13.50.5 1. Goal 1624
13.50.6 2. Approach 1624
13.50.7 3. Execution 1624
13.50.8 4. Check 1625
13.50.9 5. Report 1625
13.50.10 Common pitfalls 1625
13.50.11 Further reading 1625
13.50.12 Session info 1625
13.50.13 See also — chapter index 1626
13.51 Week 2, Session 4 — Imaging and sequence models intro . 1626
13.51.1 Learning objectives 1626
13.51.2 Prerequisites 1626
13.51.3 Background 1626
13.51.4 Setup 1626
13.51.5 1. Goal 1627
13.51.6 2. Approach 1627
13.51.7 3. Execution 1627
13.51.8 4. Check 1628
13.51.9 5. Report 1628
13.51.10 Common pitfalls 1628
13.51.11 Further reading 1629
13.51.12 Session info 1629
13.51.13 See also — chapter index 1629
13.52 Week 2, Session 2 — Interpretability and SHAP 1629
13.52.1 Learning objectives 1629
13.52.2 Prerequisites 1629
13.52.3 Background 1629
13.52.4 Setup 1630
13.52.5 1. Goal 1630
13.52.6 2. Approach 1630
13.52.7 3. Execution 1630
13.52.8 4. Check 1631
13.52.9 5. Report 1631
13.52.10 Common pitfalls 1631

13.52.11 Further reading . . . 1632
13.52.12 Session info . . . 1632
13.52.13 See also — chapter index . . . 1632
13.53 Week 1, Session 2 — Regularisation: ridge, lasso, elastic net 1632
13.53.1 Learning objectives . . . 1632
13.53.2 Prerequisites . . . 1632
13.53.3 Background . . . 1632
13.53.4 Setup . . . 1633
13.53.5 1. Goal . . . 1633
13.53.6 2. Approach . . . 1633
13.53.7 3. Execution . . . 1633
13.53.8 4. Check . . . 1634
13.53.9 5. Report . . . 1634
13.53.10 Common pitfalls . . . 1634
13.53.11 Further reading . . . 1634
13.53.12 Session info . . . 1635
13.53.13 See also — chapter index . . . 1635
13.54 Week 2, Session 3 — Tabular neural networks with torch . 1635
13.54.1 Learning objectives . . . 1635
13.54.2 Prerequisites . . . 1635
13.54.3 Background . . . 1635
13.54.4 Setup . . . 1636
13.54.5 1. Goal . . . 1636
13.54.6 2. Approach . . . 1636
13.54.7 3. Execution . . . 1636
13.54.8 4. Check . . . 1637
13.54.9 5. Report . . . 1637
13.54.10 Common pitfalls . . . 1637
13.54.11 Further reading . . . 1638
13.54.12 Session info . . . 1638
13.54.13 See also — chapter index . . . 1638
13.55 Week 2, Session 5 — tidymodels pipelines . . . 1638
13.55.1 Learning objectives . . . 1638
13.55.2 Prerequisites . . . 1638
13.55.3 Background . . . 1638
13.55.4 Setup . . . 1639
13.55.5 1. Goal . . . 1639
13.55.6 2. Approach . . . 1639
13.55.7 3. Execution . . . 1639
13.55.8 4. Check . . . 1639
13.55.9 5. Report . . . 1640
13.55.10 Common pitfalls . . . 1640
13.55.11 Further reading . . . 1640
13.55.12 Session info . . . 1640
13.55.13 See also — chapter index . . . 1640
13.56 Week 2, Session 1 — Trees, random forests, gradient boosting 1641

13.56.1 Learning objectives . . . 1641
13.56.2 Prerequisites . . . 1641
13.56.3 Background . . . 1641
13.56.4 Setup . . . 1641
13.56.5 1. Goal . . . 1642
13.56.6 2. Approach . . . 1642
13.56.7 3. Execution . . . 1642
13.56.8 4. Check . . . 1642
13.56.9 5. Report . . . 1643
13.56.10 Common pitfalls . . . 1643
13.56.11 Further reading . . . 1643
13.56.12 Session info . . . 1643
13.56.13 See also — chapter index . . . 1644

14 Clinical Biostatistics **1645**
14.1 Method pages . . . 1645
14.2 Labs . . . 1646
14.3 Adaptive Trial Designs . . . 1646
14.3.1 Introduction . . . 1646
14.3.2 Prerequisites . . . 1647
14.3.3 Theory . . . 1647
14.3.4 Assumptions . . . 1647
14.3.5 R Implementation . . . 1647
14.3.6 Output & Results . . . 1647
14.3.7 Interpretation . . . 1648
14.3.8 Practical Tips . . . 1648
14.3.9 For Reviewers . . . 1648
14.3.10 See also — labs in this chapter . . . 1648
14.4 Alpha-Spending Functions . . . 1648
14.4.1 Introduction . . . 1648
14.4.2 Prerequisites . . . 1648
14.4.3 Theory . . . 1649
14.4.4 Assumptions . . . 1649
14.4.5 R Implementation . . . 1649
14.4.6 Output & Results . . . 1649
14.4.7 Interpretation . . . 1650
14.4.8 Practical Tips . . . 1650
14.4.9 For Reviewers . . . 1650
14.4.10 See also — labs in this chapter . . . 1650
14.5 Baseline Adjustment with ANCOVA . . . 1650
14.5.1 Introduction . . . 1650
14.5.2 Prerequisites . . . 1650
14.5.3 Theory . . . 1651
14.5.4 Assumptions . . . 1651
14.5.5 R Implementation . . . 1651
14.5.6 Output & Results . . . 1651

14.5.7 Interpretation 1652
14.5.8 Practical Tips 1652
14.5.9 For Reviewers 1652
14.5.10 See also — labs in this chapter 1652
14.6 Bland-Altman Limits of Agreement 1652
14.6.1 Introduction 1652
14.6.2 Prerequisites 1652
14.6.3 Theory 1653
14.6.4 Assumptions 1653
14.6.5 R Implementation 1653
14.6.6 Output & Results 1653
14.6.7 Interpretation 1654
14.6.8 Practical Tips 1654
14.6.9 For Reviewers 1654
14.6.10 See also — labs in this chapter 1654
14.7 Blinding Procedures 1654
14.7.1 Introduction 1654
14.7.2 Prerequisites 1654
14.7.3 Theory 1655
14.7.4 Assumptions 1655
14.7.5 R Implementation 1655
14.7.6 Output & Results 1655
14.7.7 Interpretation 1655
14.7.8 Practical Tips 1656
14.7.9 For Reviewers 1656
14.7.10 See also — labs in this chapter 1656
14.8 Block Randomisation 1656
14.8.1 Introduction 1656
14.8.2 Prerequisites 1656
14.8.3 Theory 1657
14.8.4 Assumptions 1657
14.8.5 R Implementation 1657
14.8.6 Output & Results 1657
14.8.7 Interpretation 1658
14.8.8 Practical Tips 1658
14.8.9 R Packages Used 1658
14.8.10 For Reviewers 1659
14.8.11 See also — labs in this chapter 1659
14.9 Cluster-Randomised Trials 1659
14.9.1 Introduction 1659
14.9.2 Prerequisites 1659
14.9.3 Theory 1659
14.9.4 Assumptions 1659
14.9.5 R Implementation 1660
14.9.6 Output & Results 1660
14.9.7 Interpretation 1660

14.9.8 Practical Tips . . . 1660
14.9.9 Reporting . . . 1661
14.9.10 For Reviewers . . . 1661
14.9.11 See also — labs in this chapter . . . 1661
14.10 Cohen's Kappa . . . 1661
14.10.1 Introduction . . . 1661
14.10.2 Prerequisites . . . 1662
14.10.3 Theory . . . 1662
14.10.4 Assumptions . . . 1662
14.10.5 R Implementation . . . 1662
14.10.6 Output & Results . . . 1663
14.10.7 Interpretation . . . 1663
14.10.8 Practical Tips . . . 1663
14.10.9 R Packages Used . . . 1663
14.10.10 For Reviewers . . . 1664
14.10.11 See also — labs in this chapter . . . 1664
14.11 Conditional Power . . . 1664
14.11.1 Introduction . . . 1664
14.11.2 Prerequisites . . . 1664
14.11.3 Theory . . . 1664
14.11.4 Assumptions . . . 1665
14.11.5 R Implementation . . . 1665
14.11.6 Output & Results . . . 1665
14.11.7 Interpretation . . . 1665
14.11.8 Practical Tips . . . 1665
14.11.9 Reporting . . . 1666
14.11.10 For Reviewers . . . 1666
14.11.11 See also — labs in this chapter . . . 1666
14.12 Cutpoint Selection . . . 1666
14.12.1 Introduction . . . 1666
14.12.2 Prerequisites . . . 1666
14.12.3 Theory . . . 1667
14.12.4 Assumptions . . . 1667
14.12.5 R Implementation . . . 1667
14.12.6 Output & Results . . . 1667
14.12.7 Interpretation . . . 1667
14.12.8 Practical Tips . . . 1668
14.12.9 For Reviewers . . . 1668
14.12.10 See also — labs in this chapter . . . 1668
14.13 Diagnostic Test Accuracy . . . 1668
14.13.1 Introduction . . . 1668
14.13.2 Prerequisites . . . 1668
14.13.3 Theory . . . 1669
14.13.4 Assumptions . . . 1669
14.13.5 R Implementation . . . 1670
14.13.6 Output & Results . . . 1670

14.13.7 Interpretation 1670
14.13.8 Practical Tips 1671
14.13.9 For Reviewers 1671
14.13.10 See also — labs in this chapter 1671
14.14 Clinical Equivalence Trials 1671
14.14.1 Introduction 1671
14.14.2 Prerequisites 1672
14.14.3 Theory 1672
14.14.4 Assumptions 1672
14.14.5 R Implementation 1672
14.14.6 Output & Results 1673
14.14.7 Interpretation 1673
14.14.8 Practical Tips 1673
14.14.9 R Packages Used 1674
14.14.10 For Reviewers 1674
14.14.11 See also — labs in this chapter 1674
14.15 Factorial Trials 1674
14.15.1 Introduction 1674
14.15.2 Prerequisites 1674
14.15.3 Theory 1675
14.15.4 Assumptions 1675
14.15.5 R Implementation 1675
14.15.6 Output & Results 1675
14.15.7 Interpretation 1675
14.15.8 Practical Tips 1675
14.15.9 For Reviewers 1676
14.15.10 See also — labs in this chapter 1676
14.16 Subgroup Forest Plots 1676
14.16.1 Introduction 1676
14.16.2 Prerequisites 1676
14.16.3 Theory 1677
14.16.4 Assumptions 1677
14.16.5 R Implementation 1677
14.16.6 Output & Results 1678
14.16.7 Interpretation 1678
14.16.8 Practical Tips 1678
14.16.9 R Packages Used 1678
14.16.10 For Reviewers 1679
14.16.11 See also — labs in this chapter 1679
14.17 Intraclass Correlation Coefficient (ICC) 1679
14.17.1 Introduction 1679
14.17.2 Prerequisites 1679
14.17.3 Theory 1679
14.17.4 Assumptions 1679
14.17.5 R Implementation 1680
14.17.6 Output & Results 1680

14.17.7 Interpretation 1680
14.17.8 Practical Tips 1680
14.17.9 For Reviewers 1681
14.17.10 See also — labs in this chapter 1681
14.18 Interim Analyses and Group Sequential Designs 1681
14.18.1 Introduction 1681
14.18.2 Prerequisites 1681
14.18.3 Theory 1681
14.18.4 Assumptions 1681
14.18.5 R Implementation 1682
14.18.6 Output & Results 1682
14.18.7 Interpretation 1682
14.18.8 Practical Tips 1682
14.18.9 For Reviewers 1683
14.18.10 See also — labs in this chapter 1683
14.19 ITT vs Per-Protocol Analysis 1683
14.19.1 Introduction 1683
14.19.2 Prerequisites 1683
14.19.3 Theory 1683
14.19.4 Assumptions 1683
14.19.5 R Implementation 1684
14.19.6 Output & Results 1684
14.19.7 Interpretation 1684
14.19.8 Practical Tips 1684
14.19.9 For Reviewers 1685
14.19.10 See also — labs in this chapter 1685
14.20 Likelihood Ratios 1685
14.20.1 Introduction 1685
14.20.2 Prerequisites 1685
14.20.3 Theory 1685
14.20.4 Assumptions 1686
14.20.5 R Implementation 1686
14.20.6 Output & Results 1686
14.20.7 Interpretation 1686
14.20.8 Practical Tips 1687
14.20.9 R Packages Used 1687
14.20.10 For Reviewers 1687
14.20.11 See also — labs in this chapter 1688
14.21 Minimisation Algorithm 1688
14.21.1 Introduction 1688
14.21.2 Prerequisites 1688
14.21.3 Theory 1688
14.21.4 Assumptions 1689
14.21.5 R Implementation 1689
14.21.6 Output & Results 1689
14.21.7 Interpretation 1689

14.21.8 Practical Tips 1690
14.21.9 R Packages Used 1690
14.21.10 For Reviewers 1690
14.21.11 See also — labs in this chapter 1691
14.22 Missing Data in RCTs 1691
14.22.1 Introduction 1691
14.22.2 Prerequisites 1691
14.22.3 Theory 1691
14.22.4 Assumptions 1691
14.22.5 R Implementation 1691
14.22.6 Output & Results 1692
14.22.7 Interpretation 1692
14.22.8 Practical Tips 1692
14.22.9 For Reviewers 1692
14.22.10 See also — labs in this chapter 1693
14.23 Multiple Imputation 1693
14.23.1 Introduction 1693
14.23.2 Prerequisites 1693
14.23.3 Theory 1693
14.23.4 Assumptions 1693
14.23.5 R Implementation 1693
14.23.6 Output & Results 1694
14.23.7 Interpretation 1694
14.23.8 Practical Tips 1694
14.23.9 For Reviewers 1694
14.23.10 See also — labs in this chapter 1695
14.24 Non-Inferiority Margin Selection 1695
14.24.1 Introduction 1695
14.24.2 Prerequisites 1695
14.24.3 Theory 1695
14.24.4 Assumptions 1695
14.24.5 R Implementation 1695
14.24.6 Output & Results 1696
14.24.7 Interpretation 1696
14.24.8 Practical Tips 1696
14.24.9 For Reviewers 1696
14.24.10 See also — labs in this chapter 1697
14.25 O'Brien-Fleming Boundary 1697
14.25.1 Introduction 1697
14.25.2 Prerequisites 1697
14.25.3 Theory 1697
14.25.4 Assumptions 1697
14.25.5 R Implementation 1698
14.25.6 Output & Results 1698
14.25.7 Interpretation 1698
14.25.8 Practical Tips 1698

14.25.9 R Packages Used . . . 1699
14.25.10 For Reviewers . . . 1699
14.25.11 See also — labs in this chapter . . . 1699
14.26 Pocock Boundary . . . 1700
14.26.1 Introduction . . . 1700
14.26.2 Prerequisites . . . 1700
14.26.3 Theory . . . 1700
14.26.4 Assumptions . . . 1700
14.26.5 R Implementation . . . 1700
14.26.6 Output & Results . . . 1701
14.26.7 Interpretation . . . 1701
14.26.8 Practical Tips . . . 1701
14.26.9 R Packages Used . . . 1702
14.26.10 For Reviewers . . . 1702
14.26.11 See also — labs in this chapter . . . 1702
14.27 Predictive Values and Prevalence . . . 1703
14.27.1 Introduction . . . 1703
14.27.2 Prerequisites . . . 1703
14.27.3 Theory . . . 1703
14.27.4 Assumptions . . . 1703
14.27.5 R Implementation . . . 1703
14.27.6 Output & Results . . . 1704
14.27.7 Interpretation . . . 1704
14.27.8 Practical Tips . . . 1704
14.27.9 For Reviewers . . . 1704
14.27.10 See also — labs in this chapter . . . 1704
14.28 Randomisation Methods . . . 1705
14.28.1 Introduction . . . 1705
14.28.2 Prerequisites . . . 1705
14.28.3 Theory . . . 1705
14.28.4 Assumptions . . . 1705
14.28.5 R Implementation . . . 1705
14.28.6 Output & Results . . . 1706
14.28.7 Interpretation . . . 1706
14.28.8 Practical Tips . . . 1706
14.28.9 For Reviewers . . . 1706
14.28.10 See also — labs in this chapter . . . 1706
14.29 Crossover RCT Design . . . 1707
14.29.1 Introduction . . . 1707
14.29.2 Prerequisites . . . 1707
14.29.3 Theory . . . 1707
14.29.4 Assumptions . . . 1707
14.29.5 R Implementation . . . 1708
14.29.6 Output & Results . . . 1708
14.29.7 Interpretation . . . 1708
14.29.8 Practical Tips . . . 1709

14.29.9 R Packages Used . . . 1709
14.29.10 For Reviewers . . . 1709
14.29.11 See also — labs in this chapter . . . 1709
14.30 Parallel-Group RCT Design . . . 1710
14.30.1 Introduction . . . 1710
14.30.2 Prerequisites . . . 1710
14.30.3 Theory . . . 1710
14.30.4 Assumptions . . . 1710
14.30.5 R Implementation . . . 1710
14.30.6 Output & Results . . . 1711
14.30.7 Interpretation . . . 1711
14.30.8 Practical Tips . . . 1711
14.30.9 For Reviewers . . . 1711
14.30.10 See also — labs in this chapter . . . 1711
14.31 Reliability and Cronbach's Alpha . . . 1712
14.31.1 Introduction . . . 1712
14.31.2 Prerequisites . . . 1712
14.31.3 Theory . . . 1712
14.31.4 Assumptions . . . 1712
14.31.5 R Implementation . . . 1713
14.31.6 Output & Results . . . 1713
14.31.7 Interpretation . . . 1713
14.31.8 Practical Tips . . . 1713
14.31.9 R Packages Used . . . 1714
14.31.10 For Reviewers . . . 1714
14.31.11 See also — labs in this chapter . . . 1714
14.32 ROC Analysis . . . 1714
14.32.1 Introduction . . . 1714
14.32.2 Prerequisites . . . 1715
14.32.3 Theory . . . 1715
14.32.4 Assumptions . . . 1715
14.32.5 R Implementation . . . 1715
14.32.6 Output & Results . . . 1716
14.32.7 Interpretation . . . 1716
14.32.8 Practical Tips . . . 1716
14.32.9 R Packages Used . . . 1717
14.32.10 For Reviewers . . . 1717
14.32.11 See also — labs in this chapter . . . 1717
14.33 Sample Size Re-Estimation . . . 1717
14.33.1 Introduction . . . 1717
14.33.2 Prerequisites . . . 1718
14.33.3 Theory . . . 1718
14.33.4 Assumptions . . . 1718
14.33.5 R Implementation . . . 1718
14.33.6 Output & Results . . . 1719
14.33.7 Interpretation . . . 1719

14.33.8 Practical Tips . . . 1719
14.33.9 R Packages Used . . . 1720
14.33.10 For Reviewers . . . 1720
14.33.11 See also — labs in this chapter . . . 1720
14.34 Sensitivity Analyses in Clinical Trials . . . 1720
14.34.1 Introduction . . . 1720
14.34.2 Prerequisites . . . 1720
14.34.3 Theory . . . 1720
14.34.4 Assumptions . . . 1721
14.34.5 R Implementation . . . 1721
14.34.6 Output & Results . . . 1721
14.34.7 Interpretation . . . 1722
14.34.8 Practical Tips . . . 1722
14.34.9 For Reviewers . . . 1722
14.34.10 See also — labs in this chapter . . . 1722
14.35 Stepped-Wedge Trial . . . 1722
14.35.1 Introduction . . . 1722
14.35.2 Prerequisites . . . 1723
14.35.3 Theory . . . 1723
14.35.4 Assumptions . . . 1723
14.35.5 R Implementation . . . 1723
14.35.6 Output & Results . . . 1724
14.35.7 Interpretation . . . 1724
14.35.8 Practical Tips . . . 1724
14.35.9 R Packages Used . . . 1725
14.35.10 For Reviewers . . . 1725
14.35.11 See also — labs in this chapter . . . 1725
14.36 Stratified Randomisation . . . 1725
14.36.1 Introduction . . . 1725
14.36.2 Prerequisites . . . 1726
14.36.3 Theory . . . 1726
14.36.4 Assumptions . . . 1726
14.36.5 R Implementation . . . 1726
14.36.6 Output & Results . . . 1727
14.36.7 Interpretation . . . 1727
14.36.8 Practical Tips . . . 1727
14.36.9 R Packages Used . . . 1728
14.36.10 For Reviewers . . . 1728
14.36.11 See also — labs in this chapter . . . 1728
14.37 Subgroup Analyses . . . 1728
14.37.1 Introduction . . . 1728
14.37.2 Prerequisites . . . 1729
14.37.3 Theory . . . 1729
14.37.4 Assumptions . . . 1729
14.37.5 R Implementation . . . 1729
14.37.6 Output & Results . . . 1730

14.37.7 Interpretation 1730
14.37.8 Practical Tips 1730
14.37.9 R Packages Used 1731
14.37.10 For Reviewers 1731
14.37.11 See also — labs in this chapter 1731
14.38 Weighted Kappa 1731
14.38.1 Introduction 1731
14.38.2 Prerequisites 1731
14.38.3 Theory 1731
14.38.4 Assumptions 1732
14.38.5 R Implementation 1732
14.38.6 Output & Results 1732
14.38.7 Interpretation 1732
14.38.8 Practical Tips 1732
14.38.9 For Reviewers 1733
14.38.10 See also — labs in this chapter 1733
14.39 Week 3, Session 3 — Biomarker statistics (Youden, NRI, decision curves) 1733
14.39.1 Learning objectives 1733
14.39.2 Prerequisites 1733
14.39.3 Background 1733
14.39.4 Setup 1734
14.39.5 1. Hypothesis 1734
14.39.6 2. Visualise 1734
14.39.7 3. Assumptions 1734
14.39.8 4. Conduct 1734
14.39.9 5. Concluding statement 1735
14.39.10 Common pitfalls 1736
14.39.11 Further reading 1736
14.39.12 Session info 1736
14.39.13 See also — chapter index 1736
14.40 Week 2, Session 3 — Diagnostic testing: Se, Sp, PPV, NPV, LR 1736
14.40.1 Learning objectives 1736
14.40.2 Prerequisites 1736
14.40.3 Background 1736
14.40.4 Setup 1737
14.40.5 1. Hypothesis 1737
14.40.6 2. Visualise 1737
14.40.7 3. Assumptions 1738
14.40.8 4. Conduct 1738
14.40.9 5. Concluding statement 1739
14.40.10 Common pitfalls 1739
14.40.11 Further reading 1739
14.40.12 Session info 1740
14.40.13 See also — chapter index 1740

14.41 Week 4, Session 2 — Kappa, ICC, Bland–Altman 1740
14.41.1 Learning objectives 1740
14.41.2 Prerequisites 1740
14.41.3 Background 1740
14.41.4 Setup 1741
14.41.5 1. Goal 1741
14.41.6 2. Approach 1741
14.41.7 3. Execution 1741
14.41.8 4. Check 1742
14.41.9 5. Report 1742
14.41.10 Common pitfalls 1742
14.41.11 Further reading 1742
14.41.12 Session info 1743
14.41.13 See also — chapter index 1743
14.42 Week 4, Session 5 — TRIPOD-AI, fairness auditing, reproducibility at scale 1743
14.42.1 Learning objectives 1743
14.42.2 Prerequisites 1743
14.42.3 Background 1743
14.42.4 Setup 1744
14.42.5 1. Goal 1744
14.42.6 2. Approach 1744
14.42.7 3. Execution 1744
14.42.8 4. Check 1745
14.42.9 5. Report 1746
14.42.10 Common pitfalls 1746
14.42.11 Further reading 1746
14.42.12 Session info 1746
14.42.13 See also — chapter index 1746

15 Meta-Analysis **1747**
15.1 Method pages 1747
15.2 Labs 1748
15.3 Bivariate SROC Curves 1748
15.3.1 Introduction 1748
15.3.2 Prerequisites 1748
15.3.3 Theory 1748
15.3.4 Assumptions 1749
15.3.5 R Implementation 1749
15.3.6 Output & Results 1749
15.3.7 Interpretation 1749
15.3.8 Practical Tips 1749
15.3.9 For Reviewers 1750
15.3.10 See also — labs in this chapter 1750
15.4 Pooling Correlation Coefficients 1750
15.4.1 Introduction 1750

15.4.2 Prerequisites 1750
15.4.3 Theory 1750
15.4.4 Assumptions 1751
15.4.5 R Implementation 1751
15.4.6 Output & Results 1751
15.4.7 Interpretation 1752
15.4.8 Practical Tips 1752
15.4.9 R Packages Used 1752
15.4.10 For Reviewers 1752
15.4.11 See also — labs in this chapter 1752
15.5 Cumulative Meta-Analysis 1753
15.5.1 Introduction 1753
15.5.2 Prerequisites 1753
15.5.3 Theory 1753
15.5.4 Assumptions 1753
15.5.5 R Implementation 1754
15.5.6 Output & Results 1754
15.5.7 Interpretation 1754
15.5.8 Practical Tips 1754
15.5.9 R Packages Used 1755
15.5.10 For Reviewers 1755
15.5.11 See also — labs in this chapter 1755
15.6 Meta-Analysis of Diagnostic Accuracy 1755
15.6.1 Introduction 1755
15.6.2 Prerequisites 1756
15.6.3 Theory 1756
15.6.4 Assumptions 1756
15.6.5 R Implementation 1756
15.6.6 Output & Results 1757
15.6.7 Interpretation 1757
15.6.8 Practical Tips 1757
15.6.9 R Packages Used 1758
15.6.10 For Reviewers 1758
15.6.11 See also — labs in this chapter 1758
15.7 Converting Between Effect Sizes 1758
15.7.1 Introduction 1758
15.7.2 Prerequisites 1758
15.7.3 Theory 1759
15.7.4 Assumptions 1759
15.7.5 R Implementation 1759
15.7.6 Output & Results 1760
15.7.7 Interpretation 1760
15.7.8 Practical Tips 1760
15.7.9 R Packages Used 1760
15.7.10 For Reviewers 1761
15.7.11 See also — labs in this chapter 1761

15.8 Egger's Test for Funnel Asymmetry . . . 1761
15.8.1 Introduction . . . 1761
15.8.2 Prerequisites . . . 1761
15.8.3 Theory . . . 1761
15.8.4 Assumptions . . . 1762
15.8.5 R Implementation . . . 1762
15.8.6 Output & Results . . . 1762
15.8.7 Interpretation . . . 1763
15.8.8 Practical Tips . . . 1763
15.8.9 R Packages Used . . . 1763
15.8.10 For Reviewers . . . 1763
15.8.11 See also — labs in this chapter . . . 1764
15.9 Fixed-Effect vs. Random-Effects Meta-Analysis . . . 1764
15.9.1 Introduction . . . 1764
15.9.2 Prerequisites . . . 1764
15.9.3 Theory . . . 1764
15.9.4 Assumptions . . . 1765
15.9.5 R Implementation . . . 1765
15.9.6 Output & Results . . . 1766
15.9.7 Interpretation . . . 1766
15.9.8 Practical Tips . . . 1766
15.9.9 For Reviewers . . . 1767
15.9.10 See also — labs in this chapter . . . 1767
15.10 Forest Plots in Meta-Analysis . . . 1767
15.10.1 Introduction . . . 1767
15.10.2 Prerequisites . . . 1767
15.10.3 Theory . . . 1768
15.10.4 Assumptions . . . 1768
15.10.5 R Implementation . . . 1768
15.10.6 Output & Results . . . 1768
15.10.7 Interpretation . . . 1769
15.10.8 Practical Tips . . . 1769
15.10.9 R Packages Used . . . 1769
15.10.10 For Reviewers . . . 1769
15.10.11 See also — labs in this chapter . . . 1770
15.11 Funnel Plots . . . 1770
15.11.1 Introduction . . . 1770
15.11.2 Prerequisites . . . 1770
15.11.3 Theory . . . 1770
15.11.4 Assumptions . . . 1770
15.11.5 R Implementation . . . 1771
15.11.6 Output & Results . . . 1771
15.11.7 Interpretation . . . 1771
15.11.8 Practical Tips . . . 1771
15.11.9 R Packages Used . . . 1772
15.11.10 For Reviewers . . . 1772

15.11.11 See also — labs in this chapter 1772
15.12 Quantifying Heterogeneity: Q and I^2 1772
15.12.1 Introduction 1772
15.12.2 Prerequisites 1773
15.12.3 Theory 1773
15.12.4 Assumptions 1773
15.12.5 R Implementation 1773
15.12.6 Output & Results 1774
15.12.7 Interpretation 1774
15.12.8 Practical Tips 1774
15.12.9 R Packages Used 1775
15.12.10 For Reviewers 1775
15.12.11 See also — labs in this chapter 1775
15.13 Individual Participant Data Meta-Analysis 1775
15.13.1 Introduction 1775
15.13.2 Prerequisites 1775
15.13.3 Theory 1776
15.13.4 Assumptions 1776
15.13.5 R Implementation 1776
15.13.6 Output & Results 1777
15.13.7 Interpretation 1777
15.13.8 Practical Tips 1777
15.13.9 For Reviewers 1777
15.13.10 See also — labs in this chapter 1777
15.14 Leave-One-Out Meta-Analysis 1778
15.14.1 Introduction 1778
15.14.2 Prerequisites 1778
15.14.3 Theory 1778
15.14.4 Assumptions 1778
15.14.5 R Implementation 1778
15.14.6 Output & Results 1779
15.14.7 Interpretation 1779
15.14.8 Practical Tips 1779
15.14.9 R Packages Used 1779
15.14.10 For Reviewers 1780
15.14.11 See also — labs in this chapter 1780
15.15 Pooling Log-Odds Ratios 1780
15.15.1 Introduction 1780
15.15.2 Prerequisites 1780
15.15.3 Theory 1780
15.15.4 Assumptions 1781
15.15.5 R Implementation 1781
15.15.6 Output & Results 1781
15.15.7 Interpretation 1782
15.15.8 Practical Tips 1782
15.15.9 R Packages Used 1782

15.15.10 For Reviewers . . . 1782
15.15.11 See also — labs in this chapter . . . 1783
15.16 Meta-Regression . . . 1783
15.16.1 Introduction . . . 1783
15.16.2 Prerequisites . . . 1783
15.16.3 Theory . . . 1783
15.16.4 Assumptions . . . 1783
15.16.5 R Implementation . . . 1784
15.16.6 Output & Results . . . 1784
15.16.7 Interpretation . . . 1784
15.16.8 Practical Tips . . . 1784
15.16.9 R Packages Used . . . 1785
15.16.10 For Reviewers . . . 1785
15.16.11 See also — labs in this chapter . . . 1785
15.17 Network Meta-Analysis: Introduction . . . 1785
15.17.1 Introduction . . . 1785
15.17.2 Prerequisites . . . 1786
15.17.3 Theory . . . 1786
15.17.4 Assumptions . . . 1786
15.17.5 R Implementation . . . 1786
15.17.6 Output & Results . . . 1786
15.17.7 Interpretation . . . 1787
15.17.8 Practical Tips . . . 1787
15.17.9 For Reviewers . . . 1787
15.17.10 See also — labs in this chapter . . . 1787
15.18 NMA Consistency and Inconsistency . . . 1787
15.18.1 Introduction . . . 1787
15.18.2 Prerequisites . . . 1788
15.18.3 Theory . . . 1788
15.18.4 Assumptions . . . 1788
15.18.5 R Implementation . . . 1788
15.18.6 Output & Results . . . 1789
15.18.7 Interpretation . . . 1789
15.18.8 Practical Tips . . . 1789
15.18.9 R Packages Used . . . 1790
15.18.10 For Reviewers . . . 1790
15.18.11 See also — labs in this chapter . . . 1790
15.19 NMA League Tables . . . 1790
15.19.1 Introduction . . . 1790
15.19.2 Prerequisites . . . 1790
15.19.3 Theory . . . 1790
15.19.4 Assumptions . . . 1791
15.19.5 R Implementation . . . 1791
15.19.6 Output & Results . . . 1791
15.19.7 Interpretation . . . 1792
15.19.8 Practical Tips . . . 1792

15.19.9 R Packages Used 1792
15.19.10 For Reviewers 1792
15.19.11 See also — labs in this chapter 1793
15.20 SUCRA Rankings 1793
15.20.1 Introduction 1793
15.20.2 Prerequisites 1793
15.20.3 Theory 1793
15.20.4 Assumptions 1793
15.20.5 R Implementation 1794
15.20.6 Output & Results 1794
15.20.7 Interpretation 1794
15.20.8 Practical Tips 1795
15.20.9 R Packages Used 1795
15.20.10 For Reviewers 1795
15.20.11 See also — labs in this chapter 1795
15.21 Prediction Intervals in Meta-Analysis 1796
15.21.1 Introduction 1796
15.21.2 Prerequisites 1796
15.21.3 Theory 1796
15.21.4 Assumptions 1796
15.21.5 R Implementation 1796
15.21.6 Output & Results 1797
15.21.7 Interpretation 1797
15.21.8 Practical Tips 1797
15.21.9 R Packages Used 1798
15.21.10 For Reviewers 1798
15.21.11 See also — labs in this chapter 1798
15.22 Pooling Risk Ratios 1798
15.22.1 Introduction 1798
15.22.2 Prerequisites 1798
15.22.3 Theory 1799
15.22.4 Assumptions 1799
15.22.5 R Implementation 1799
15.22.6 Output & Results 1799
15.22.7 Interpretation 1800
15.22.8 Practical Tips 1800
15.22.9 R Packages Used 1800
15.22.10 For Reviewers 1800
15.22.11 See also — labs in this chapter 1801
15.23 Selection Models for Publication Bias 1801
15.23.1 Introduction 1801
15.23.2 Prerequisites 1801
15.23.3 Theory 1801
15.23.4 Assumptions 1802
15.23.5 R Implementation 1802
15.23.6 Output & Results 1802

15.23.7 Interpretation . . . 1802
15.23.8 Practical Tips . . . 1802
15.23.9 R Packages Used . . . 1803
15.23.10 For Reviewers . . . 1803
15.23.11 See also — labs in this chapter . . . 1803
15.24 Standardised Mean Difference: Hedges' g . . . 1804
15.24.1 Introduction . . . 1804
15.24.2 Prerequisites . . . 1804
15.24.3 Theory . . . 1804
15.24.4 Assumptions . . . 1804
15.24.5 R Implementation . . . 1804
15.24.6 Output & Results . . . 1805
15.24.7 Interpretation . . . 1805
15.24.8 Practical Tips . . . 1805
15.24.9 R Packages Used . . . 1806
15.24.10 For Reviewers . . . 1806
15.24.11 See also — labs in this chapter . . . 1806
15.25 Subgroup Meta-Analysis . . . 1806
15.25.1 Introduction . . . 1806
15.25.2 Prerequisites . . . 1806
15.25.3 Theory . . . 1806
15.25.4 Assumptions . . . 1807
15.25.5 R Implementation . . . 1807
15.25.6 Output & Results . . . 1807
15.25.7 Interpretation . . . 1808
15.25.8 Practical Tips . . . 1808
15.25.9 R Packages Used . . . 1808
15.25.10 For Reviewers . . . 1808
15.25.11 See also — labs in this chapter . . . 1809
15.26 Tau-Squared Estimation . . . 1809
15.26.1 Introduction . . . 1809
15.26.2 Prerequisites . . . 1809
15.26.3 Theory . . . 1809
15.26.4 Assumptions . . . 1809
15.26.5 R Implementation . . . 1809
15.26.6 Output & Results . . . 1810
15.26.7 Interpretation . . . 1810
15.26.8 Practical Tips . . . 1810
15.26.9 For Reviewers . . . 1810
15.26.10 See also — labs in this chapter . . . 1810
15.27 Trim-and-Fill . . . 1811
15.27.1 Introduction . . . 1811
15.27.2 Prerequisites . . . 1811
15.27.3 Theory . . . 1811
15.27.4 Assumptions . . . 1811
15.27.5 R Implementation . . . 1812

15.27.6 Output & Results . . . 1812
15.27.7 Interpretation . . . 1812
15.27.8 Practical Tips . . . 1813
15.27.9 R Packages Used . . . 1813
15.27.10 For Reviewers . . . 1813
15.27.11 See also — labs in this chapter . . . 1813
15.28 Week 4, Session 2 — Meta-analysis basics . . . 1814
15.28.1 Learning objectives . . . 1814
15.28.2 Prerequisites . . . 1814
15.28.3 Background . . . 1814
15.28.4 Setup . . . 1814
15.28.5 1. Hypothesis . . . 1814
15.28.6 2. Visualise — use a built-in dataset . . . 1815
15.28.7 3. Assumptions . . . 1815
15.28.8 4. Conduct . . . 1815
15.28.9 5. Concluding statement . . . 1815
15.28.10 Common pitfalls . . . 1815
15.28.11 Further reading . . . 1816
15.28.12 Session info . . . 1816
15.28.13 See also — chapter index . . . 1816
15.29 Week 4, Session 3 — Network meta-analysis . . . 1816
15.29.1 Learning objectives . . . 1816
15.29.2 Prerequisites . . . 1816
15.29.3 Background . . . 1816
15.29.4 Setup . . . 1817
15.29.5 1. Hypothesis . . . 1817
15.29.6 2. Visualise . . . 1817
15.29.7 3. Assumptions . . . 1817
15.29.8 4. Conduct . . . 1817
15.29.9 5. Concluding statement . . . 1817
15.29.10 Common pitfalls . . . 1818
15.29.11 Further reading . . . 1818
15.29.12 Session info . . . 1818
15.29.13 See also — chapter index . . . 1818
15.30 Week 4, Session 1 — Systematic reviews and PRISMA . . . 1818
15.30.1 Learning objectives . . . 1818
15.30.2 Prerequisites . . . 1818
15.30.3 Background . . . 1819
15.30.4 Setup . . . 1819
15.30.5 1. Goal . . . 1819
15.30.6 2. Approach . . . 1819
15.30.7 3. Execution — simulated screening counts . . . 1819
15.30.8 4. Check . . . 1820
15.30.9 5. Report . . . 1820
15.30.10 Common pitfalls . . . 1820
15.30.11 Further reading . . . 1820

15.30.12 Session info 1820
15.30.13 See also — chapter index 1821

16 Experimental Design **1823**
16.1 Method pages 1823
16.2 Labs 1824
16.3 Balanced Incomplete Block Designs 1824
16.3.1 Introduction 1824
16.3.2 Prerequisites 1825
16.3.3 Theory 1825
16.3.4 Assumptions 1825
16.3.5 R Implementation 1825
16.3.6 Output & Results 1826
16.3.7 Interpretation 1826
16.3.8 Practical Tips 1826
16.3.9 R Packages Used 1827
16.3.10 For Reviewers 1827
16.3.11 See also — labs in this chapter 1827
16.4 Principles of Blocking 1828
16.4.1 Introduction 1828
16.4.2 Prerequisites 1828
16.4.3 Theory 1828
16.4.4 Assumptions 1828
16.4.5 R Implementation 1829
16.4.6 Output & Results 1829
16.4.7 Interpretation 1829
16.4.8 Practical Tips 1829
16.4.9 R Packages Used 1830
16.4.10 For Reviewers 1830
16.4.11 See also — labs in this chapter 1830
16.5 Box-Behnken Designs 1831
16.5.1 Introduction 1831
16.5.2 Prerequisites 1831
16.5.3 Theory 1831
16.5.4 Assumptions 1831
16.5.5 R Implementation 1831
16.5.6 Output & Results 1832
16.5.7 Interpretation 1832
16.5.8 Practical Tips 1832
16.5.9 R Packages Used 1833
16.5.10 For Reviewers 1833
16.5.11 See also — labs in this chapter 1833
16.6 Central Composite Designs 1833
16.6.1 Introduction 1833
16.6.2 Prerequisites 1834
16.6.3 Theory 1834

16.6.4 Assumptions . . . 1834
16.6.5 R Implementation . . . 1834
16.6.6 Output & Results . . . 1835
16.6.7 Interpretation . . . 1835
16.6.8 Practical Tips . . . 1835
16.6.9 R Packages Used . . . 1836
16.6.10 For Reviewers . . . 1836
16.6.11 See also — labs in this chapter . . . 1836
16.7 Completely Randomised Design . . . 1836
16.7.1 Introduction . . . 1836
16.7.2 Prerequisites . . . 1837
16.7.3 Theory . . . 1837
16.7.4 Assumptions . . . 1837
16.7.5 R Implementation . . . 1837
16.7.6 Output & Results . . . 1838
16.7.7 Interpretation . . . 1838
16.7.8 Practical Tips . . . 1838
16.7.9 R Packages Used . . . 1839
16.7.10 For Reviewers . . . 1839
16.7.11 See also — labs in this chapter . . . 1839
16.8 Constrained Mixture Designs . . . 1839
16.8.1 Introduction . . . 1839
16.8.2 Prerequisites . . . 1840
16.8.3 Theory . . . 1840
16.8.4 Assumptions . . . 1840
16.8.5 R Implementation . . . 1840
16.8.6 Output & Results . . . 1841
16.8.7 Interpretation . . . 1841
16.8.8 Practical Tips . . . 1841
16.8.9 R Packages Used . . . 1842
16.8.10 For Reviewers . . . 1842
16.8.11 See also — labs in this chapter . . . 1842
16.9 Crossover Designs . . . 1842
16.9.1 Introduction . . . 1842
16.9.2 Prerequisites . . . 1843
16.9.3 Theory . . . 1843
16.9.4 Assumptions . . . 1843
16.9.5 R Implementation . . . 1843
16.9.6 Output & Results . . . 1844
16.9.7 Interpretation . . . 1844
16.9.8 Practical Tips . . . 1844
16.9.9 R Packages Used . . . 1845
16.9.10 For Reviewers . . . 1845
16.9.11 See also — labs in this chapter . . . 1845
16.10 Desirability Functions . . . 1845
16.10.1 Introduction . . . 1845

16.10.2 Prerequisites . . . 1846
16.10.3 Theory . . . 1846
16.10.4 Assumptions . . . 1846
16.10.5 R Implementation . . . 1846
16.10.6 Output & Results . . . 1847
16.10.7 Interpretation . . . 1847
16.10.8 Practical Tips . . . 1847
16.10.9 R Packages Used . . . 1848
16.10.10 For Reviewers . . . 1848
16.10.11 See also — labs in this chapter . . . 1848
16.11 2^k Factorial Designs . . . 1849
16.11.1 Introduction . . . 1849
16.11.2 Prerequisites . . . 1849
16.11.3 Theory . . . 1849
16.11.4 Assumptions . . . 1849
16.11.5 R Implementation . . . 1850
16.11.6 Output & Results . . . 1850
16.11.7 Interpretation . . . 1850
16.11.8 Practical Tips . . . 1850
16.11.9 R Packages Used . . . 1851
16.11.10 For Reviewers . . . 1851
16.11.11 See also — labs in this chapter . . . 1851
16.12 3^k Factorial Designs . . . 1852
16.12.1 Introduction . . . 1852
16.12.2 Prerequisites . . . 1852
16.12.3 Theory . . . 1852
16.12.4 Assumptions . . . 1852
16.12.5 R Implementation . . . 1852
16.12.6 Output & Results . . . 1853
16.12.7 Interpretation . . . 1853
16.12.8 Practical Tips . . . 1853
16.12.9 R Packages Used . . . 1854
16.12.10 For Reviewers . . . 1854
16.12.11 See also — labs in this chapter . . . 1854
16.13 Fractional Factorial Designs . . . 1854
16.13.1 Introduction . . . 1854
16.13.2 Prerequisites . . . 1855
16.13.3 Theory . . . 1855
16.13.4 Assumptions . . . 1855
16.13.5 R Implementation . . . 1855
16.13.6 Output & Results . . . 1856
16.13.7 Interpretation . . . 1856
16.13.8 Practical Tips . . . 1856
16.13.9 R Packages Used . . . 1856
16.13.10 For Reviewers . . . 1857
16.13.11 See also — labs in this chapter . . . 1857

16.14 Graeco-Latin Square Designs 1857
16.14.1 Introduction 1857
16.14.2 Prerequisites 1858
16.14.3 Theory 1858
16.14.4 Assumptions 1858
16.14.5 R Implementation 1858
16.14.6 Output & Results 1858
16.14.7 Interpretation 1859
16.14.8 Practical Tips 1859
16.14.9 R Packages Used 1859
16.14.10 For Reviewers 1860
16.14.11 See also — labs in this chapter 1860
16.15 Latin Square Designs 1860
16.15.1 Introduction 1860
16.15.2 Prerequisites 1860
16.15.3 Theory 1861
16.15.4 Assumptions 1861
16.15.5 R Implementation 1861
16.15.6 Output & Results 1861
16.15.7 Interpretation 1862
16.15.8 Practical Tips 1862
16.15.9 R Packages Used 1862
16.15.10 For Reviewers 1862
16.15.11 See also — labs in this chapter 1863
16.16 Simplex Centroid Designs 1863
16.16.1 Introduction 1863
16.16.2 Prerequisites 1863
16.16.3 Theory 1863
16.16.4 Assumptions 1864
16.16.5 R Implementation 1864
16.16.6 Output & Results 1864
16.16.7 Interpretation 1864
16.16.8 Practical Tips 1865
16.16.9 R Packages Used 1865
16.16.10 For Reviewers 1865
16.16.11 See also — labs in this chapter 1865
16.17 Simplex Lattice Mixture Designs 1866
16.17.1 Introduction 1866
16.17.2 Prerequisites 1866
16.17.3 Theory 1866
16.17.4 Assumptions 1867
16.17.5 R Implementation 1867
16.17.6 Output & Results 1867
16.17.7 Interpretation 1867
16.17.8 Practical Tips 1868
16.17.9 R Packages Used 1868

16.17.10 For Reviewers . . . 1868
16.17.11 See also — labs in this chapter . . . 1868
16.18 Optimal Designs: D, A, I . . . 1869
16.18.1 Introduction . . . 1869
16.18.2 Prerequisites . . . 1869
16.18.3 Theory . . . 1869
16.18.4 Assumptions . . . 1869
16.18.5 R Implementation . . . 1869
16.18.6 Output & Results . . . 1870
16.18.7 Interpretation . . . 1870
16.18.8 Practical Tips . . . 1870
16.18.9 For Reviewers . . . 1870
16.18.10 See also — labs in this chapter . . . 1871
16.19 Orthogonal Arrays . . . 1871
16.19.1 Introduction . . . 1871
16.19.2 Prerequisites . . . 1871
16.19.3 Theory . . . 1871
16.19.4 Assumptions . . . 1872
16.19.5 R Implementation . . . 1872
16.19.6 Output & Results . . . 1872
16.19.7 Interpretation . . . 1872
16.19.8 Practical Tips . . . 1873
16.19.9 R Packages Used . . . 1873
16.19.10 For Reviewers . . . 1873
16.19.11 See also — labs in this chapter . . . 1873
16.20 Plackett-Burman Designs . . . 1874
16.20.1 Introduction . . . 1874
16.20.2 Prerequisites . . . 1874
16.20.3 Theory . . . 1874
16.20.4 Assumptions . . . 1875
16.20.5 R Implementation . . . 1875
16.20.6 Output & Results . . . 1875
16.20.7 Interpretation . . . 1875
16.20.8 Practical Tips . . . 1875
16.20.9 R Packages Used . . . 1876
16.20.10 For Reviewers . . . 1876
16.20.11 See also — labs in this chapter . . . 1876
16.21 Power Analysis for DOE . . . 1877
16.21.1 Introduction . . . 1877
16.21.2 Prerequisites . . . 1877
16.21.3 Theory . . . 1877
16.21.4 Assumptions . . . 1877
16.21.5 R Implementation . . . 1878
16.21.6 Output & Results . . . 1878
16.21.7 Interpretation . . . 1878
16.21.8 Practical Tips . . . 1879

16.21.9 R Packages Used . . . 1879
16.21.10 For Reviewers . . . 1879
16.21.11 See also — labs in this chapter . . . 1879
16.22 Randomised Complete Block Design . . . 1880
16.22.1 Introduction . . . 1880
16.22.2 Prerequisites . . . 1880
16.22.3 Theory . . . 1880
16.22.4 Assumptions . . . 1881
16.22.5 R Implementation . . . 1881
16.22.6 Output & Results . . . 1881
16.22.7 Interpretation . . . 1882
16.22.8 Practical Tips . . . 1882
16.22.9 For Reviewers . . . 1882
16.22.10 See also — labs in this chapter . . . 1882
16.23 Randomisation in Design . . . 1883
16.23.1 Introduction . . . 1883
16.23.2 Prerequisites . . . 1883
16.23.3 Theory . . . 1883
16.23.4 Assumptions . . . 1883
16.23.5 R Implementation . . . 1883
16.23.6 Output & Results . . . 1884
16.23.7 Interpretation . . . 1884
16.23.8 Practical Tips . . . 1884
16.23.9 For Reviewers . . . 1884
16.23.10 See also — labs in this chapter . . . 1885
16.24 Repeated-Measures Designs . . . 1885
16.24.1 Introduction . . . 1885
16.24.2 Prerequisites . . . 1885
16.24.3 Theory . . . 1885
16.24.4 Assumptions . . . 1886
16.24.5 R Implementation . . . 1886
16.24.6 Output & Results . . . 1886
16.24.7 Interpretation . . . 1887
16.24.8 Practical Tips . . . 1887
16.24.9 R Packages Used . . . 1887
16.24.10 For Reviewers . . . 1888
16.24.11 See also — labs in this chapter . . . 1888
16.25 Resolution and Aliasing . . . 1888
16.25.1 Introduction . . . 1888
16.25.2 Prerequisites . . . 1888
16.25.3 Theory . . . 1889
16.25.4 Assumptions . . . 1889
16.25.5 R Implementation . . . 1889
16.25.6 Output & Results . . . 1889
16.25.7 Interpretation . . . 1890
16.25.8 Practical Tips . . . 1890

16.25.9 R Packages Used . . . 1890
16.25.10 For Reviewers . . . 1890
16.25.11 See also — labs in this chapter . . . 1891
16.26 Response Surface Methodology . . . 1891
16.26.1 Introduction . . . 1891
16.26.2 Prerequisites . . . 1891
16.26.3 Theory . . . 1891
16.26.4 Assumptions . . . 1892
16.26.5 R Implementation . . . 1892
16.26.6 Output & Results . . . 1892
16.26.7 Interpretation . . . 1893
16.26.8 Practical Tips . . . 1893
16.26.9 R Packages Used . . . 1893
16.26.10 For Reviewers . . . 1894
16.26.11 See also — labs in this chapter . . . 1894
16.27 Robust Parameter Design . . . 1894
16.27.1 Introduction . . . 1894
16.27.2 Prerequisites . . . 1894
16.27.3 Theory . . . 1894
16.27.4 Assumptions . . . 1895
16.27.5 R Implementation . . . 1895
16.27.6 Output & Results . . . 1895
16.27.7 Interpretation . . . 1895
16.27.8 Practical Tips . . . 1896
16.27.9 For Reviewers . . . 1896
16.27.10 See also — labs in this chapter . . . 1896
16.28 Signal-to-Noise Ratios . . . 1896
16.28.1 Introduction . . . 1896
16.28.2 Prerequisites . . . 1897
16.28.3 Theory . . . 1897
16.28.4 Assumptions . . . 1897
16.28.5 R Implementation . . . 1897
16.28.6 Output & Results . . . 1897
16.28.7 Interpretation . . . 1897
16.28.8 Practical Tips . . . 1898
16.28.9 For Reviewers . . . 1898
16.28.10 See also — labs in this chapter . . . 1898
16.29 Split-Plot Designs . . . 1898
16.29.1 Introduction . . . 1898
16.29.2 Prerequisites . . . 1899
16.29.3 Theory . . . 1899
16.29.4 Assumptions . . . 1899
16.29.5 R Implementation . . . 1899
16.29.6 Output & Results . . . 1900
16.29.7 Interpretation . . . 1900
16.29.8 Practical Tips . . . 1900

16.29.9 R Packages Used . . . 1901
16.29.10 For Reviewers . . . 1901
16.29.11 See also — labs in this chapter . . . 1901
16.30 Method of Steepest Ascent . . . 1901
16.30.1 Introduction . . . 1901
16.30.2 Prerequisites . . . 1902
16.30.3 Theory . . . 1902
16.30.4 Assumptions . . . 1902
16.30.5 R Implementation . . . 1902
16.30.6 Output & Results . . . 1903
16.30.7 Interpretation . . . 1903
16.30.8 Practical Tips . . . 1903
16.30.9 R Packages Used . . . 1904
16.30.10 For Reviewers . . . 1904
16.30.11 See also — labs in this chapter . . . 1904
16.31 Strip-Plot Designs . . . 1904
16.31.1 Introduction . . . 1904
16.31.2 Prerequisites . . . 1905
16.31.3 Theory . . . 1905
16.31.4 Assumptions . . . 1905
16.31.5 R Implementation . . . 1905
16.31.6 Output & Results . . . 1906
16.31.7 Interpretation . . . 1906
16.31.8 Practical Tips . . . 1906
16.31.9 R Packages Used . . . 1907
16.31.10 For Reviewers . . . 1907
16.31.11 See also — labs in this chapter . . . 1907
16.32 Taguchi Methods . . . 1908
16.32.1 Introduction . . . 1908
16.32.2 Prerequisites . . . 1908
16.32.3 Theory . . . 1908
16.32.4 Assumptions . . . 1908
16.32.5 R Implementation . . . 1908
16.32.6 Output & Results . . . 1909
16.32.7 Interpretation . . . 1909
16.32.8 Practical Tips . . . 1909
16.32.9 For Reviewers . . . 1909
16.32.10 See also — labs in this chapter . . . 1909
16.33 Week 1, Session 3 — Adaptive, non-inferiority, equivalence trials . . . 1910
16.33.1 Learning objectives . . . 1910
16.33.2 Prerequisites . . . 1910
16.33.3 Background . . . 1910
16.33.4 Setup . . . 1911
16.33.5 1. Hypothesis . . . 1911
16.33.6 2. Visualise . . . 1911

16.33.7 3. Assumptions . . . 1911
16.33.8 4. Conduct . . . 1911
16.33.9 5. Concluding statement . . . 1912
16.33.10 Common pitfalls . . . 1912
16.33.11 Further reading . . . 1912
16.33.12 Session info . . . 1912
16.33.13 See also — chapter index . . . 1913
16.34 Week 1, Session 4 — Bench and translational design . . . 1913
16.34.1 Learning objectives . . . 1913
16.34.2 Prerequisites . . . 1913
16.34.3 Background . . . 1913
16.34.4 Setup . . . 1914
16.34.5 1. Hypothesis . . . 1914
16.34.6 2. Visualise . . . 1914
16.34.7 3. Assumptions . . . 1914
16.34.8 4. Conduct . . . 1915
16.34.9 5. Concluding statement . . . 1915
16.34.10 Common pitfalls . . . 1915
16.34.11 Further reading . . . 1915
16.34.12 Session info . . . 1916
16.34.13 See also — chapter index . . . 1916
16.35 Week 3, Session 3 — DAGs with dagitty and ggdag . . . 1916
16.35.1 Learning objectives . . . 1916
16.35.2 Prerequisites . . . 1916
16.35.3 Background . . . 1916
16.35.4 Setup . . . 1917
16.35.5 1. Goal . . . 1917
16.35.6 2. Approach . . . 1917
16.35.7 3. Execution . . . 1918
16.35.8 4. Check . . . 1918
16.35.9 5. Report . . . 1918
16.35.10 Common pitfalls . . . 1919
16.35.11 Further reading . . . 1919
16.35.12 Session info . . . 1919
16.35.13 See also — chapter index . . . 1919
16.36 Week 3, Session 5 — G-methods, IV, DiD, RDD; heterogeneity of treatment effect . . . 1919
16.36.1 Learning objectives . . . 1919
16.36.2 Prerequisites . . . 1919
16.36.3 Background . . . 1920
16.36.4 Setup . . . 1920
16.36.5 1. Goal . . . 1920
16.36.6 2. Approach . . . 1920
16.36.7 3. Execution . . . 1920
16.36.8 4. Check . . . 1921
16.36.9 5. Report . . . 1921

16.36.10 Common pitfalls . . . 1922
16.36.11 Further reading . . . 1922
16.36.12 Session info . . . 1922
16.36.13 See also — chapter index . . . 1922
16.37 Week 2, Session 4 — GLMMs and GEE . . . 1922
16.37.1 Learning objectives . . . 1922
16.37.2 Prerequisites . . . 1922
16.37.3 Background . . . 1922
16.37.4 Setup . . . 1923
16.37.5 1. Hypothesis . . . 1923
16.37.6 2. Visualise . . . 1923
16.37.7 3. Assumptions . . . 1924
16.37.8 4. Conduct . . . 1924
16.37.9 5. Concluding statement . . . 1924
16.37.10 Common pitfalls . . . 1924
16.37.11 Further reading . . . 1925
16.37.12 Session info . . . 1925
16.37.13 See also — chapter index . . . 1925
16.38 Week 2, Session 3 — Linear mixed models with lme4 . . . 1925
16.38.1 Learning objectives . . . 1925
16.38.2 Prerequisites . . . 1925
16.38.3 Background . . . 1925
16.38.4 Setup . . . 1926
16.38.5 1. Hypothesis . . . 1926
16.38.6 2. Visualise . . . 1926
16.38.7 3. Assumptions . . . 1926
16.38.8 4. Conduct . . . 1927
16.38.9 5. Concluding statement . . . 1927
16.38.10 Common pitfalls . . . 1927
16.38.11 Further reading . . . 1927
16.38.12 Session info . . . 1928
16.38.13 See also — chapter index . . . 1928
16.39 Week 2, Session 1 — MCAR, MAR, MNAR . . . 1928
16.39.1 Learning objectives . . . 1928
16.39.2 Prerequisites . . . 1928
16.39.3 Background . . . 1928
16.39.4 Setup . . . 1929
16.39.5 1. Hypothesis . . . 1929
16.39.6 2. Visualise . . . 1929
16.39.7 3. Assumptions . . . 1930
16.39.8 4. Conduct . . . 1930
16.39.9 5. Concluding statement . . . 1930
16.39.10 Common pitfalls . . . 1930
16.39.11 Further reading . . . 1930
16.39.12 Session info . . . 1931
16.39.13 See also — chapter index . . . 1931

16.40 Week 2, Session 2 — Multiple imputation with mice 1931
16.40.1 Learning objectives 1931
16.40.2 Prerequisites 1931
16.40.3 Background 1931
16.40.4 Setup 1932
16.40.5 1. Hypothesis 1932
16.40.6 2. Visualise 1932
16.40.7 3. Assumptions 1932
16.40.8 4. Conduct 1932
16.40.9 5. Concluding statement 1933
16.40.10 Common pitfalls 1933
16.40.11 Further reading 1933
16.40.12 Session info 1933
16.40.13 See also — chapter index 1933
16.41 Week 1, Session 1 — Observational designs and STROBE 1933
16.41.1 Learning objectives 1934
16.41.2 Prerequisites 1934
16.41.3 Background 1934
16.41.4 Setup 1934
16.41.5 1. Goal 1934
16.41.6 2. Approach 1935
16.41.7 3. Execution 1935
16.41.8 4. Check 1936
16.41.9 5. Report 1936
16.41.10 Common pitfalls 1936
16.41.11 Further reading 1936
16.41.12 Session info 1937
16.41.13 See also — chapter index 1937
16.42 Week 4, Session 5 — Pre-registration and statistical analysis plans 1937
16.42.1 Learning objectives 1937
16.42.2 Prerequisites 1937
16.42.3 Background 1937
16.42.4 Setup 1938
16.42.5 1. Goal 1938
16.42.6 2. Approach 1938
16.42.7 3. Execution — template 1938
16.42.8 4. Check 1939
16.42.9 5. Report 1939
16.42.10 Common pitfalls 1939
16.42.11 Further reading 1939
16.42.12 Session info 1939
16.42.13 See also — chapter index 1940
16.43 Week 3, Session 4 — Propensity scores and IPTW 1940
16.43.1 Learning objectives 1940
16.43.2 Prerequisites 1940

16.43.3 Background . . . 1940
16.43.4 Setup . . . 1941
16.43.5 1. Hypothesis . . . 1941
16.43.6 2. Visualise . . . 1941
16.43.7 3. Assumptions . . . 1941
16.43.8 4. Conduct . . . 1941
16.43.9 5. Concluding statement . . . 1942
16.43.10 Common pitfalls . . . 1942
16.43.11 Further reading . . . 1942
16.43.12 Session info . . . 1942
16.43.13 See also — chapter index . . . 1943
16.44 Week 1, Session 2 — Randomised controlled trials . . . 1943
16.44.1 Learning objectives . . . 1943
16.44.2 Prerequisites . . . 1943
16.44.3 Background . . . 1943
16.44.4 Setup . . . 1944
16.44.5 1. Hypothesis . . . 1944
16.44.6 2. Visualise . . . 1944
16.44.7 3. Assumptions . . . 1944
16.44.8 4. Conduct . . . 1945
16.44.9 5. Concluding statement . . . 1945
16.44.10 Common pitfalls . . . 1945
16.44.11 Further reading . . . 1945
16.44.12 Session info . . . 1945
16.44.13 See also — chapter index . . . 1945
16.45 Week 4, Session 4 — SIR and SEIR models with deSolve . . . 1946
16.45.1 Learning objectives . . . 1946
16.45.2 Prerequisites . . . 1946
16.45.3 Background . . . 1946
16.45.4 Setup . . . 1946
16.45.5 1. Goal . . . 1946
16.45.6 2. Approach . . . 1947
16.45.7 3. Execution . . . 1947
16.45.8 4. Check . . . 1947
16.45.9 5. Report . . . 1948
16.45.10 Common pitfalls . . . 1948
16.45.11 Further reading . . . 1948
16.45.12 Session info . . . 1948
16.45.13 See also — chapter index . . . 1948

17 Reporting Templates 1949

18 STROBE — observational studies 1951
18.0.1 2^k Factorial Designs . . . 1951
18.0.2 3^k Factorial Designs . . . 1952
18.0.3 Accelerated Failure Time Models . . . 1952

18.0.4	Added-Variable Plots	1952
18.0.5	Anderson-Darling Test	1953
18.0.6	Balanced Incomplete Block Designs	1953
18.0.7	Bartlett's Test	1953
18.0.8	Benjamini-Hochberg FDR	1954
18.0.9	Benjamini-Yekutieli FDR	1954
18.0.10	Best Subset Selection	1955
18.0.11	Beta Regression	1955
18.0.12	Bonferroni Correction	1955
18.0.13	Bootstrap Confidence Intervals	1956
18.0.14	Box-Behnken Designs	1956
18.0.15	Censoring Types	1956
18.0.16	Central Composite Designs	1957
18.0.17	Centring and Scaling Predictors	1957
18.0.18	Checking Cox Assumptions	1958
18.0.19	Chi-Squared Goodness-of-Fit Test	1958
18.0.20	Chi-Squared Test of Independence	1958
18.0.21	Cochran-Mantel-Haenszel Test	1959
18.0.22	Competing Risks: Cumulative Incidence	1959
18.0.23	Completely Randomised Design	1959
18.0.24	Concordance and the C-Index	1960
18.0.25	Conditional Survival	1960
18.0.26	Constrained Mixture Designs	1961
18.0.27	Contrasts in R	1961
18.0.28	Cook's Distance	1961
18.0.29	Cox Proportional Hazards Regression	1962
18.0.30	Cox Regression as Regression	1962
18.0.31	Crossover Designs	1962
18.0.32	Desirability Functions	1963
18.0.33	Dirichlet Regression	1963
18.0.34	Dummy Coding	1964
18.0.35	Effect (Deviation) Coding	1964
18.0.36	Effect Size: Cohen's d	1964
18.0.37	Effect Size: Cohen's h	1965
18.0.38	Effect Size: Eta-Squared	1965
18.0.39	Effect Sizes: Overview	1965
18.0.40	Elastic Net	1966
18.0.41	Equivalence Testing with TOST	1966
18.0.42	Fine-Gray Subdistribution Hazards	1966
18.0.43	Fisher's Exact Test	1967
18.0.44	Fractional Factorial Designs	1967
18.0.45	Frailty Models	1968
18.0.46	Friedman Test	1968
18.0.47	GAMMs: Additive Mixed Models	1968
18.0.48	GAMs: Introduction	1969
18.0.49	Generalised Estimating Equations	1969

18.0.50	glmer: Generalised Mixed Models	1969
18.0.51	Graeco-Latin Square Designs	1970
18.0.52	Hazard, Survival, and Cumulative Hazard	1970
18.0.53	Holm's Step-Down Correction	1970
18.0.54	Hurdle Models	1971
18.0.55	Interactions in Regression	1971
18.0.56	Interval-Censored Data	1972
18.0.57	Joint Longitudinal-Survival Models	1972
18.0.58	Kaplan-Meier Estimation	1972
18.0.59	Kendall's Tau	1973
18.0.60	Kolmogorov-Smirnov Test	1973
18.0.61	Kruskal-Wallis Test	1973
18.0.62	Landmark Analysis	1974
18.0.63	Lasso Regression	1974
18.0.64	Latin Square Designs	1974
18.0.65	Left Truncation	1975
18.0.66	Levene's Test of Variances	1975
18.0.67	Leverage and Influence	1976
18.0.68	Linear Regression Assumptions	1976
18.0.69	Log-Rank Test: Details	1976
18.0.70	Logistic Regression	1977
18.0.71	Logistic Regression Diagnostics	1977
18.0.72	Mann-Whitney U Test	1977
18.0.73	McNemar's Test	1978
18.0.74	Method of Steepest Ascent	1978
18.0.75	Minimum Detectable Effect	1979
18.0.76	Mixed ANOVA	1979
18.0.77	Mixed-Effects Models: Introduction	1979
18.0.78	Model Selection with AIC and BIC	1980
18.0.79	Multi-State Models	1980
18.0.80	Multicollinearity and VIF	1980
18.0.81	Multinomial Logistic Regression	1981
18.0.82	Multiple Comparisons: Overview	1981
18.0.83	Multiple Linear Regression	1982
18.0.84	Negative Binomial Regression	1982
18.0.85	Nested vs Crossed Random Effects	1982
18.0.86	Null and Alternative Hypotheses	1983
18.0.87	Offsets in Regression	1983
18.0.88	One-Proportion Test	1983
18.0.89	One-Sample t-Test	1984
18.0.90	One-Sample z-Test	1984
18.0.91	One-Way ANOVA	1985
18.0.92	Optimal Designs: D, A, I	1985
18.0.93	Ordinal Logistic Regression	1985
18.0.94	Orthogonal Arrays	1986
18.0.95	P-Values Explained	1986

18.0.96	Paired t-Test	1986
18.0.97	Parametric Survival: Log-Normal	1987
18.0.98	Parametric Survival: Weibull	1987
18.0.99	Partial Correlation	1987
18.0.100	Pearson Correlation Test	1988
18.0.101	Permutation Tests	1988
18.0.102	Plackett-Burman Designs	1989
18.0.103	Poisson Regression	1989
18.0.104	Polynomial Regression	1989
18.0.105	Post-Hoc Power: A Controversy	1990
18.0.106	Post-Hoc Tests with Tukey HSD	1990
18.0.107	Power Analysis for DOE	1991
18.0.108	Power Analysis: Introduction	1991
18.0.109	Power for Agreement (Kappa)	1991
18.0.110	Power for Bland-Altman Studies	1992
18.0.111	Power for Chi-Squared Tests	1992
18.0.112	Power for Cluster-RCT	1992
18.0.113	Power for Correlation Tests	1993
18.0.114	Power for Cox Regression	1993
18.0.115	Power for Crossover Trials	1994
18.0.116	Power for Diagnostic Accuracy	1994
18.0.117	Power for Equivalence (TOST)	1994
18.0.118	Power for ICC	1995
18.0.119	Power for Linear Regression	1995
18.0.120	Power for Logistic Regression	1995
18.0.121	Power for McNemar's Test	1996
18.0.122	Power for Non-Inferiority Trials	1996
18.0.123	Power for One-Proportion Test	1997
18.0.124	Power for One-Sample t-Test	1997
18.0.125	Power for One-Way ANOVA	1997
18.0.126	Power for Paired t-Test	1998
18.0.127	Power for Repeated-Measures ANOVA	1998
18.0.128	Power for Stepped-Wedge Trials	1998
18.0.129	Power for the Log-Rank Test	1999
18.0.130	Power for Two-Proportion Test	1999
18.0.131	Principles of Blocking	2000
18.0.132	Probit Regression	2000
18.0.133	Quantile Regression	2000
18.0.134	Quasi-Poisson Regression	2001
18.0.135	R-Squared and Adjusted R-Squared	2001
18.0.136	Random-Intercept Models	2001
18.0.137	Random-Slope Models	2002
18.0.138	Randomisation in Design	2002
18.0.139	Randomised Complete Block Design	2003
18.0.140	Recurrent Events	2003
18.0.141	Repeated-Measures ANOVA	2003

18.0.142	Repeated-Measures Designs	2004
18.0.143	Residual Diagnostics	2004
18.0.144	Resolution and Aliasing	2004
18.0.145	Response Surface Methodology	2005
18.0.146	Restricted Mean Survival Time (RMST)	2005
18.0.147	Ridge Regression	2006
18.0.148	Robust Parameter Design	2006
18.0.149	Robust Regression	2006
18.0.150	ROC and AUC for Logistic Models	2007
18.0.151	Royston-Parmar Flexible Parametric Models	2007
18.0.152	Sample Size for a Two-Sample t-Test	2007
18.0.153	Sample Size Sensitivity Analysis	2008
18.0.154	Schoenfeld Residuals	2008
18.0.155	Shapiro-Wilk Normality Test	2009
18.0.156	Signal-to-Noise Ratios	2009
18.0.157	Simple Linear Regression	2009
18.0.158	Simplex Centroid Designs	2010
18.0.159	Simplex Lattice Mixture Designs	2010
18.0.160	Simulating Survival Data	2010
18.0.161	Spearman Rank Correlation Test	2011
18.0.162	Spline Regression	2011
18.0.163	Split-Plot Designs	2012
18.0.164	Stepwise Regression: Pitfalls	2012
18.0.165	Stratified Cox Models	2012
18.0.166	Strip-Plot Designs	2013
18.0.167	Taguchi Methods	2013
18.0.168	The Bootstrap: Introduction	2014
18.0.169	The Brier Score for Survival	2014
18.0.170	The Jackknife	2014
18.0.171	The Nelson-Aalen Estimator	2015
18.0.172	The Runs Test	2015
18.0.173	The Sign Test	2015
18.0.174	Time-Dependent ROC	2016
18.0.175	Time-Varying Covariates	2016
18.0.176	Tobit Regression	2016
18.0.177	Truncated Regression	2017
18.0.178	Two-Proportion Test	2017
18.0.179	Two-Sample t-Test	2018
18.0.180	Two-Way ANOVA	2018
18.0.181	Type I and Type II Errors	2018
18.0.182	Weighted Log-Rank Tests	2019
18.0.183	Welch's t-Test	2019
18.0.184	Wilcoxon Signed-Rank Test	2019
18.0.185	Worked lme4::lmer Examples	2020
18.0.186	Zero-Inflated Models	2020

19 CONSORT — randomised trials **2021**
19.0.1 2^k Factorial Designs . . . 2021
19.0.2 3^k Factorial Designs . . . 2022
19.0.3 Adaptive Trial Designs . . . 2022
19.0.4 Alpha-Spending Functions . . . 2022
19.0.5 Anderson-Darling Test . . . 2023
19.0.6 Balanced Incomplete Block Designs . . . 2023
19.0.7 Bartlett's Test . . . 2023
19.0.8 Baseline Adjustment with ANCOVA . . . 2024
19.0.9 Benjamini-Hochberg FDR . . . 2024
19.0.10 Benjamini-Yekutieli FDR . . . 2025
19.0.11 Bland-Altman Limits of Agreement . . . 2025
19.0.12 Blinding Procedures . . . 2025
19.0.13 Block Randomisation . . . 2026
19.0.14 Bonferroni Correction . . . 2026
19.0.15 Bootstrap Confidence Intervals . . . 2027
19.0.16 Box-Behnken Designs . . . 2027
19.0.17 Central Composite Designs . . . 2027
19.0.18 Chi-Squared Goodness-of-Fit Test . . . 2028
19.0.19 Chi-Squared Test of Independence . . . 2028
19.0.20 Clinical Equivalence Trials . . . 2028
19.0.21 Cluster-Randomised Trials . . . 2029
19.0.22 Cochran-Mantel-Haenszel Test . . . 2029
19.0.23 Cohen's Kappa . . . 2030
19.0.24 Completely Randomised Design . . . 2030
19.0.25 Conditional Power . . . 2030
19.0.26 Constrained Mixture Designs . . . 2031
19.0.27 Crossover Designs . . . 2031
19.0.28 Crossover RCT Design . . . 2031
19.0.29 Cutpoint Selection . . . 2032
19.0.30 Desirability Functions . . . 2032
19.0.31 Diagnostic Test Accuracy . . . 2033
19.0.32 Effect Size: Cohen's d . . . 2033
19.0.33 Effect Size: Cohen's h . . . 2033
19.0.34 Effect Size: Eta-Squared . . . 2034
19.0.35 Effect Sizes: Overview . . . 2034
19.0.36 Equivalence Testing with TOST . . . 2034
19.0.37 Factorial Trials . . . 2035
19.0.38 Fisher's Exact Test . . . 2035
19.0.39 Fractional Factorial Designs . . . 2036
19.0.40 Friedman Test . . . 2036
19.0.41 Graeco-Latin Square Designs . . . 2036
19.0.42 Holm's Step-Down Correction . . . 2037
19.0.43 Interim Analyses and Group Sequential Designs 2037
19.0.44 Intraclass Correlation Coefficient (ICC) . . . 2037
19.0.45 ITT vs Per-Protocol Analysis . . . 2038

19.0.46	Kendall's Tau	2038
19.0.47	Kolmogorov-Smirnov Test	2039
19.0.48	Kruskal-Wallis Test	2039
19.0.49	Latin Square Designs	2039
19.0.50	Levene's Test of Variances	2040
19.0.51	Likelihood Ratios	2040
19.0.52	Mann-Whitney U Test	2040
19.0.53	McNemar's Test	2041
19.0.54	Method of Steepest Ascent	2041
19.0.55	Minimisation Algorithm	2041
19.0.56	Minimum Detectable Effect	2042
19.0.57	Missing Data in RCTs	2042
19.0.58	Mixed ANOVA	2043
19.0.59	Multiple Comparisons: Overview	2043
19.0.60	Multiple Imputation	2043
19.0.61	Non-Inferiority Margin Selection	2044
19.0.62	Null and Alternative Hypotheses	2044
19.0.63	O'Brien-Fleming Boundary	2044
19.0.64	One-Proportion Test	2045
19.0.65	One-Sample t-Test	2045
19.0.66	One-Sample z-Test	2046
19.0.67	One-Way ANOVA	2046
19.0.68	Optimal Designs: D, A, I	2046
19.0.69	Orthogonal Arrays	2047
19.0.70	P-Values Explained	2047
19.0.71	Paired t-Test	2047
19.0.72	Parallel-Group RCT Design	2048
19.0.73	Pearson Correlation Test	2048
19.0.74	Permutation Tests	2049
19.0.75	Plackett-Burman Designs	2049
19.0.76	Pocock Boundary	2049
19.0.77	Post-Hoc Power: A Controversy	2050
19.0.78	Post-Hoc Tests with Tukey HSD	2050
19.0.79	Power Analysis for DOE	2050
19.0.80	Power Analysis: Introduction	2051
19.0.81	Power for Agreement (Kappa)	2051
19.0.82	Power for Bland-Altman Studies	2052
19.0.83	Power for Chi-Squared Tests	2052
19.0.84	Power for Cluster-RCT	2052
19.0.85	Power for Correlation Tests	2053
19.0.86	Power for Cox Regression	2053
19.0.87	Power for Crossover Trials	2054
19.0.88	Power for Diagnostic Accuracy	2054
19.0.89	Power for Equivalence (TOST)	2054
19.0.90	Power for ICC	2055
19.0.91	Power for Linear Regression	2055

19.0.92	Power for Logistic Regression	2055
19.0.93	Power for McNemar's Test	2056
19.0.94	Power for Non-Inferiority Trials	2056
19.0.95	Power for One-Proportion Test	2057
19.0.96	Power for One-Sample t-Test	2057
19.0.97	Power for One-Way ANOVA	2057
19.0.98	Power for Paired t-Test	2058
19.0.99	Power for Repeated-Measures ANOVA	2058
19.0.100	Power for Stepped-Wedge Trials	2058
19.0.101	Power for the Log-Rank Test	2059
19.0.102	Power for Two-Proportion Test	2059
19.0.103	Predictive Values and Prevalence	2060
19.0.104	Principles of Blocking	2060
19.0.105	Randomisation in Design	2060
19.0.106	Randomisation Methods	2061
19.0.107	Randomised Complete Block Design	2061
19.0.108	Reliability and Cronbach's Alpha	2061
19.0.109	Repeated-Measures ANOVA	2062
19.0.110	Repeated-Measures Designs	2062
19.0.111	Resolution and Aliasing	2063
19.0.112	Response Surface Methodology	2063
19.0.113	Robust Parameter Design	2063
19.0.114	ROC Analysis	2064
19.0.115	Sample Size for a Two-Sample t-Test	2064
19.0.116	Sample Size Re-Estimation	2065
19.0.117	Sample Size Sensitivity Analysis	2065
19.0.118	Sensitivity Analyses in Clinical Trials	2065
19.0.119	Shapiro-Wilk Normality Test	2066
19.0.120	Signal-to-Noise Ratios	2066
19.0.121	Simplex Centroid Designs	2066
19.0.122	Simplex Lattice Mixture Designs	2067
19.0.123	Spearman Rank Correlation Test	2067
19.0.124	Split-Plot Designs	2068
19.0.125	Stepped-Wedge Trial	2068
19.0.126	Stratified Randomisation	2068
19.0.127	Strip-Plot Designs	2069
19.0.128	Subgroup Analyses	2069
19.0.129	Subgroup Forest Plots	2070
19.0.130	Taguchi Methods	2070
19.0.131	The Bootstrap: Introduction	2070
19.0.132	The Jackknife	2071
19.0.133	The Runs Test	2071
19.0.134	The Sign Test	2071
19.0.135	Two-Proportion Test	2072
19.0.136	Two-Sample t-Test	2072
19.0.137	Two-Way ANOVA	2072

19.0.138 Type I and Type II Errors 2073
19.0.139 Weighted Kappa 2073
19.0.140 Welch's t-Test 2074
19.0.141 Wilcoxon Signed-Rank Test 2074

20 TRIPOD-AI — prediction models 2075
20.0.1 A Complete mlr3 Workflow 2075
20.0.2 A Complete tidymodels Workflow 2076
20.0.3 Accelerated Failure Time Models 2076
20.0.4 Adaptive Trial Designs 2076
20.0.5 Added-Variable Plots 2077
20.0.6 Alpha-Spending Functions 2077
20.0.7 Anomaly Detection 2077
20.0.8 Bagging 2078
20.0.9 Baseline Adjustment with ANCOVA 2078
20.0.10 Best Subset Selection 2079
20.0.11 Beta Regression 2079
20.0.12 Bland-Altman Limits of Agreement 2079
20.0.13 Blinding Procedures 2080
20.0.14 Block Randomisation 2080
20.0.15 Calibration Plots 2080
20.0.16 CatBoost 2081
20.0.17 Censoring Types 2081
20.0.18 Centring and Scaling Predictors 2081
20.0.19 Checking Cox Assumptions 2082
20.0.20 Class Weights for Imbalanced Data 2082
20.0.21 Clinical Equivalence Trials 2083
20.0.22 Cluster-Randomised Trials 2083
20.0.23 Cohen's Kappa 2083
20.0.24 Competing Risks: Cumulative Incidence 2084
20.0.25 Concordance and the C-Index 2084
20.0.26 Conditional Power 2084
20.0.27 Conditional Survival 2085
20.0.28 Contrasts in R 2085
20.0.29 Convolutional Neural Networks: Introduction . 2086
20.0.30 Cook's Distance 2086
20.0.31 Cox Proportional Hazards Regression 2086
20.0.32 Cox Regression as Regression 2087
20.0.33 Cross-Validation 2087
20.0.34 Crossover RCT Design 2087
20.0.35 Cutpoint Selection 2088
20.0.36 DBSCAN as an ML Tool 2088
20.0.37 Decision Trees (CART) 2089
20.0.38 Diagnostic Test Accuracy 2089
20.0.39 Dirichlet Regression 2089
20.0.40 Dropout and Early Stopping 2090

20.0.41 Dummy Coding . . . 2090
20.0.42 Effect (Deviation) Coding . . . 2090
20.0.43 Elastic Net . . . 2091
20.0.44 Extremely Randomised Trees . . . 2091
20.0.45 Factorial Trials . . . 2091
20.0.46 Feature Engineering . . . 2092
20.0.47 Feature Selection . . . 2092
20.0.48 Feedforward Networks in torch . . . 2093
20.0.49 Fine-Gray Subdistribution Hazards . . . 2093
20.0.50 Frailty Models . . . 2093
20.0.51 GAMMs: Additive Mixed Models . . . 2094
20.0.52 GAMs: Introduction . . . 2094
20.0.53 Generalised Estimating Equations . . . 2094
20.0.54 glmer: Generalised Mixed Models . . . 2095
20.0.55 Gradient Boosting . . . 2095
20.0.56 Hazard, Survival, and Cumulative Hazard . . . 2096
20.0.57 Hurdle Models . . . 2096
20.0.58 Interactions in Regression . . . 2096
20.0.59 Interim Analyses and Group Sequential Designs 2097
20.0.60 Interval-Censored Data . . . 2097
20.0.61 Intraclass Correlation Coefficient (ICC) . . . 2097
20.0.62 Isolation Forest . . . 2098
20.0.63 Isotonic Regression Calibration . . . 2098
20.0.64 ITT vs Per-Protocol Analysis . . . 2099
20.0.65 Joint Longitudinal-Survival Models . . . 2099
20.0.66 k-Means as an ML Tool . . . 2099
20.0.67 k-Nearest Neighbours Classification . . . 2100
20.0.68 k-Nearest Neighbours Regression . . . 2100
20.0.69 Kaplan-Meier Estimation . . . 2100
20.0.70 Kernel SVMs . . . 2101
20.0.71 Landmark Analysis . . . 2101
20.0.72 Lasso Regression . . . 2102
20.0.73 Left Truncation . . . 2102
20.0.74 Leverage and Influence . . . 2102
20.0.75 LightGBM . . . 2103
20.0.76 Likelihood Ratios . . . 2103
20.0.77 LIME Explanations . . . 2103
20.0.78 Linear Discriminant as ML . . . 2104
20.0.79 Linear Regression Assumptions . . . 2104
20.0.80 Linear Support Vector Machines . . . 2104
20.0.81 Log-Rank Test: Details . . . 2105
20.0.82 Logistic Regression . . . 2105
20.0.83 Logistic Regression as ML . . . 2106
20.0.84 Logistic Regression Diagnostics . . . 2106
20.0.85 Minimisation Algorithm . . . 2106
20.0.86 Missing Data in RCTs . . . 2107

20.0.87	Mixed-Effects Models: Introduction	2107
20.0.88	Model Selection with AIC and BIC	2107
20.0.89	Multi-State Models	2108
20.0.90	Multicollinearity and VIF	2108
20.0.91	Multinomial Logistic Regression	2109
20.0.92	Multiple Imputation	2109
20.0.93	Multiple Linear Regression	2109
20.0.94	Naive Bayes	2110
20.0.95	Negative Binomial Regression	2110
20.0.96	Nested Cross-Validation	2110
20.0.97	Nested vs Crossed Random Effects	2111
20.0.98	Neural Networks: Introduction	2111
20.0.99	Non-Inferiority Margin Selection	2112
20.0.100	O'Brien-Fleming Boundary	2112
20.0.101	Offsets in Regression	2112
20.0.102	Ordinal Logistic Regression	2113
20.0.103	Parallel-Group RCT Design	2113
20.0.104	Parametric Survival: Log-Normal	2113
20.0.105	Parametric Survival: Weibull	2114
20.0.106	Partial Correlation	2114
20.0.107	Partial Dependence Plots	2115
20.0.108	Platt Scaling	2115
20.0.109	Pocock Boundary	2115
20.0.110	Poisson Regression	2116
20.0.111	Polynomial Regression	2116
20.0.112	Predictive Values and Prevalence	2116
20.0.113	Preprocessing with recipes	2117
20.0.114	Probit Regression	2117
20.0.115	Quantile Regression	2117
20.0.116	Quasi-Poisson Regression	2118
20.0.117	R-Squared and Adjusted R-Squared	2118
20.0.118	Random Forests	2119
20.0.119	Random-Intercept Models	2119
20.0.120	Random-Slope Models	2119
20.0.121	Randomisation Methods	2120
20.0.122	Recurrent Events	2120
20.0.123	Regularisation in ML	2120
20.0.124	Reliability and Cronbach's Alpha	2121
20.0.125	Residual Diagnostics	2121
20.0.126	Restricted Mean Survival Time (RMST)	2122
20.0.127	Ridge Regression	2122
20.0.128	RNNs and LSTMs: Introduction	2122
20.0.129	Robust Regression	2123
20.0.130	ROC Analysis	2123
20.0.131	ROC and AUC for Logistic Models	2123
20.0.132	Royston-Parmar Flexible Parametric Models	2124

20.0.133 Sample Size Re-Estimation 2124
20.0.134 Schoenfeld Residuals 2125
20.0.135 Sensitivity Analyses in Clinical Trials 2125
20.0.136 SHAP Values 2125
20.0.137 Simple Linear Regression 2126
20.0.138 Simulating Survival Data 2126
20.0.139 SMOTE for Imbalanced Classes 2126
20.0.140 Spline Regression 2127
20.0.141 Stacking Ensembles 2127
20.0.142 Stepped-Wedge Trial 2128
20.0.143 Stepwise Regression: Pitfalls 2128
20.0.144 Stratified Cox Models 2128
20.0.145 Stratified Randomisation 2129
20.0.146 Subgroup Analyses 2129
20.0.147 Subgroup Forest Plots 2129
20.0.148 Supervised Learning: Overview 2130
20.0.149 The Bias-Variance Tradeoff 2130
20.0.150 The Brier Score for Survival 2131
20.0.151 The Nelson-Aalen Estimator 2131
20.0.152 Time-Dependent ROC 2131
20.0.153 Time-Varying Covariates 2132
20.0.154 Tobit Regression 2132
20.0.155 Train-Test Splits 2132
20.0.156 Transformers: Overview 2133
20.0.157 Truncated Regression 2133
20.0.158 Variable Importance 2134
20.0.159 Weighted Kappa 2134
20.0.160 Weighted Log-Rank Tests 2134
20.0.161 Worked lme4::lmer Examples 2135
20.0.162 XGBoost 2135
20.0.163 Zero-Inflated Models 2135

21 PRISMA — systematic reviews & meta-analyses 2137
21.0.1 Bivariate SROC Curves 2137
21.0.2 Converting Between Effect Sizes 2138
21.0.3 Cumulative Meta-Analysis 2138
21.0.4 Egger's Test for Funnel Asymmetry 2138
21.0.5 Fixed-Effect vs. Random-Effects Meta-Analysis 2139
21.0.6 Forest Plots in Meta-Analysis 2139
21.0.7 Funnel Plots 2139
21.0.8 Individual Participant Data Meta-Analysis . . . 2140
21.0.9 Leave-One-Out Meta-Analysis 2140
21.0.10 Meta-Analysis of Diagnostic Accuracy 2140
21.0.11 Meta-Regression 2141
21.0.12 Network Meta-Analysis: Introduction 2141
21.0.13 NMA Consistency and Inconsistency 2142

21.0.14 NMA League Tables 2142
21.0.15 Pooling Correlation Coefficients 2142
21.0.16 Pooling Log-Odds Ratios 2143
21.0.17 Pooling Risk Ratios 2143
21.0.18 Prediction Intervals in Meta-Analysis 2143
21.0.19 Quantifying Heterogeneity: Q and I^2 2144
21.0.20 Selection Models for Publication Bias 2144
21.0.21 Standardised Mean Difference: Hedges' g 2145
21.0.22 Subgroup Meta-Analysis 2145
21.0.23 SUCRA Rankings 2145
21.0.24 Tau-Squared Estimation 2146
21.0.25 Trim-and-Fill 2146

22 ARRIVE — animal research 2147
22.0.1 Alignment with BWA and Bowtie 2147
22.0.2 ATAC-Seq Analysis 2148
22.0.3 Batch Correction with ComBat 2148
22.0.4 Bulk RNA-seq Differential Expression with DESeq2 2148
22.0.5 Cell-Type Annotation 2149
22.0.6 ChIP-Seq Analysis 2149
22.0.7 Copy Number Variation Analysis 2149
22.0.8 Counting Reads with featureCounts 2150
22.0.9 Differential Expression with edgeR 2150
22.0.10 Differential Expression with limma-voom 2151
22.0.11 DNA Methylation Analysis 2151
22.0.12 Drug-Target Interaction Mining 2151
22.0.13 FASTQ Quality Control 2152
22.0.14 Finding Marker Genes 2152
22.0.15 Gene Annotation with biomaRt 2152
22.0.16 Gene Ontology Enrichment 2153
22.0.17 GSEA Preranked Analysis 2153
22.0.18 GSVA Single-Sample Enrichment 2154
22.0.19 Heatmaps for RNA-seq 2154
22.0.20 Integration with Harmony 2154
22.0.21 KEGG Pathway Enrichment 2155
22.0.22 MA Plots 2155
22.0.23 Metagenomic Profiling 2155
22.0.24 Microbiome Analysis with DADA2 2156
22.0.25 Microbiome Diversity Metrics 2156
22.0.26 Multi-Omics Integration 2157
22.0.27 Multiple Sequence Alignment 2157
22.0.28 PCA of RNA-seq Samples 2157
22.0.29 Phylogenetic Trees with ape 2158
22.0.30 Population Genetics Basics 2158
22.0.31 Protein Structure Prediction 2158

22.0.32 Proteomics with MSstats . . . 2159
22.0.33 Pseudoalignment with Salmon and Kallisto . . 2159
22.0.34 Read Trimming and Adapter Removal 2159
22.0.35 RNA-seq Normalisation . . . 2160
22.0.36 scRNA Clustering . . . 2160
22.0.37 scRNA Normalisation . . . 2161
22.0.38 scRNA QC and Filtering . . . 2161
22.0.39 Sequence Alignment: Overview . . . 2161
22.0.40 Single-Cell with Seurat . . . 2162
22.0.41 Spatial Transcriptomics . . . 2162
22.0.42 STAR: Spliced Alignment . . . 2162
22.0.43 Structural Variant Detection . . . 2163
22.0.44 Surrogate Variable Analysis (SVA) . . . 2163
22.0.45 Trajectory Analysis . . . 2164
22.0.46 Transcript-to-Gene Summarisation . . . 2164
22.0.47 Variant Annotation with VEP . . . 2164
22.0.48 Variant Calling with GATK . . . 2165
22.0.49 VCF Manipulation . . . 2165
22.0.50 Volcano Plots . . . 2166
22.1 Week 4, Session 5 — Explanation vs prediction; reporting 2166
22.1.1 Learning objectives . . . 2166
22.1.2 Prerequisites . . . 2166
22.1.3 Background . . . 2166
22.1.4 Setup . . . 2167
22.1.5 1. Goal . . . 2167
22.1.6 2. Approach . . . 2167
22.1.7 3. Execution . . . 2167
22.1.8 4. Check . . . 2167
22.1.9 5. Report . . . 2168
22.1.10 Common pitfalls . . . 2168
22.1.11 Further reading . . . 2169
22.1.12 Session info . . . 2169

23 The research workflow 2171
23.1 1. Question . . . 2171
23.2 2. Measurements . . . 2172
23.3 3. Design . . . 2172
23.4 4. Acquisition . . . 2172
23.5 5. Description . . . 2172
23.6 6. Analysis . . . 2172
23.7 7. Interpretation / prediction . . . 2172
23.8 8. Validation . . . 2172
23.9 9. Knowledge / decisions . . . 2173

24 Glossary 2175
24.1 A . . . 2175

24.2 B . . . 2175
24.3 C . . . 2176
24.4 D . . . 2176
24.5 E . . . 2176
24.6 F . . . 2176
24.7 G . . . 2176
24.8 H . . . 2177
24.9 I . . . 2177
24.10 K . . . 2177
24.11 L . . . 2177
24.12 M . . . 2178
24.13 N . . . 2178
24.14 O . . . 2178
24.15 P . . . 2178
24.16 R . . . 2179
24.17 S . . . 2180
24.18 T . . . 2180
24.19 V . . . 2180
24.20 W . . . 2180

25 Common errors **2181**
25.1 R . . . 2181
25.2 Quarto . . . 2182
25.3 `renv` . . . 2183
25.4 knitr / rmarkdown . . . 2183
25.5 Git . . . 2183

26 Writing a report **2185**
26.1 IMRaD . . . 2185
26.2 Effect sizes and intervals . . . 2185
26.3 Figures . . . 2186
26.4 Tables . . . 2186
26.5 Citing R and packages . . . 2186
26.6 Reporting guidelines . . . 2186
26.7 Reproducibility . . . 2187

27 Cheatsheets **2189**
27.1 Foundations . . . 2189
27.1.1 Toolchain, tidy data, and `ggplot2` . . . 2189
27.2 R / RStudio / Quarto / renv . . . 2189
27.3 Data types and scales . . . 2190
27.4 Tidy data . . . 2190
27.5 `dplyr` verbs . . . 2190
27.6 `ggplot2` grammar . . . 2190
27.7 Plot families . . . 2190
27.8 Decision rule . . . 2191

27.9 Common pitfalls . . . 2191
27.10 Further reading . . . 2191
27.10.1 Descriptive statistics, Bayes, and distributions . 2191
27.11 Contingency tables . . . 2191
27.12 Bayes' theorem . . . 2191
27.13 Diagnostic testing . . . 2192
27.14 Discrete distributions . . . 2192
27.15 Continuous distributions . . . 2192
27.16 Q-Q plot . . . 2193
27.17 Decision rule . . . 2193
27.18 Common pitfalls . . . 2193
27.19 Further reading . . . 2193
27.19.1 Sampling, estimation, and one-sample tests . . 2193
27.20 Central limit theorem . . . 2193
27.21 Standard error . . . 2193
27.22 Bootstrap . . . 2194
27.23 Permutation test . . . 2194
27.24 Maximum likelihood . . . 2194
27.25 One-sample tests (five-step template) . . . 2194
27.26 Hypothesis-testing vocabulary . . . 2194
27.27 Decision rule . . . 2195
27.28 Common pitfalls . . . 2195
27.29 Further reading . . . 2195
27.29.1 Two-group comparisons, effect sizes, and power 2195
27.30 Effect sizes . . . 2195
27.31 Two proportions . . . 2196
27.32 Correlation . . . 2196
27.33 Non-parametric tests . . . 2196
27.34 Power and sample size . . . 2196
27.35 Reporting with `gtsummary` . . . 2197
27.36 Decision rule . . . 2197
27.37 Common pitfalls . . . 2197
27.38 Further reading . . . 2197
27.39 Regression . . . 2197
27.39.1 Linear regression and diagnostics . . . 2197
27.40 Simple linear regression . . . 2197
27.41 Multiple regression . . . 2198
27.42 Diagnostics . . . 2198
27.43 Robust / weighted . . . 2198
27.44 Decision rule . . . 2198
27.45 Common pitfalls . . . 2199
27.46 Further reading . . . 2199
27.46.1 ANOVA, mixed models, and non-linear regression 2199
27.47 Contrasts with `emmeans` . . . 2199
27.48 Two-way / factorial ANOVA . . . 2199
27.49 Repeated measures / blocking . . . 2200

27.50 GAMs — smooth non-linear terms . . . 2200
27.51 Non-linear regression (`nls`) . . . 2200
27.52 Decision rule . . . 2200
27.53 Common pitfalls . . . 2200
27.54 Further reading . . . 2200
27.54.1 Generalised linear models and prediction evaluation . . . 2201
27.55 Logistic regression . . . 2201
27.56 ANCOVA in an RCT . . . 2201
27.57 Poisson / negative binomial . . . 2201
27.58 Evaluation — calibration + discrimination . . . 2202
27.59 Decision rule . . . 2202
27.60 Common pitfalls . . . 2202
27.61 Further reading . . . 2202
27.61.1 Agreement, survival primer, and reporting . . . 2203
27.62 Change scores vs ANCOVA . . . 2203
27.63 Agreement & reliability . . . 2203
27.64 Survival primer . . . 2203
27.65 Decision curves, NRI, IDI . . . 2204
27.66 Explanation vs prediction (Shmueli) . . . 2204
27.67 Reporting guidelines . . . 2204
27.68 Decision rule . . . 2204
27.69 Common pitfalls . . . 2204
27.70 Further reading . . . 2205
27.71 Design & Causal Inference . . . 2205
27.71.1 Study designs and power . . . 2205
27.72 Trial designs . . . 2205
27.73 Bench / translational . . . 2206
27.74 Power — closed form . . . 2206
27.75 Power — simulation . . . 2206
27.76 Decision rule . . . 2206
27.77 Common pitfalls . . . 2206
27.78 Further reading . . . 2207
27.78.1 Missing data, mixed models, and longitudinal . 2207
27.79 Multiple imputation with `mice` . . . 2207
27.80 Linear mixed models . . . 2207
27.81 GLMMs & GEE . . . 2208
27.82 Time series basics . . . 2208
27.83 Decision rule . . . 2208
27.84 Common pitfalls . . . 2208
27.85 Further reading . . . 2208
27.85.1 Causal inference methods . . . 2209
27.86 Competing risks . . . 2209
27.87 DAGs . . . 2209
27.88 Propensity scores / IPTW . . . 2209
27.89 G-methods, IV, DiD, RDD . . . 2210

27.90 Heterogeneous treatment effects . . . 2210
27.91 Decision rule . . . 2210
27.92 Common pitfalls . . . 2210
27.93 Further reading . . . 2210
27.93.1 Synthesis and pre-registration . . . 2211
27.94 Meta-analysis . . . 2211
27.95 Network meta-analysis . . . 2211
27.96 Infectious disease — SIR / SEIR . . . 2212
27.97 Pre-registration and SAP . . . 2212
27.98 Decision rule . . . 2212
27.99 Common pitfalls . . . 2212
27.100 Further reading . . . 2213
27.101 Machine Learning & High-Dimensional Data . . . 2213
27.101.1 Cross-validation, regularisation, and unsupervised 2213
27.102 Regularisation . . . 2213
27.103 Multivariate workhorses . . . 2213
27.104 Clustering . . . 2214
27.105 Non-linear embeddings . . . 2214
27.106 Decision rule . . . 2214
27.107 Common pitfalls . . . 2214
27.108 Further reading . . . 2215
27.108.1 Trees, boosting, and interpretability . . . 2215
27.109 Interpretability . . . 2215
27.110 Tabular neural networks with `torch` . . . 2215
27.111 tidymodels pipeline . . . 2216
27.112 Decision rule . . . 2216
27.113 Common pitfalls . . . 2216
27.114 Further reading . . . 2217
27.114.1 Bayesian and survival ML . . . 2217
27.115 `brms` / Stan . . . 2217
27.116 Biomarker statistics . . . 2217
27.117 Survival ML . . . 2218
27.118 External validation . . . 2218
27.119 Decision rule . . . 2218
27.120 Common pitfalls . . . 2218
27.121 Further reading . . . 2218
27.121.1 Omics, FDR, and TRIPOD-AI . . . 2219
27.122 Enrichment . . . 2219
27.123 Single-cell RNA-seq (Seurat pattern) . . . 2219
27.124 FDR, knockoffs, replication . . . 2220
27.125 TRIPOD-AI + fairness . . . 2220
27.126 Reproducibility at scale with `targets` . . . 2220
27.127 Decision rule . . . 2221
27.128 Common pitfalls . . . 2221
27.129 Further reading . . . 2221

Colophon **2223**

Citing this Guide **2225**

References **2227**

Welcome!

R. Heller · Date: 2026-05-08 · Last changed: 2026-11-11

Welcome to **"Methods in R: A Topic Reference for Biomedical Research and Peer Review"**.

This book is shaped for two readers it expects to serve in the same week: the **author** who has data in front of them and needs to pick a method, run it in R, and write the Methods paragraph; and the **peer reviewer** who has a manuscript on their desk and needs to know what a competent application of that method looks like, which diagnostics should have been reported, and what red flags to flag.

The body covers sixteen topic areas — from foundations and inference through regression, survival, Bayesian methods, machine learning, bioinformatics, and study design — with hands-on labs that consolidate the methods of each area into a single runnable file. Every method page ends with a *For Reviewers* section: the misapplications, the missing diagnostics, the red flags in tables and figures, and the things to verify against the authors' references and arithmetic. A final part, *Reporting Templates*, gives copy-paste Methods and Results paragraphs tagged by **STROBE / CONSORT / TRIPOD-AI / PRISMA / ARRIVE**.

The book is not a textbook of statistics. It assumes you can read an R session and follow a method already named to you. Where it differs from the existing biostatistics shelf is in being optimised for lookup, not linear reading, and in pairing every method with a concrete review checklist.

Open Source Repository

This book has been built using **{rmarkdown}** and **{bookdown}**. Formulas are rendered using MathJax. All source files are available on **GitHub**: https:

//github.com/CTTIR/methods-in-r. You are free to fork, share, and reuse the contents under the CC BY 4.0 license.

How To Use The Guide

- **Linear vs. lookup.** If you already know the method, jump to its chapter. If not, start at Find your method and follow the decision tree.
- **Where the code lives.** Code in each method page and lab is display-only in the rendered book; copy it into an R session with the packages cited in that section attached. Reusable helpers are kept in `R/helpers.R` in the source repository.
- **Per-chapter downloads.** Every chapter page offers the chapter as PDF and EPUB below its title.
- **Whole-book PDF and EPUB.** Use the navbar *Download* menu for a single-file PDF or EPUB.
- **Corrections and suggestions.** Open an issue at https://github.com/CTTIR/methods-in-r/issues.

Contributing

Issues and pull requests are welcome — see CONTRIBUTING.md.

Citation

The suggested citation is:

> **Heller, R.** (2026). *Methods in R: A Topic Reference for Biomedical Research and Peer Review* (Version 1.0.0). Self-published via GitHub Pages. https://cttir.github.io/methods-in-r/.

Download the reference as BibTeX or .ris.

Impressum

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-09

Methods in R: A Topic Reference for Biomedical Research and Peer Review

© 2026 Raban Heller. ORCID: 0000-0001-8006-9742.

Content released under the Creative Commons Attribution 4.0 International License (CC BY 4.0).

Source code released under the MIT License.

Published online: https://cttir.github.io/methods-in-r/

Source repository: https://github.com/CTTIR/methods-in-r

Build date: 2026-11-14.

Suggested citation

> Heller, R. (2026). *Methods in R: A Topic Reference for Biomedical Research and Peer Review.* https://cttir.github.io/methods-in-r/.

Disclaimer

This book is provided "as is" without warranty of any kind. The author assumes no responsibility for errors or omissions, and the content is not a substitute for professional statistical or clinical advice.

Contact

For corrections, suggestions, or issues: https://github.com/CTTIR/methods-in-r/issues

Acknowledgments

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

This book is built with bookdown by Yihui Xie and the R Markdown community. The wider R ecosystem — base R, the tidyverse, and the many domain-specific packages cited in the body — is the foundation on which every chapter rests.

The original lab and tutorial material that inspired this volume was authored under the `CTTIR/courses` and `CTTIR/tutorials` repositories. Those repositories are credited in the Preface and at the points of use.

The decision-tree wizard hosted at `cttir.github.io/tutorials/decision-tree/` draws structural inspiration from the University of Zurich's *Methodenberatung* (methodenberatung.uzh.ch).

Drafting assistance from Anthropic's Claude was used during preparation; all editorial decisions, errors, and omissions remain the author's.

Inspiration

The structural and pedagogical model for this book draws heavily on Mathias Harrer, Pim Cuijpers, Toshi A. Furukawa, and David D. Ebert's *Doing Meta-Analysis with R: A Hands-On Guide* — both the book and its open-source repository at https://github.com/MathiasHarrer/Doing-Meta-Analysis-in-R. Their thorough, transparent, and beautifully laid-out bookdown setup — sidebar grouping, semantic callout boxes, the citation block, the dark/light toggle, MathJax-rendered formulas, downloadable BibTeX/RIS citations — set a clear standard for what a reproducible R-based research book can look like. Where the present book mirrors their conventions, it does so deliberately and gratefully. Where it diverges (colors, typography, individual icons), the divergence is cosmetic; the underlying craft is theirs to credit.

Use of LLM tools

Portions of this book were prepared with assistance from large language model tooling for narrowly defined, non-authorial tasks: copyediting, prose smoothing, Markdown/LaTeX formatting, scaffolding of boilerplate files (CI configs, build scripts), code refactoring. The tools used were Chat AI, the LLM service of KISSKI (GWDG), and a self-hosted **Mistral Small (24B, Apache-2.0)** run locally via Ollama and the `ollamar` R package — local inference only, with no data sent to third parties for the self-hosted model.

All scientific claims, methodological choices, analyses, interpretations, and conclusions are the author's own. No LLM-generated text was incorporated without review and revision, and every reference was verified against its DOI, arXiv ID, or ISBN.

Notation

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

This page lists the symbols, conventions, and abbreviations used throughout the book. Where a chapter departs from these conventions locally, the chapter says so explicitly.

Greek symbols

Symbol	Meaning
α	Type I error rate (significance threshold).
β	Type II error rate; also a regression coefficient (context).
μ	Population mean.
σ	Population standard deviation; σ^2 for variance.
π	Population proportion.
θ	Generic population parameter; $\hat{\theta}$ for the estimator.
τ^2	Between-study variance in meta-analysis.
ρ	Population correlation; intra-class correlation (ICC).
λ	Rate parameter; eigenvalue (context).

Latin symbols

Symbol	Meaning
n	Sample size.
N	Population size.
$\bar{x}$	Sample mean.
s	Sample standard deviation; s^2 for variance.
$\hat{p}$	Sample proportion.
SE	Standard error of an estimator.
p	p-value.
L, ℓ	Likelihood and log-likelihood.

Symbol	Meaning
I^2	Heterogeneity statistic in meta-analysis.

Operators and notation

- $\sim$ — "is distributed as".
- $\hat{\theta}$ — an estimator of the parameter θ.
- $X \perp Y$ — independence.
- $E[\cdot]$, $\mathrm{Var}[\cdot]$ — expectation and variance.
- $X \mid Y$ — X given Y (conditional).

Abbreviations

Abbrev.	Expansion
CI	Confidence interval (frequentist); credible interval (Bayesian) — context disambiguates.
RR	Relative risk (risk ratio).
OR	Odds ratio.
HR	Hazard ratio.
AUC	Area under the (ROC) curve.
LR+ / LR−	Positive / negative likelihood ratio.
PPV / NPV	Positive / negative predictive value.
MLE	Maximum likelihood estimator.
GLM	Generalised linear model.
GLMM	Generalised linear mixed model.
GEE	Generalised estimating equation.
RCT	Randomised controlled trial.
ITT	Intention to treat.
PP	Per protocol.
MI	Multiple imputation.
FDR	False discovery rate.
BH	Benjamini–Hochberg (FDR procedure).
TRIPOD	Transparent Reporting of a multivariable prediction model.
STROBE	Strengthening the Reporting of Observational Studies in Epidemiology.
CONSORT	Consolidated Standards of Reporting Trials.
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses.
ARRIVE	Animal Research: Reporting of In Vivo Experiments.
STARD	Standards for Reporting of Diagnostic Accuracy.

Code conventions

- The base R pipe `|>` is used throughout. Files mixing `%>%` and `|>` are flagged in `GENERATION_LOG.md`.
- Package functions are written qualified the first time they appear in a chapter (`dplyr::filter()`), then bare on subsequent uses.
- Code blocks shown in the rendered book are display-only; the book is built with `eval=FALSE` globally and verified to knit cleanly without side effects. To run a chunk yourself, copy it into an R session with the libraries cited in that section attached.

About the author

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-09

Raban Heller is a biomedical researcher and applied statistician. This book consolidates the method pages and labs from his earlier public repositories (`CTTIR/courses`, `CTTIR/tutorials`) into a single topic-organised reference, extended with a *For Reviewers* section on every method page.

- ORCID: https://orcid.org/0000-0001-8006-9742
- GitHub: https://github.com/r-heller
- Website: https://r-heller.github.io/website/

Sibling volume

This is the second book in the *…in R* series. The first is `CTTIR/strategy-in-r` — *Strategy in R: Game Theory, Simulation, and Machine Intelligence*. The two share branding, build conventions (bookdown, `renv`, GitHub Pages deploy), and a preference for R-only worked examples over toy abstractions.

How to use this book

R. Heller · Date: 2026-05-08 · Last changed: 2026-11-10

The book is organised around **method topics**, not courses. There are sixteen topic chapters, an unnumbered method chooser, five reporting-template chapters, and five appendices.

Three reading modes

As an author with data. Open Find your method. Answer the question *"what are you trying to do?"* until you land on a method. Read the method page top to bottom. Adapt and check the displayed R code against your data, package versions and assumptions. The Output & Results section explains what to inspect. Reporting templates are starting points to complete with your actual design and results, not descriptions of analyses you have not performed.

As a peer reviewer. Search for the method by name and jump straight to the **For Reviewers** section at the bottom of the page. It links to the shared checklist below and to that method's assumptions and practical tips. Detailed method-specific reviewer notes remain editorial work; they are not yet supplied by the imported entries. Read Theory and Assumptions before judging an application.

As a learner. Each topic chapter has labs under `chapters/<topic>/labs/`. Labs follow *Hypothesis → Visualise → Assumptions → Conduct → Conclude*. Work through them in order; the methods they exercise live in the same chapter and are linked from the labs.

Conventions

- Examples are R, displayed without execution during the reference build. All 80 lab scripts pass syntax checks; the maintenance review executed the tidy-data and import/join labs. Other examples and scientific outputs still need checking in their intended data and package environment.

- Method pages organize background, prerequisites, theory, assumptions, code, interpretation and practical tips. Their reviewer sections provide navigation to the relevant checks rather than completed specialist appraisals.
- The Reporting Templates part contains draft Methods and Results wording. Replace every example-specific statement with the analysis actually performed.

Shared reviewer checklist

1. **Match the analysis to the question and design.** Identify the estimand, outcome scale, sampling unit and dependence structure. Compare them with the selected method's Theory and Assumptions sections.
2. **Ask for evidence about relevant assumptions.** Locate the diagnostics and sensitivity analyses needed for this application. A successful package call alone does not establish that the assumptions are met.
3. **Reconcile the reported analysis.** Check sample counts and denominators, exclusions, missing-data handling, transformations and uncertainty summaries against the stated model and supplied analysis code.
4. **Check reproducibility and interpretation.** Look for data provenance, package versions and sufficient code to reproduce the reported calculation. Distinguish the model's output from the scientific claim made from it.
5. **Check the Methods paragraph.** It should identify the question, analysis unit, model or test, relevant settings, handling of missing observations and diagnostic checks. Record unresolved details rather than assuming defaults.

This is a general reading aid. Method-specific validity and reporting requirements need an appraisal of the actual study and relevant primary methodological sources.

What this book is not

It is not a linear introductory textbook. It does not derive every result from first principles. It does not cover SAS, Stata, SPSS, or Python. It does not host the interactive decision-tree wizard (Shiny apps remain on the live site).

Building locally

```
Rscript --vanilla -e '
install.packages(
  c("bookdown", "rmarkdown", "knitr", "bslib",
    "downlit", "xml2", "htmltools", "sessioninfo"),
  repos = "https://cloud.r-project.org"
```

```
)'
python3 -m unittest discover -s tests
Rscript --vanilla -e 'bookdown::render_book("index.Rmd",
 ↪  output_format="bookdown::bs4_book")'
python3 scripts/build-download-manifest.py
python3 -m http.server 8000 --directory docs
```

Run these commands from the repository root and open http://localhost:8000. HTML is the primary reading format. README.md also documents the verified EPUB build and its link-finalization step. PDF requires a separate TeX qualification.

Find your method

R. Heller · Date: 2026-05-08 · Last changed: 2026-11-10

Open the book here when you don't know which chapter to read. Three routes are offered: a flowchart from research question to statistical test, a quick lookup table, and a click-through wizard hosted live.

- **Interactive wizard** — three to five questions, one recommended test.
- **Static decision tree** (below) — the same logic, single page, suitable for printing.
- **Quick lookup table** — common scenarios mapped to a recommended chapter and section.

This chapter covers the *core* tree of classical tests (differences, associations, interdependence). Methods outside that core — survival, Bayesian, machine learning, time series, multivariate, meta-analysis, bioinformatics, study-design choices — are reached through their topic chapters or via the lookup table below.

The static decision tree

Colour encodes the scale level of the outcome; a dashed border marks a non-parametric test. Use the quick lookup links below to reach the chapters. This is an orientation aid; read the method's assumptions and study-design checks before selecting a test.

Open the full-size decision tree

Quick lookup

A small table for the most common biomedical scenarios. The full tree above and the live wizard cover everything the table omits.

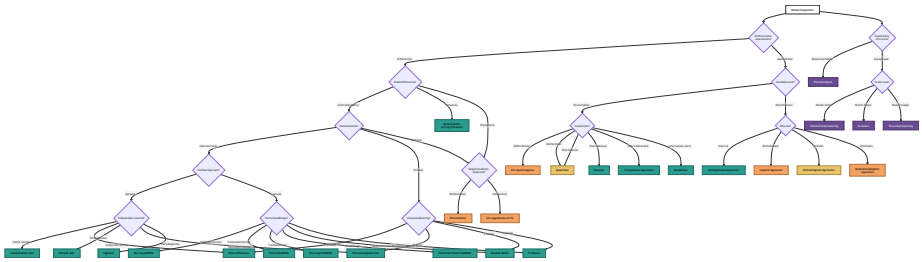


Figure 1: Decision tree from research question, outcome scale and study design to candidate statistical methods.

Scenario	Recommended method	Chapter
2 independent groups, metric, normal	Independent / Welch t-test	4
2 independent groups, non-normal or ordinal	Mann-Whitney U	4
2 paired groups, metric	Paired t-test	4
3+ groups, between, normal	One-way ANOVA	4
3+ groups, between, non-normal	Kruskal-Wallis	4
Two continuous, linear	Pearson correlation	4
Continuous outcome, 2+ predictors	Multiple linear regression	7
Binary outcome, 1+ predictors	Logistic regression	7
Time-to-event with censoring	Kaplan-Meier, Cox	11
Diagnostic test accuracy	Sensitivity/specificity, ROC	14
Aggregating effect sizes across studies	Pairwise meta-analysis	15
Predictive model with cross-validation	tidymodels workflow	13
Posterior inference with priors	brms / Stan	10
Repeated measures within subjects	Mixed-effects model	16
Two or more variances	Levene's test	4
Reduce variables to latent dimensions	PCA / factor analysis	8

The static tree is mirrored from the decision-tree wizard at https://cttir.github.io/tutorials/decision-tree/. Structure inspired by the University of Zurich Methodenberatung (methodenberatung.uzh.ch).

Chapter 1

Statistical Foundations

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

This chapter covers the prerequisites that every later chapter assumes: the scientific process, tidy data, the R/Quarto/`renv` toolchain, measurement quality, missingness, and reproducibility. If you have never built a project with `renv` or written a pre-registration, start here.

This chapter contains **31 method pages** and **4 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

1.1 Method pages

Method	Source slug
Bias and Variance of Estimators	`bias-and-variance`
Big-Op and Little-op Notation	`big-op-little-op`
Characteristic Functions	`characteristic-functions`
Confidence Intervals: Introduction	`confidence-intervals-introduction`
Convergence in Distribution	`convergence-in-distribution`
Convergence in Probability	`convergence-in-probability`
Estimators and Estimation	`estimators-and-estimation`
Fisher Information	`fisher-information`
Jensen's Inequality	`jensens-inequality`
Likelihood Ratio Tests	`likelihood-ratio-tests`
Markov and Chebyshev Inequalities	`chebyshev-markov`
Maximum Likelihood Estimation	`maximum-likelihood-estimation`
Moments and Moment Generating Functions	`moments-and-mgfs`
Order Statistics	`order-statistics`

Method	Source slug
Pivotal Quantities	`pivotal-quantities`
Population vs. Sample	`population-vs-sample`
Sampling Distributions	`sampling-distributions`
Sampling Methods	`sampling-methods`
Scales of Measurement	`scales-of-measurement`
Slutsky's Theorem	`slutsky-theorem`
Sufficient Statistics	`sufficient-statistics`
The Cauchy-Schwarz Inequality	`cauchy-schwarz`
The Central Limit Theorem	`central-limit-theorem`
The Cramer-Rao Lower Bound	`cramer-rao-lower-bound`
The Delta Method	`delta-method`
The Empirical Distribution Function	`empirical-distribution-function`
The Glivenko-Cantelli Theorem	`glivenko-cantelli`
The Law of Large Numbers	`law-of-large-numbers`
The Method of Moments	`method-of-moments`
The Standard Error	`standard-error`
Unbiasedness, Consistency, Efficiency	`unbiasedness-consistency-efficiency`

1.2 Labs

Lab
Scientific process and research workflow
R, RStudio, Quarto, and renv toolchain
Data types, tidy data, accuracy and precision
Import, joins, and missingness with dplyr

1.3 Bias and Variance of Estimators

1.3.1 Introduction

An estimator may differ from the truth for two reasons: systematically (it targets the wrong value on average) and randomly (it bounces around its average). Bias captures the first; variance captures the second. Together they determine how close the estimator is to the truth, as measured by mean squared error (MSE). Many important statistical choices – between raw and shrinkage estimators, between parametric and non-parametric methods, between simple and flexible models – are bias/variance trade-offs.

1.3.2 Prerequisites

The reader should know what an expectation is and should be comfortable simulating repeated sampling in R with `replicate()`.

1.3.3 Theory

For an estimator $\hat{\theta}$ of a population parameter θ:

- **Bias**: $\text{Bias}(\hat{\theta}) = E[\hat{\theta}] - \theta$. An unbiased estimator has zero bias.
- **Variance**: $\text{Var}(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]$. This is the square of the standard error.
- **Mean squared error**: $\text{MSE}(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]$.

These are linked by the classical decomposition:

$$\text{MSE}(\hat{\theta}) = \text{Bias}(\hat{\theta})^2 + \text{Var}(\hat{\theta}).$$

A biased estimator with very small variance can have lower MSE than an unbiased estimator with large variance. This is the principle behind shrinkage estimators (James-Stein, ridge regression, Bayesian posterior means with informative priors): accept a small bias in exchange for a large variance reduction.

Examples of bias:

- The sample variance with divisor n is biased; with divisor $n - 1$ it is unbiased. Both are consistent.
- The MLE of variance in a normal model ($\sigma^2 = \sum(x_i - \bar{x})^2/n$) is biased downward by a factor of $(n-1)/n$.
- The sample standard deviation $s = \sqrt{s^2}$ is biased even though s^2 is unbiased, because the square root is concave.

1.3.4 Assumptions

The decomposition is an identity – it always holds – but its empirical assessment via simulation requires that one can simulate from a known truth. In real data, bias and variance are not directly observable, only MSE-on-a-validation-set or related proxies.

1.3.5 R Implementation

```
library(ggplot2)
set.seed(2026)

true_sigma2 <- 9
n <- 10
reps <- 20000

mle_var <- function(x) mean((x - mean(x))^2)
ub_var  <- function(x) var(x)

est_mle <- replicate(reps, mle_var(rnorm(n, sd = sqrt(true_sigma2))))
```

```r
est_ub  <- replicate(reps, ub_var(rnorm(n, sd = sqrt(true_sigma2))))

bias_var <- function(est, truth) {
  c(bias     = mean(est) - truth,
    variance = var(est),
    mse      = mean((est - truth)^2))
}

round(bias_var(est_mle, true_sigma2), 3)
round(bias_var(est_ub,  true_sigma2), 3)
```

Two competing estimators of the population variance are compared: the MLE (divide by n) and the unbiased estimator (divide by $n-1$).

1.3.6 Output & Results

```
    bias variance      mse
   -0.90    14.68    15.49      # MLE, biased downward
    0.01    18.11    18.11      # unbiased estimator
```

For $n = 10$, the MLE is biased (expected value 8.10 vs. truth 9.00) but has lower variance and lower MSE. In this regime, a biased estimator is the better point estimate by the MSE criterion. As n grows, both bias and the gap in MSE vanish.

1.3.7 Interpretation

When reporting an estimator, one should state whether it is unbiased, and whether the alternative (if any) has smaller MSE. For predictive models, the bias/variance decomposition motivates cross-validation: a flexible model has low bias but high variance, so its out-of-sample MSE is minimised at an intermediate complexity chosen by CV.

1.3.8 Practical Tips

- Unbiasedness is a mathematical convenience, not a goal. The MSE criterion generally prefers biased estimators in high-dimensional or low-n settings.
- Bias is reported as "bias $= E[\hat{\theta}] - \theta$"; some authors flip the sign (especially in MCMC diagnostics), so read definitions carefully.
- In regression, predictions at a single input can be decomposed into bias and variance terms; this decomposition motivates regularisation.
- Shrinkage estimators (ridge, Bayesian posterior mean, empirical Bayes) explicitly trade a small bias for a larger variance reduction. This is visible in the MSE comparison.

- When simulating bias/variance to inform methodology choice, use many replicates (10 000+); small-reps estimates of bias are themselves noisy.

1.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.3.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.4 Big-Op and Little-op Notation

1.4.1 Introduction

Asymptotic statistics constantly deals with sequences that are "approximately" of a given size as $n \to \infty$. Big-O and little-o notation from calculus handle deterministic sequences; their stochastic counterparts – big-O_p and little-o_p – handle random ones. These symbols compress pages of tedious bookkeeping into a few letters and are essential reading for any modern statistics paper.

1.4.2 Prerequisites

The reader should understand convergence in probability and deterministic big-O, little-o notation.

1.4.3 Theory

Big-O_p (stochastic boundedness). A sequence Y_n is $O_p(r_n)$ if for every $\varepsilon > 0$ there exist M, N such that

$$P(|Y_n/r_n| \leq M) \geq 1 - \varepsilon \quad \text{for all } n \geq N.$$

Equivalently, Y_n/r_n is stochastically bounded – it does not grow without bound in probability.

Little-o_p (negligibility). A sequence Y_n is $o_p(r_n)$ if $Y_n/r_n \xrightarrow{P} 0$.

Standard asymptotic rates.

- $\bar{X}_n - \mu = O_p(n^{-1/2})$ by the CLT.
- $\hat{\sigma}_n^2 - \sigma^2 = O_p(n^{-1/2})$ similarly.
- If $\hat{\theta}_n$ is consistent, $\hat{\theta}_n - \theta = o_p(1)$.
- In regression, $\hat{\beta}_n - \beta = O_p(n^{-1/2})$ for OLS under standard conditions.

Algebraic rules.

- $O_p(1) + O_p(1) = O_p(1)$ (sum of bounded remains bounded).
- $o_p(1) + o_p(1) = o_p(1)$.
- $O_p(a_n) \cdot O_p(b_n) = O_p(a_n b_n)$.
- $O_p(1) \cdot o_p(1) = o_p(1)$ (bounded times negligible is negligible).

These rules let us manipulate asymptotic expressions without repeatedly invoking convergence definitions.

Delta-method expansion. If $\hat{\theta}_n - \theta = O_p(n^{-1/2})$ and g is smooth,

$$g(\hat{\theta}_n) - g(\theta) = g'(\theta)(\hat{\theta}_n - \theta) + O_p(n^{-1}).$$

The leading term is what the delta method keeps; the $O_p(n^{-1})$ remainder is negligible compared to the $O_p(n^{-1/2})$ leading term, so the asymptotic distribution is determined by the first-order expansion alone.

1.4.4 Assumptions

Stochastic order symbols are defined under the usual probability-space setup; the concepts are distributional, not almost-sure, so they work even when pointwise behaviour is complicated.

1.4.5 R Implementation

```r
set.seed(2026)

n_vals <- round(10^seq(1, 4, by = 0.5))
reps <- 1000

sim <- sapply(n_vals, function(n) {
  dev <- replicate(reps, abs(mean(rnorm(n)) - 0))
  mean(dev * sqrt(n))
})

# If xbar - mu = O_p(n^-1/2), sqrt(n) * (xbar - mu) should be bounded
data.frame(n = n_vals, mean_abs_sqrt_n = sim)
```

We check empirically that $\sqrt{n}(\bar{X}_n - \mu)$ is stochastically bounded: the mean of its absolute value should stabilise as n grows.

1.4.6 Output & Results

```
     n  mean_abs_sqrt_n
1   10       0.829
2   32       0.808
3  100       0.794
4  316       0.801
5 1000       0.797
6 3162       0.798
7 10000      0.797
```

The product stabilises near $\sqrt{2/\pi} \approx 0.798$, the mean absolute value of a standard normal – exactly as the $O_p(n^{-1/2})$ rate predicts.

1.4.7 Interpretation

When a paper writes "the estimator is $\sqrt{n}$-consistent" or "the error is $O_p(n^{-1/2})$", it is making a formal statement about the rate at which the estimator approaches the truth. Different estimators can have different rates (parametric vs. non-parametric, standard vs. super-efficient), and these rates directly affect sample-size planning.

1.4.8 Practical Tips

- $\sqrt{n}$-consistent is the gold standard for parametric estimators; non-parametric estimators often have slower rates like $n^{-1/3}$ or $n^{-2/5}$.
- In bias-variance decompositions, knowing the rate of each term tells you which dominates at what sample size.
- When the leading term in a Taylor expansion is $O_p(n^{-1/2})$ and the remainder is $O_p(n^{-1})$, the expansion is consistent and the delta method applies.
- Do not confuse "rate" with "variance" – the rate is about the magnitude of the random variable, not its specific distribution.
- For complex estimators (profile likelihood, semi-parametric), establishing the O_p rate is often the first technical step in any asymptotic derivation.

1.4.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.4.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain

- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.5 The Cauchy-Schwarz Inequality

1.5.1 Introduction

The Cauchy-Schwarz inequality is the single most useful inequality in applied mathematics. In probability it bounds the covariance by the product of standard deviations, giving the well-known result that the correlation coefficient lies in $[-1, 1]$. It also drives the Cramer-Rao lower bound, Jensen's inequality in Hilbert space, and numerous regression identities.

1.5.2 Prerequisites

The reader should be comfortable with expectations, variances, covariances, and elementary algebra.

1.5.3 Theory

Cauchy-Schwarz for expectations. For any random variables X, Y with finite second moments,

$$|E[XY]| \leq \sqrt{E[X^2]\, E[Y^2]}.$$

Equality holds if and only if $Y = aX$ for some constant a almost surely (i.e., X, Y are linearly dependent).

Corollary for covariances. Applied to $X - \mu_X$ and $Y - \mu_Y$,

$$|\mathrm{Cov}(X, Y)| \leq \sigma_X \sigma_Y,$$

equivalently, $|\rho_{XY}| \leq 1$. This is why the Pearson correlation is bounded between -1 and $+1$.

Inner-product form. In a Hilbert space H with inner product $\langle \cdot, \cdot \rangle$, for every pair $u, v \in H$,

$$|\langle u, v \rangle| \leq \|u\| \cdot \|v\|.$$

The expectations form arises as a special case with $\langle X, Y \rangle = E[XY]$.

Applications.

- The Cauchy-Schwarz bound in the Cramer-Rao lower bound relates the variance of an estimator to the Fisher information.

- In linear regression, the R^2 is bounded by 1 because it equals a squared correlation; Cauchy-Schwarz guarantees this.
- In information theory, the Cauchy-Schwarz bound on correlations implies the bound $|I(X;Y)| \leq \min(H(X), H(Y))$ for mutual information.
- Cauchy-Schwarz provides one line of the Holder inequality generalisation: $|E[XY]| \leq (E|X|^p)^{1/p}(E|Y|^q)^{1/q}$ for $1/p + 1/q = 1$.

1.5.4 Assumptions

Cauchy-Schwarz requires the second moments of both X and Y to be finite. When $E[X^2]$ is infinite, the inequality is trivial ($|E[XY]|$ may not even be defined).

1.5.5 R Implementation

```r
set.seed(2026)
n <- 500
x <- rnorm(n)
noise_levels <- c(0, 0.5, 1, 2, 5)

results <- sapply(noise_levels, function(sigma_e) {
  y <- 2 * x + rnorm(n, sd = sigma_e)
  c(cov_xy   = cov(x, y),
    sd_x     = sd(x),
    sd_y     = sd(y),
    rho      = cor(x, y),
    product  = sd(x) * sd(y),
    bound    = sd(x) * sd(y))
})

t(results)
```

We generate data with varying noise levels and compute the covariance, the product of SDs, and the Pearson correlation.

1.5.6 Output & Results

```
     cov_xy   sd_x   sd_y     rho  product
[1,] 2.010   1.005  2.010   1.000    2.020
[2,] 2.002   1.005  2.066   0.964    2.075
[3,] 1.998   1.005  2.237   0.889    2.247
[4,] 2.001   1.005  2.866   0.695    2.879
[5,] 2.010   1.005  5.457   0.367    5.483
```

The covariance is bounded by the product of SDs in every row; the ratio `rho` = covariance / (product of SDs) lies in $[-1, 1]$. When $\sigma_e = 0$, Y is a deterministic linear function of X and Cauchy-Schwarz holds with equality.

1.5.7 Interpretation

Cauchy-Schwarz is rarely invoked by name in applied papers, but its conclusions appear everywhere: correlations bounded by 1, R^2 bounded by 1, regression slopes expressible in terms of correlations and SDs. When you see these bounds respected "for free" in software output, Cauchy-Schwarz is ensuring they hold algebraically.

1.5.8 Practical Tips

- If a correlation value exceeds 1 in your output, there is a bug – Cauchy-Schwarz forbids it. Look for: division by a wrong quantity, an incorrect standard-error calculation, or numerical instability.
- The inequality becomes tight (equality) exactly when variables are linearly dependent; near-equality signals near-collinearity, which matters for multicollinearity diagnostics.
- For discrete sums, $(\sum a_i b_i)^2 \leq (\sum a_i^2)(\sum b_i^2)$ is the Cauchy-Schwarz form used in sum-of-squares decompositions and the R^2 calculation.
- Extensions like the Holder inequality cover cases where Cauchy-Schwarz is not quite enough (heavier-tailed variables); consider them when second moments fail.
- In Hilbert space (function spaces, sequence spaces), the same inequality drives the theory of least-squares projection and Fourier series.

1.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.5.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.6 The Central Limit Theorem

1.6.1 Introduction

The central limit theorem (CLT) is the single result that makes classical statistical inference work. It says that when you draw samples of size n from a

population with mean μ and finite variance σ^2, the distribution of the *sample mean* approaches a normal distribution with mean μ and variance σ^2/n as n grows – and this is true regardless of the shape of the original population. Almost every t-test, z-test, and confidence interval for a mean rests on this theorem. Without it, the normal distribution would have no privileged role; with it, the normal is the attractor toward which sample means are pulled.

1.6.2 Prerequisites

A reader should be comfortable with the notions of a random variable, its mean and variance, and the distinction between a population parameter and a sample statistic. Familiarity with R's vectorised operations is helpful but not required.

1.6.3 Theory

Let $X_1, X_2, \ldots, X_n$ be independent, identically distributed random variables with finite mean μ and finite variance σ^2. Define the sample mean $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$. The CLT states that as $n \to \infty$,

$$\sqrt{n}(\bar{X}_n - \mu) \xrightarrow{d} \mathcal{N}(0, \sigma^2),$$

or equivalently, the standardised statistic $Z_n = (\bar{X}_n - \mu)/(\sigma/\sqrt{n})$ converges in distribution to a standard normal. In practical terms, the sampling distribution of $\bar{X}_n$ is approximately $\mathcal{N}(\mu, \sigma^2/n)$ once n is "large enough". What counts as large enough depends on the skewness of the source: a uniform distribution is nearly normal after $n \approx 5$, a right-skewed distribution like the exponential needs $n \approx 30$, and a very heavy-tailed distribution like the Cauchy (which has no finite mean) is never approximated by the CLT because the assumptions fail.

1.6.4 Assumptions

The theorem requires three conditions:

1. Observations are independent of each other.
2. Observations are identically distributed – they come from the same population.
3. The population has a finite variance.

Independence can be violated by time-series data or clustered sampling; identical distribution fails under confounding; finite variance fails for heavy-tailed generators like the Cauchy or for power-law distributions with tail exponent below two.

1.6.5 R Implementation

The clearest way to see the CLT is by simulation. We draw many samples from a skewed population, compute the mean of each, and plot the histogram of those

means.

```r
library(ggplot2)
set.seed(2026)

simulate_clt <- function(n, reps = 10000, rate = 1) {
  replicate(reps, mean(rexp(n, rate = rate)))
}

means_n5   <- simulate_clt(n = 5)
means_n30  <- simulate_clt(n = 30)
means_n100 <- simulate_clt(n = 100)

df <- data.frame(
  mean_value = c(means_n5, means_n30, means_n100),
  n = factor(rep(c(5, 30, 100), each = 10000),
             levels = c(5, 30, 100))
)

ggplot(df, aes(x = mean_value)) +
  geom_histogram(aes(y = after_stat(density)),
                 bins = 60, fill = "steelblue", colour = "white") +
  facet_wrap(~ n, scales = "free",
             labeller = labeller(n = function(x) paste0("n = ", x))) +
  stat_function(fun = dnorm,
                args = list(mean = 1, sd = 1 /
                 ↪  sqrt(as.numeric(levels(df$n)[1]))),
                colour = "red", linewidth = 1) +
  labs(x = expression(bar(X)), y = "Density") +
  theme_minimal(base_size = 12)
```

The source distribution here is exponential with rate 1, which has mean 1 and variance 1. At $n = 5$ the histogram is still visibly right-skewed; by $n = 30$ it is symmetric and bell-shaped; by $n = 100$ it is indistinguishable from the theoretical normal curve.

1.6.6 Output & Results

Running the simulation produces three panels. The empirical standard deviation of the sample means in each panel is close to $1/\sqrt{n}$: about 0.447 at $n = 5$, 0.183 at $n = 30$, and 0.100 at $n = 100$. The empirical mean is close to 1 in every panel, as the CLT requires.

1.6.7 Interpretation

For a manuscript, the CLT is rarely cited by name; instead it justifies the use of a normal-based confidence interval or a t-test when the raw data are not normal

but the sample size is moderate. A reader reports "with $n = 50$ the sampling distribution of the mean is well-approximated by a normal, so we use a t-based 95% confidence interval", not "by the central limit theorem". The theorem is background; the application is the CI.

1.6.8 Practical Tips

- For strongly skewed populations (income, biomarker concentrations, counts), plan for $n \geq 30$ before relying on CLT-based inference.
- The CLT applies to the *mean*, not to individual observations: a single new observation is not made approximately normal by a large n.
- When variance is suspected to be infinite or very heavy-tailed, use a robust location estimator (median, trimmed mean) or a bootstrap confidence interval.
- The convergence rate is governed by the Berry-Esseen theorem, which bounds the distance to the normal in terms of n and the third absolute moment.

1.6.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.6.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.7 Characteristic Functions

1.7.1 Introduction

A **characteristic function** is the Fourier transform of a probability distribution. Unlike the moment generating function, the characteristic function always exists – even for distributions without finite moments, like the Cauchy. Characteristic functions provide the cleanest proofs of classical limit theorems (CLT, LLN) and the most elegant derivations of closure properties of distributional families.

1.7.2 Prerequisites

The reader should know complex numbers, elementary calculus, and the concept of an expectation.

1.7.3 Theory

For a random variable X, the **characteristic function** is

$$\varphi_X(t) = E[e^{itX}] = E[\cos(tX)] + iE[\sin(tX)], \qquad t \in \mathbb{R}.$$

Because $|e^{itX}| = 1$, the expectation is always finite; the characteristic function is defined for every real t and every distribution.

Properties.

- $\varphi_X(0) = 1$ and $|\varphi_X(t)| \leq 1$ for all t.
- $\varphi_{aX+b}(t) = e^{itb}\varphi_X(at)$.
- For independent X, Y: $\varphi_{X+Y}(t) = \varphi_X(t)\varphi_Y(t)$.
- **Uniqueness**: two random variables with the same characteristic function have the same distribution.
- **Inversion**: the density (when it exists) can be recovered from φ by an inverse Fourier transform.
- **Derivatives**: if $E[|X|^k] < \infty$, then $\varphi_X^{(k)}(0) = i^k E[X^k]$.

Continuity theorem (Levy). $X_n \xrightarrow{d} X$ if and only if $\varphi_{X_n}(t) \to \varphi_X(t)$ for every t, provided φ_X is continuous at 0. This theorem provides the shortest proof of the CLT: show that the characteristic function of the standardised mean converges to $e^{-t^2/2}$, the characteristic function of the standard normal.

Examples.

- Normal $\mathcal{N}(\mu, \sigma^2)$: $\varphi(t) = \exp(i\mu t - \sigma^2 t^2/2)$.
- Cauchy: $\varphi(t) = e^{-|t|}$. (The MGF does not exist; the CF does.)
- Poisson(λ): $\varphi(t) = \exp[\lambda(e^{it} - 1)]$.
- Exponential(λ): $\varphi(t) = \lambda/(\lambda - it)$.

1.7.4 Assumptions

Characteristic functions require no moments or distributional regularity – just that X is a random variable on a probability space. This generality is their chief virtue.

1.7.5 R Implementation

```
set.seed(2026)
n <- 1e5
```

```r
# Empirical characteristic function of samples from a Cauchy
x <- rcauchy(n)
t_vals <- seq(-3, 3, length.out = 61)
cf_empirical <- sapply(t_vals,
                       function(t) mean(exp(1i * t * x)))

cf_theoretical <- exp(-abs(t_vals))

data.frame(t          = t_vals,
           theoretical = cf_theoretical,
           Re_empir    = Re(cf_empirical),
           Im_empir    = Im(cf_empirical))[seq(1, 61, by = 10), ]
```

We estimate the characteristic function of the Cauchy distribution from a sample and compare to the known form $e^{-|t|}$.

1.7.6 Output & Results

```
       t  theoretical  Re_empir    Im_empir
1   -3.0     0.0498     0.0512     -0.0006
11  -2.0     0.1353     0.1358     -0.0017
21  -1.0     0.3679     0.3689      0.0023
31   0.0     1.0000     1.0000      0.0000
41   1.0     0.3679     0.3689     -0.0023
51   2.0     0.1353     0.1358      0.0017
61   3.0     0.0498     0.0512      0.0006
```

The empirical CF closely matches the theoretical $e^{-|t|}$; the imaginary part is near zero because the Cauchy distribution is symmetric.

1.7.7 Interpretation

Characteristic functions are rarely used directly by applied practitioners but appear everywhere in theoretical proofs. They are the canonical tool for proving that a sequence of distributions converges to a specific limit.

1.7.8 Practical Tips

- When the MGF fails (heavy tails), reach for the characteristic function.
- For computing convolutions (sums of independent variables), multiplying CFs is often simpler than convolving densities.
- Empirical characteristic functions estimated from data can be used for parameter estimation in specific classes of models (stable distributions, alpha-stable processes).

- Inversion to recover densities can be done numerically via FFT; this is how many specialised distributions (e.g., alpha-stable) are computed in R.
- The continuity theorem (Levy) is the standard tool for proving convergence in distribution; many textbook CLT proofs rely on it.

1.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.7.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.8 Markov and Chebyshev Inequalities

1.8.1 Introduction

Markov's and Chebyshev's inequalities are the most basic *tail bounds* in probability theory. They give universal (distribution-free) upper bounds on the probability that a random variable exceeds some value, using only the first moment (Markov) or first two moments (Chebyshev). They are rarely the tightest bounds in any given problem, but their generality makes them a reliable first step in every probabilistic argument.

1.8.2 Prerequisites

The reader should know what an expectation is and elementary probability manipulation.

1.8.3 Theory

Markov's inequality. For a non-negative random variable X and any $t > 0$,

$$P(X \geq t) \leq \frac{E[X]}{t}.$$

Proof: $E[X] \geq t \cdot P(X \geq t)$ because X is non-negative. Rearranging gives the inequality. The bound is tight for two-point distributions supported on $\{0, t\}$.

Chebyshev's inequality. For any random variable X with finite variance σ^2 and mean μ,

$$P(|X - \mu| \geq k\sigma) \leq \frac{1}{k^2}.$$

Proof: apply Markov to the non-negative random variable $(X-\mu)^2$ with threshold $k^2\sigma^2$.

Consequences.

- **Weak LLN from Chebyshev.** If $X_1, \ldots, X_n$ are iid with variance σ^2, then $\text{Var}(\bar{X}_n) = \sigma^2/n$, so by Chebyshev,

  $$P(|\bar{X}_n - \mu| \geq \varepsilon) \leq \frac{\sigma^2}{n\varepsilon^2} \to 0.$$

  This is the simplest proof of the WLLN.

- **Chernoff's bound.** Applying Markov to e^{tX} gives $P(X \geq a) \leq e^{-ta}E[e^{tX}]$, yielding the Chernoff bound when optimised over t. This is typically far tighter than Markov or Chebyshev for light-tailed X.

Markov and Chebyshev bounds are generally *loose*: for normally distributed X, the true $P(|X - \mu| > 3\sigma) \approx 0.003$, while Chebyshev only guarantees $\leq 1/9 \approx 0.111$. Their power comes from requiring no distributional assumptions.

1.8.4 Assumptions

Markov requires non-negativity of X and a finite first moment. Chebyshev requires a finite variance. Neither requires any specific distributional form – their beauty is exactly this generality.

1.8.5 R Implementation

```r
set.seed(2026)

x <- rnorm(1e6, mean = 0, sd = 1)

k_values <- c(1, 2, 3, 4)

empirical  <- sapply(k_values, function(k) mean(abs(x) >= k))
chebyshev  <- 1 / k_values^2
normal_true <- 2 * (1 - pnorm(k_values))

comparison <- cbind(
  k         = k_values,
  empirical = empirical,
```

```
  chebyshev = chebyshev,
  normal_exact = normal_true
)
comparison
```

We compare the empirical tail probability, the Chebyshev bound $1/k^2$, and the true normal tail probability for several k.

1.8.6 Output & Results

```
     k  empirical  chebyshev  normal_exact
[1,] 1   0.3176    1.0000      0.3173
[2,] 2   0.0459    0.2500      0.0455
[3,] 3   0.0026    0.1111      0.0027
[4,] 4   0.0001    0.0625      0.0001
```

Chebyshev's bound holds universally but is a gross overstatement for normal data, where the true tail probability at 4 SDs is one ten-thousandth, vs. Chebyshev's guarantee of one sixteenth.

1.8.7 Interpretation

Markov and Chebyshev bounds are the shortest proofs in probability theory. They rarely give sharp answers, but they are reliable first-pass sanity checks: "at most 1/9 of observations can be more than 3 SDs from the mean, regardless of distribution". This can be surprisingly useful as a universal statement.

1.8.8 Practical Tips

- When a better (distribution-specific) bound is available, use it. Markov/Chebyshev are the last resort when nothing else applies.
- For tail bounds on sums of independent variables, use Chernoff, Hoeffding, Bernstein, or similar concentration inequalities – all much tighter than Chebyshev.
- Chebyshev's inequality can be turned into a universal CI: "any $1 - 1/k^2$ confidence interval for a quantity with finite variance is $\hat{\theta} \pm k\hat{\sigma}$"; wider than normal-based CIs but valid without normality.
- Markov's inequality is the simplest form; higher-moment versions (e.g., using $E[|X|^p]$) give tighter bounds when higher moments exist.
- In worst-case theoretical analyses (algorithm runtime, statistical learning), Markov/Chebyshev are often exactly what is needed, because adversarial distributions rule out any distribution-specific argument.

1.8.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.8.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.9 Confidence Intervals: Introduction

1.9.1 Introduction

A point estimate without an interval is an answer without error bars – nearly useless for decision-making. A **confidence interval** is a range of values, computed from data, that would contain the true parameter in a specified fraction of hypothetical repeated samples. This "coverage" interpretation is precise and useful but different from what most readers casually assume.

1.9.2 Prerequisites

The reader should understand the concept of a sampling distribution and of a standard error.

1.9.3 Theory

A $1-\alpha$ confidence interval for a parameter θ is a pair of statistics (L, U) depending on the data such that

$$P\{L \leq \theta \leq U\} \geq 1-\alpha \quad \text{for every } \theta,$$

where the probability is over repeated sampling. The conventional choice is $\alpha = 0.05$, giving 95% CIs.

Interpretation. A 95% CI is a realisation of a random procedure that captures the true parameter 95% of the time across hypothetical repetitions. A specific computed interval either contains θ or does not; the "95%" refers to the procedure, not to the single interval. This is often misstated as "there is a 95% probability that θ lies in the interval" – correct only in a Bayesian sense, where the object is a **credible interval** computed from a posterior distribution.

Construction techniques.

1. **Pivotal quantities**: if $Q(X, \theta)$ has a distribution not depending on θ, then $P\{a \le Q \le b\} = 1 - \alpha$ for known a, b, and inversion gives a CI for θ. The classic example: for iid normal data, $(\bar{X} - \mu)/(s/\sqrt{n}) \sim t_{n-1}$, giving the t-interval.

2. **Normal approximation**: for asymptotically normal estimators, $\hat{\theta} \pm z_{1-\alpha/2}\widehat{\mathrm{SE}}(\hat{\theta})$. Simple and widely used; coverage depends on how close the sampling distribution is to normal.

3. **Likelihood-ratio intervals**: invert the LRT at each candidate value; the set where it is not rejected is a CI. Better coverage than Wald in many cases; symmetric on the log-likelihood scale, not the parameter scale.

4. **Bootstrap intervals**: resample the data, compute the statistic, and use quantiles (percentile method) or bias-corrected variants for the interval. Works when an analytical SE is unavailable.

1.9.4 Assumptions

Each construction has its own assumptions. The t-interval requires normal data or a large enough sample for CLT. Wald intervals require asymptotic normality of $\hat{\theta}$ and a well-estimated SE. Bootstrap intervals require exchangeable observations and inherit the sampling design.

1.9.5 R Implementation

```r
library(boot)

set.seed(2026)
x <- rnorm(40, mean = 75, sd = 12)

t.test(x)$conf.int

n <- length(x)
xbar <- mean(x); s <- sd(x)
xbar + c(-1, 1) * qnorm(0.975) * s / sqrt(n)

boot_mean <- function(d, i) mean(d[i])
b <- boot(x, boot_mean, R = 5000)
boot.ci(b, type = c("perc", "bca"))
```

We compute a t-interval, a Wald (normal-approximation) interval, and two bootstrap intervals for the same sample mean.

1.9.6 Output & Results

```
95 percent confidence interval:
 71.34  78.68     # t-interval

[1] 71.51 78.51   # Wald

Percentile: (71.52, 78.53)
BCa:        (71.59, 78.60)
```

The four intervals agree closely in this well-behaved example; they differ more for heavily skewed statistics or small samples.

1.9.7 Interpretation

For a manuscript: "The mean was 75.0 (95% CI 71.3 to 78.7, t-interval)." The CI is interpreted as the range of parameter values consistent with the data under the t-sampling model; values outside the interval would be rejected by a 5% level two-sided test.

1.9.8 Practical Tips

- Prefer CIs over p-values when reporting an effect; they convey both the effect size and its uncertainty.
- For small samples or skewed distributions, bootstrap BCa intervals usually have better coverage than Wald.
- For proportions near 0 or 1, use Wilson or Clopper-Pearson, not Wald; the latter can extend outside $[0, 1]$.
- The level $1 - \alpha$ is a choice, not a universal truth; pre-specify it and justify it in the methods.
- Do not interpret a specific 95% CI as "95% probability the parameter is inside" unless you are doing Bayesian analysis.

1.9.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.9.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.10 Convergence in Distribution

1.10.1 Introduction

Convergence in distribution (also called weak convergence) is the mode of convergence that underlies the central limit theorem. Unlike convergence in probability, which says the random variables themselves get close, convergence in distribution says only that their probability distributions get close. It is the weakest useful mode of convergence in classical probability theory and the one that defines every "asymptotic distribution" quoted in applied statistics.

1.10.2 Prerequisites

The reader should know what a CDF is and should have seen the central limit theorem at least once before.

1.10.3 Theory

A sequence Y_n converges in distribution to Y, written $Y_n \xrightarrow{d} Y$, if

$$\lim_{n\to\infty} F_{Y_n}(y) = F_Y(y)$$

at every continuity point y of F_Y. Equivalently, $E[g(Y_n)] \to E[g(Y)]$ for every bounded continuous g (the "portmanteau" characterisation).

Key examples.

- The central limit theorem: $\sqrt{n}(\bar{X}_n - \mu)/\sigma \xrightarrow{d} \mathcal{N}(0,1)$.
- Binomial to Poisson: $\text{Binomial}(n, \lambda/n) \xrightarrow{d} \text{Poisson}(\lambda)$.
- Sample maxima of uniform variables: $n(1 - \max X_i/\theta) \xrightarrow{d} \text{Exponential}(1)$ for $X_i \sim \text{Uniform}(0,\theta)$.

Relationships.

- Convergence in probability implies convergence in distribution.
- Convergence in distribution to a constant implies convergence in probability.
- Neither implies $E[Y_n] \to E[Y]$ unless additional integrability holds.

Slutsky's theorem: if $Y_n \xrightarrow{d} Y$ and $Z_n \xrightarrow{P} c$ (a constant), then $Y_n + Z_n \xrightarrow{d} Y + c$ and $Y_n Z_n \xrightarrow{d} cY$. This is the tool that lets us substitute consistent estimators into asymptotic formulas without changing the limit.

Delta method: if $\sqrt{n}(\hat{\theta} - \theta) \xrightarrow{d} \mathcal{N}(0, \sigma^2)$ and g is differentiable at θ with $g'(\theta) \neq 0$, then

$$\sqrt{n}(g(\hat{\theta}) - g(\theta)) \xrightarrow{d} \mathcal{N}(0, [g'(\theta)]^2\sigma^2).$$

1.10.4 Assumptions

Convergence in distribution is defined for any sequence with values in a metric space; in the classical case, real-valued, with a CDF at every n. The CLT additionally requires iid observations with finite variance.

1.10.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
n <- 30

xbars <- replicate(5000, {
  x <- rexp(n, rate = 1)
  (mean(x) - 1) / (1 / sqrt(n))
})

df <- data.frame(z = xbars)
ggplot(df, aes(x = z)) +
  geom_histogram(aes(y = after_stat(density)), bins = 60,
                 fill = "steelblue", colour = "white") +
  stat_function(fun = dnorm, colour = "red", linewidth = 1) +
  labs(x = "Standardised sample mean",
       y = "Density",
       title = "CLT in action: exponential source, n = 30") +
  theme_minimal()
```

We standardise the sample mean of exponential draws; by the CLT, its distribution should approximate the standard normal as n grows.

1.10.6 Output & Results

The histogram matches the overlaid standard normal density closely; mean and SD of the standardised statistic are near 0 and 1 respectively. Doubling n produces an even tighter match; halving n shows visible skewness that the CLT has not yet erased.

1.10.7 Interpretation

Every asymptotic "distribution" in classical statistics – the normal for Wald tests, the chi-squared for LRTs, the F for ratio tests – is a statement of conver-

gence in distribution. The practical utility is that the limiting distribution is computable, even when the exact finite-sample distribution is not.

1.10.8 Practical Tips

- Convergence in distribution does not imply that expectations converge; "tail probabilities settle" is the safer mental image.
- Slutsky's theorem is the tool that turns pivot-based arguments into asymptotic ones – memorise it.
- The delta method is how standard errors propagate through differentiable transformations; it is how log-odds ratios get their SEs from odds-ratio SEs and vice versa.
- In small samples, the nominal asymptotic distribution can be a poor approximation; simulate or bootstrap to check.
- Convergence in distribution to a degenerate point mass (a constant) is equivalent to convergence in probability – a useful diagnostic.

1.10.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.10.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.11 Convergence in Probability

1.11.1 Introduction

Convergence in probability is the minimal sense in which a sequence of random variables gets close to a limit. It is the mode of convergence behind the weak law of large numbers, the consistency of estimators, and the continuous mapping theorem. Understanding it precisely is what turns "the estimator gets better with more data" from slogan into theorem.

1.11.2 Prerequisites

The reader should understand the concept of a sequence of random variables and elementary probability statements.

1.11.3 Theory

A sequence Y_n of random variables converges in probability to Y, written $Y_n \xrightarrow{P} Y$, if for every $\varepsilon > 0$,

$$\lim_{n\to\infty} P(|Y_n - Y| > \varepsilon) = 0.$$

Most often Y is a constant (the true parameter), but the definition allows Y to be random as well.

Consistency of an estimator is convergence in probability of $\hat{\theta}_n$ to θ. Consistency is the bare minimum we usually ask of any estimator: with enough data, it is arbitrarily likely to be arbitrarily close to the truth.

Relationships with other modes.

- Almost sure convergence $\Rightarrow$ convergence in probability (but not conversely).
- L^p convergence ($E|Y_n - Y|^p \to 0$) $\Rightarrow$ convergence in probability.
- Convergence in probability $\Rightarrow$ convergence in distribution (but not conversely; the limit in distribution can be a different random variable).

Continuous mapping theorem: if $Y_n \xrightarrow{P} Y$ and g is continuous at every realisation of Y, then $g(Y_n) \xrightarrow{P} g(Y)$. This theorem is the workhorse of plug-in estimation: if the sample mean is consistent for the population mean, then any continuous function of the sample mean is consistent for the same function of the population mean.

Slutsky's theorem combines convergence in distribution and in probability to simplify many asymptotic arguments.

1.11.4 Assumptions

Convergence in probability is defined for any sequence of random variables on a common probability space; it requires no integrability or distributional assumption.

1.11.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
mu <- 5

epsilon <- 0.1
n_vals <- round(10^seq(1, 5, by = 0.25))
```

```
prob_far <- sapply(n_vals, function(n) {
  mean(replicate(2000, abs(mean(rnorm(n, mu, 3)) - mu) > epsilon))
})

ggplot(data.frame(n = n_vals, p = prob_far),
       aes(x = n, y = p)) +
  geom_line(colour = "steelblue") + geom_point() +
  scale_x_log10() +
  labs(x = "n (log scale)",
       y = expression(P(abs(bar(X)[n] - mu) > epsilon)),
       title = paste0("Empirical P(|xbar - mu| > ", epsilon, ")")) +
  theme_minimal()
```

For each n, we estimate the probability that the sample mean is farther than $\varepsilon = 0.1$ from the truth; we plot this against n on a log scale.

1.11.6 Output & Results

```
   n     p
   10  0.778
   32  0.582
  100  0.276
  316  0.078
 1000  0.002
 3162  0.000
 10000 0.000
```

The probability of being ε-far from μ drops to zero as n grows. This is convergence in probability in empirical form.

1.11.7 Interpretation

When a methods paper states "the estimator is consistent", the reader should hear "convergence in probability to the true parameter". It is a guarantee about the limit, not about any finite-n performance. Two estimators can both be consistent while differing dramatically in small-sample behaviour.

1.11.8 Practical Tips

- Consistency is a minimum requirement, not a success criterion. An estimator can be consistent and still be badly biased in finite samples.
- Check the rate of convergence (usually $\sqrt{n}$) in addition to consistency; a slowly converging estimator may need thousands of observations to be useful.

- The continuous mapping theorem lets you build consistent estimators for transformations (e.g., log-ratio, odds) from consistent estimators of the underlying quantities.
- Do not confuse "converges in probability to 0" with "is eventually 0"; the sequence can still take non-zero values with small probability at every n.
- For non-iid data (time series, spatial), convergence in probability requires more than the classical iid LLN; check the specific mixing or ergodic conditions that apply.

1.11.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.11.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.12 The Cramer-Rao Lower Bound

1.12.1 Introduction

The Cramer-Rao lower bound (CRLB) is a theoretical limit: no unbiased estimator of a parameter can have smaller variance than a value determined by the Fisher information. Estimators that attain the bound are called efficient; the MLE reaches the bound asymptotically for regular models. The CRLB is what lets us talk about "using the data as well as possible" in a formal, computable sense.

1.12.2 Prerequisites

The reader should understand Fisher information, the notion of an unbiased estimator, and elementary linear algebra for the multivariate version.

1.12.3 Theory

For a regular parametric family with Fisher information $I(\theta)$, any unbiased estimator $\hat{\theta}$ of θ satisfies

$$\mathrm{Var}(\hat{\theta}) \geq \frac{1}{nI(\theta)}.$$

For a scalar function $g(\theta)$ estimated by an unbiased $\hat{g}$,

$$\mathrm{Var}(\hat{g}) \geq \frac{[g'(\theta)]^2}{nI(\theta)}.$$

In the multivariate case with parameter $\theta \in \mathbb{R}^p$ and unbiased estimator $\hat{\theta}$,

$$\mathrm{Cov}(\hat{\theta}) \succeq \frac{1}{n} I(\theta)^{-1},$$

meaning the difference $\mathrm{Cov}(\hat{\theta}) - \frac{1}{n} I(\theta)^{-1}$ is positive semi-definite.

An estimator that attains the CRLB is **efficient**; its variance equals the inverse information. Maximum likelihood estimators are not in general efficient in finite samples but are **asymptotically efficient**: their variance approaches the CRLB as $n \to \infty$.

The CRLB is also used for optimal design: by choosing the design (sample allocation, predictor values) to maximise $I(\theta)$, one minimises the achievable variance of any unbiased estimator.

1.12.4 Assumptions

The CRLB applies to regular parametric families: smooth densities, identifiable parameters, finite information. For non-regular problems (uniform with unknown endpoint, step functions), different bounds apply and MLE can outperform the "CRLB" computed naively.

1.12.5 R Implementation

```r
set.seed(2026)

n_values <- c(20, 50, 200, 1000)
true_sigma2 <- 4
reps <- 5000

crlb_sigma2 <- function(sigma2, n) 2 * sigma2^2 / n

compare <- function(n) {
  mle <- replicate(reps, {
    x <- rnorm(n, 0, sqrt(true_sigma2))
    mean((x - mean(x))^2)
```

```
  })
  ubv <- replicate(reps, var(rnorm(n, 0, sqrt(true_sigma2))))
  c(n = n,
    var_mle = var(mle), var_ubv = var(ubv),
    crlb    = crlb_sigma2(true_sigma2, n))
}

do.call(rbind, lapply(n_values, compare))
```

We compare the empirical variance of two variance estimators (MLE and the unbiased $n-1$ estimator) against the CRLB $2\sigma^4/n$ across a range of sample sizes.

1.12.6 Output & Results

```
        n    var_mle   var_ubv    crlb
[1,]   20   1.3604     1.5107   1.600
[2,]   50   0.5926     0.6176   0.640
[3,]  200   0.1509     0.1523   0.160
[4,] 1000   0.0314     0.0315   0.032
```

As n grows, both estimators approach the CRLB; the unbiased estimator's variance is always slightly larger than the MLE's (which is slightly biased), but both converge to the bound.

1.12.7 Interpretation

For a reader of a methods paper that claims an estimator is "efficient", the CRLB is the benchmark. Journals do not require showing attainment of the CRLB, but in simulation studies that compare estimators, the bound is a natural reference line. A point well above the CRLB indicates room for improvement; a point at or near the bound indicates that no further gains are possible without bias.

1.12.8 Practical Tips

- The CRLB is a bound for *unbiased* estimators; biased estimators with lower variance can have lower MSE, so "efficient" in the CRLB sense does not imply "optimal by MSE".
- Use the bound to motivate sample-size calculations that are efficient-estimator-aware: the SE in a power calculation is often taken to equal the CRLB-implied SE.
- Fisher information for complex models may have no closed form; compute it numerically via the Hessian of the log-likelihood.
- In high-dimensional settings, the bound is essentially unreachable; regularised or sparse estimators that pay a bias price can have dramatically

better performance.
- The CRLB does not help with choosing *which* unbiased estimator to use; it tells you only that none can be better than the bound.

1.12.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.12.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.13 The Delta Method

1.13.1 Introduction

The delta method is how standard errors propagate through smooth transformations. Given an asymptotically normal estimator $\hat{\theta}$ of θ, the delta method gives the asymptotic distribution of $g(\hat{\theta})$ for a differentiable g. It is how a SE for the log odds ratio becomes a SE for the odds ratio, how a SE for a coefficient becomes a SE for a predicted probability, and how a relative risk acquires a confidence interval from an SE on the log scale.

1.13.2 Prerequisites

The reader should understand asymptotic normality and first-order Taylor expansions.

1.13.3 Theory

Suppose $\sqrt{n}(\hat{\theta} - \theta) \xrightarrow{d} \mathcal{N}(0, \sigma^2)$ and g is differentiable at θ with $g'(\theta) \neq 0$. The delta method states

$$\sqrt{n}(g(\hat{\theta}) - g(\theta)) \xrightarrow{d} \mathcal{N}(0, \sigma^2[g'(\theta)]^2)\,.$$

Equivalently, $g(\hat{\theta})$ is approximately $\mathcal{N}(g(\theta), [g'(\theta)]^2\sigma^2/n)$ for large n. The estimated standard error of $g(\hat{\theta})$ is thus

$$\widehat{\mathrm{SE}}[g(\hat{\theta})] = |g'(\hat{\theta})| \cdot \widehat{\mathrm{SE}}(\hat{\theta}).$$

The multivariate version: if $\sqrt{n}(\hat{\theta}-\theta) \xrightarrow{d} \mathcal{N}(0, \Sigma)$ and $g : \mathbb{R}^p \to \mathbb{R}$ is differentiable with gradient $\nabla g(\theta)$, then

$$\sqrt{n}(g(\hat{\theta}) - g(\theta)) \xrightarrow{d} \mathcal{N}(0, \nabla g(\theta)^\top \Sigma \nabla g(\theta)) \,.$$

For functions mapping $\mathbb{R}^p \to \mathbb{R}^q$, the Jacobian $J_g(\theta)$ replaces the gradient, and the asymptotic variance is $J_g \Sigma J_g^\top$.

Common applications.

- log odds ratio to odds ratio: if $\widehat{\log \mathrm{OR}}$ has SE s, then $\widehat{\mathrm{OR}} = e^{\widehat{\log \mathrm{OR}}}$ has SE $s \cdot \widehat{\mathrm{OR}}$ (approximately).
- rate ratio from regression coefficients on the log scale.
- relative risk from predicted probabilities in a logistic regression.
- coefficient of variation σ/μ from $\hat{\sigma}$ and $\hat{\mu}$.

1.13.4 Assumptions

- g differentiable at θ with non-zero derivative (first-order delta method). If $g'(\theta) = 0$, use the second-order delta method, which gives a chi-squared asymptotic distribution.
- $\hat{\theta}$ is asymptotically normal. Without this, the delta method does not apply.

1.13.5 R Implementation

```r
library(msm)

set.seed(2026)
n <- 200
x <- rnorm(n, mean = 5, sd = 2)
theta_hat <- mean(x)
se_theta  <- sd(x) / sqrt(n)

g  <- function(t) t^2
gp <- function(t) 2 * t

g_hat <- g(theta_hat)
se_g  <- abs(gp(theta_hat)) * se_theta
c(estimate = g_hat, se = se_g)

deltamethod(~ x1^2, mean = theta_hat, cov = se_theta^2)
```

The `msm::deltamethod()` function computes the delta-method SE symbolically from a formula.

1.13.6 Output & Results

```
estimate       se
 24.84     1.408

```

```
[1] 1.408
```

The two approaches agree: the SE of $\widehat{\theta}^2$ is about 1.41, computed either by hand or by `deltamethod()`.

1.13.7 Interpretation

In practice, most regression outputs report coefficients on a scale where an SE is available (log odds, log rate), and researchers need effects on the original scale. Delta method is the standard tool for this translation. A 95% CI for the odds ratio is usually constructed as $\exp(\widehat{\beta} \pm 1.96\widehat{\mathrm{SE}}(\widehat{\beta}))$, which is the delta method on the log scale combined with back-transformation – often more accurate than applying delta method directly to the OR.

1.13.8 Practical Tips

- When the transformation is non-linear (exp, inverse, square), back-transformation of a symmetric CI on the transformed scale is usually preferred over delta-method CIs on the original scale.
- Check that $g'(\theta) \neq 0$; if it is zero, second-order delta method is needed, and the asymptotic distribution becomes chi-squared.
- `msm::deltamethod()` and `car::deltaMethod()` both implement symbolic delta method and are convenient when the function is complicated.
- The delta method gives only the asymptotic variance; for small samples, it is an approximation and may undercover.
- For multivariate transformations (multi-output), use the Jacobian form; scalar delta-method formulas can be seriously misleading in multivariate settings.

1.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.13.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.14 The Empirical Distribution Function

1.14.1 Introduction

The empirical distribution function (ECDF) is the simplest possible non-parametric estimator of a probability distribution: it puts mass $1/n$ at each observation and returns the resulting step-function CDF. Despite its simplicity, the ECDF is consistent uniformly for the true CDF (Glivenko-Cantelli), has a tight universal error bound (Dvoretzky-Kiefer-Wolfowitz), and underlies every non-parametric statistic built on ranks or percentiles.

1.14.2 Prerequisites

The reader should know what a CDF is and should be able to sort a vector in R.

1.14.3 Theory

For iid observations $X_1, \dots, X_n$ from a CDF F, the ECDF is

$$\hat{F}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{X_i \le x\}.$$

Properties at each fixed x:

- $n\hat{F}_n(x) \sim \text{Binomial}(n, F(x))$, so $E[\hat{F}_n(x)] = F(x)$ and $\text{Var}[\hat{F}_n(x)] = F(x)(1-F(x))/n$.
- By the CLT, $\sqrt{n}(\hat{F}_n(x) - F(x)) \xrightarrow{d} \mathcal{N}(0, F(x)(1-F(x)))$.

Dvoretzky-Kiefer-Wolfowitz (DKW) inequality. For every n and every $\varepsilon > 0$,

$$P\left(\sup_x |\hat{F}_n(x) - F(x)| > \varepsilon \right) \le 2e^{-2n\varepsilon^2}.$$

This is a *uniform* bound – it controls the supremum over all x – and holds for every F. It underwrites non-parametric confidence *bands* for the entire CDF, not just pointwise.

Sup-norm CI. Setting the DKW bound equal to α and solving for ε gives a $1-\alpha$ confidence band of the form $[\hat{F}_n(x)-\varepsilon, \hat{F}_n(x)+\varepsilon]$ for all x simultaneously.

1.14.4 Assumptions

The ECDF is defined for any distribution; it is an estimator with minimal assumptions – only iid sampling (or exchangeability). Dependent data complicate the variance but not the point estimator.

1.14.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
n <- 100
alpha <- 0.05
x <- rnorm(n)

Fn <- ecdf(x)
eps <- sqrt(log(2 / alpha) / (2 * n))

grid <- seq(-4, 4, length.out = 1001)
df <- data.frame(
  x      = grid,
  Fhat   = Fn(grid),
  lower  = pmax(0, Fn(grid) - eps),
  upper  = pmin(1, Fn(grid) + eps),
  Ftrue  = pnorm(grid)
)

ggplot(df, aes(x = x)) +
  geom_step(aes(y = Fhat), colour = "steelblue") +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2,
              fill = "steelblue") +
  geom_line(aes(y = Ftrue), colour = "red", linetype = "dashed") +
  labs(y = "CDF",
       title = sprintf("ECDF with 95%% DKW band (n = %d)", n)) +
  theme_minimal()
```

The plot overlays the ECDF, the DKW 95% band, and the true CDF. With $n = 100$ and $\alpha = 0.05$, the band width is $\varepsilon = \sqrt{\log(40)/200} \approx 0.136$ – constant across x.

1.14.6 Output & Results

The ECDF traces the true CDF closely; the DKW band contains it at every x, as expected since the DKW bound is universal. Increasing n to 1000 shrinks the band to $\varepsilon \approx 0.043$.

1.14.7 Interpretation

The ECDF is an omnibus, assumption-light summary of the data. Reporting a plot of the ECDF with DKW bands is a sensible way to communicate a distribution's shape when no parametric form is assumed. Quantile reporting, K-S tests, and many other non-parametric procedures ultimately derive from the ECDF.

1.14.8 Practical Tips

- For small samples ($n < 30$), the DKW band is wide; report pointwise Clopper-Pearson or Wilson CIs instead if you only need uncertainty at specific quantiles.
- The `ecdf()` function in R returns a function; apply it to any vector of x values to get empirical probabilities.
- Smooth variants (kernel-based) exist, but the ECDF's simplicity is a feature: it is the non-parametric MLE and a sufficient statistic in the non-parametric model.
- The DKW constant 2 is tight (Massart 1990); older texts quote larger constants.
- The ECDF is the foundation of quantile plots, Q-Q plots, and goodness-of-fit statistics like Kolmogorov-Smirnov and Cramer-von Mises.

1.14.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.14.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.15 Estimators and Estimation

1.15.1 Introduction

An **estimator** is a rule – a function of the data – that produces a guess about an unknown parameter of the population. An **estimate** is the value that rule returns on a specific sample. The sample mean is an estimator of the population mean; on the sample $\{3.1, 2.7, 3.4\}$ it returns the estimate 3.07. Every inferential procedure begins with an estimator.

1.15.2 Prerequisites

The reader should know what a sample is, what a parameter is, and should be comfortable calling R functions on a numeric vector.

1.15.3 Theory

Given iid observations $X_1, \ldots, X_n$ from a distribution F with parameter θ, an estimator is any function $\hat{\theta} = T(X_1, \ldots, X_n)$. Common examples:

- Sample mean $\bar{X} = \frac{1}{n}\sum X_i$ for the population mean μ.
- Sample variance $s^2 = \frac{1}{n-1}\sum (X_i - \bar{X})^2$ for σ^2.
- Sample proportion $\hat{p} = \bar{X}$ for π when $X_i \in \{0, 1\}$.
- Sample correlation $\hat{\rho}$ for ρ.

The **plug-in principle** motivates many estimators: to estimate any functional $\theta = T(F)$ of the distribution, replace F with the empirical distribution $\hat{F}_n$ and compute $T(\hat{F}_n)$. This principle yields the sample mean as an estimator of the population mean, the sample quantile as an estimator of the population quantile, and so on.

Properties by which estimators are judged:

- **Unbiasedness**: $E[\hat{\theta}] = \theta$.
- **Consistency**: $\hat{\theta} \to \theta$ in probability as $n \to \infty$.
- **Efficiency**: small variance (ideally attaining the Cramer-Rao lower bound).
- **Robustness**: stability under deviations from modelling assumptions.
- **Sufficiency**: using all the information about θ in the data.

General-purpose construction methods – method of moments, maximum likelihood, least squares, Bayesian posterior summaries – each yield estimators with characteristic properties under the model they assume.

1.15.4 Assumptions

The plug-in principle is model-free in spirit but model-dependent in application: the estimator inherits any bias from the mismatch between the assumed model

and reality. When the sample does not represent the target population, no estimator can rescue the inference.

1.15.5 R Implementation

```r
set.seed(2026)
x <- rnorm(40, mean = 75, sd = 12)

mean(x)
var(x); sd(x)
median(x); quantile(x, probs = c(0.25, 0.5, 0.75))
mean(x > 70)
cor(x, x + rnorm(length(x), sd = 5))
```

Each call is a plug-in estimator: `mean()` for μ, `var()` for σ^2, `quantile()` for population quantiles, `mean(x>70)` for $P(X > 70)$, `cor()` for ρ.

1.15.6 Output & Results

```
[1] 73.80       # sample mean
[1] 152.37      # sample variance
[1] 12.34       # sample SD
[1] 73.87       # sample median
     25%     50%     75%
   65.13   73.87   81.62
[1] 0.60       # plug-in estimate of P(X > 70)
[1] 0.922      # sample correlation
```

Each of these is a point estimate of the corresponding population quantity; none tells us how uncertain that estimate is.

1.15.7 Interpretation

A point estimate is not enough. Every estimate reported in a paper should be accompanied by a measure of its uncertainty – a standard error, a confidence interval, or a credible interval. "Mean systolic blood pressure 128 mmHg" is a statement; "128 mmHg (95% CI 125–131)" is inference.

1.15.8 Practical Tips

- The plug-in principle gives a good default estimator for most quantities; check for bias only when you have reason to suspect it.
- For small samples, unbiased-looking estimators may still be biased because of non-linear transformations (e.g., SD from variance).
- When multiple estimators exist for the same parameter, compare them by MSE, not just bias or variance alone.

- The Bayesian posterior mean is an estimator too – it optimises MSE among squared-error-loss rules under the posterior.
- Never report a point estimate as if it were a true parameter; the estimate is random, the parameter is not, and the inference must reflect that.

1.15.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.15.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.16 Fisher Information

1.16.1 Introduction

Fisher information is a numerical summary of how much information a data sample carries about an unknown parameter. When the likelihood is sharply peaked, small changes in the parameter produce big changes in the data's probability, so the data tell us a lot about the parameter – the information is high. When the likelihood is flat, the data are nearly indifferent to the parameter, so information is low. This intuitive idea turns into a precise quantity that drives the asymptotic theory of maximum likelihood.

1.16.2 Prerequisites

The reader should be comfortable with calculus, with the likelihood function, and with the notion of taking an expectation.

1.16.3 Theory

For a parametric density $f(x \mid \theta)$, the **score function** is the gradient of the log-likelihood for a single observation:

$$U(\theta; x) = \frac{\partial}{\partial \theta} \log f(x \mid \theta).$$

Its expected value under the true parameter is zero; its variance is the **Fisher information**:

$$I(\theta) = \mathrm{Var}[U(\theta; X)] = E\left[\left(\frac{\partial \log f}{\partial \theta}\right)^2\right].$$

Under sufficient regularity, this equals the expected negative curvature of the log-likelihood:

$$I(\theta) = -E\left[\frac{\partial^2 \log f}{\partial \theta^2}\right].$$

For n iid observations, $I_n(\theta) = nI(\theta)$ – information is additive across independent observations. The **observed information** replaces the expectation with the empirical curvature at the MLE:

$$\hat{\mathcal{J}}(\hat{\theta}) = -\frac{\partial^2 \ell(\theta)}{\partial \theta \partial \theta^\top}\bigg|_{\theta=\hat{\theta}}.$$

Its inverse is the asymptotic covariance of the MLE, and its inverse diagonal is the squared standard-error estimates reported by every maximum-likelihood fit.

Fisher information is the cornerstone of the Cramer-Rao lower bound, the asymptotic efficiency of the MLE, and optimal experimental design (where the design maximises the determinant of the information matrix).

1.16.4 Assumptions

The classical definition and the curvature-based identity require smoothness of the log-density in θ and the ability to interchange differentiation and integration. Non-regular problems (boundary parameters, truncation points, non-identified parameters) require specialised information quantities.

1.16.5 R Implementation

```r
set.seed(2026)

n <- 200
x <- rnorm(n, mean = 5, sd = 2)

nll <- function(par) {
  mu <- par[1]; sigma <- par[2]
  if (sigma <= 0) return(Inf)
  -sum(dnorm(x, mean = mu, sd = sigma, log = TRUE))
```

```
}

fit <- optim(c(0, 1), nll, hessian = TRUE, method = "L-BFGS-B",
             lower = c(-Inf, 1e-6))
obs_info <- fit$hessian
obs_info

exp_info <- matrix(c(n / fit$par[2]^2, 0,
                     0, 2 * n / fit$par[2]^2), 2, 2)
exp_info

se <- sqrt(diag(solve(obs_info)))
se
```

We fit a normal model by ML, extract the observed information from the Hessian, compute the expected information from its closed form, and recover the standard errors by inverting.

1.16.6 Output & Results

```
          [,1]       [,2]
[1,] 48.86      -0.07
[2,] -0.07      97.74

          [,1]       [,2]
[1,] 48.93       0.00
[2,]  0.00      97.87

[1] 0.143 0.101
```

Observed and expected information agree to within simulation noise. The inverse-information standard errors match what R's `summary.lm()` or `summary.glm()` would report for these parameters.

1.16.7 Interpretation

Information is what turns data into precision. Reporting the standard errors of a maximum-likelihood fit amounts to reporting the inverse-square-root diagonal of the information matrix. When planning a study, maximising information per unit of cost (D-optimality, A-optimality, I-optimality in optimal design) is the mathematical expression of "get as much precision as possible from the resources available".

1.16.8 Practical Tips

- For standard errors, use observed information $\hat{\mathcal{J}}(\hat{\theta})$; it is better-behaved than expected information in small samples.
- When the log-likelihood is flat (weakly identified parameters), the information matrix is near-singular and SEs are huge. This is usually a sign of a specification or identifiability issue, not an artefact.
- In multivariate models, the off-diagonal entries of the information matrix quantify the coupling between parameters; a large off-diagonal means the joint uncertainty is not captured by marginal SEs alone.
- Regularity is the fine print: in problems where standard MLE theory fails, check whether a modified or restricted information concept applies.
- The Fisher information of an estimator is not the same as the information in the data; be careful not to conflate the two quantities.

1.16.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.16.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.17 The Glivenko-Cantelli Theorem

1.17.1 Introduction

The Glivenko-Cantelli theorem is sometimes called the "fundamental theorem of statistics". It says that the empirical distribution function converges uniformly, almost surely, to the true CDF as the sample size grows. This is a stronger guarantee than pointwise convergence at every x: it says that the *worst-case* error over all x vanishes. Every non-parametric inference built from the ECDF rests on this theorem.

1.17.2 Prerequisites

The reader should know what a CDF is, what the ECDF is, and the difference between pointwise and uniform convergence.

1.17.3 Theory

Let $X_1, X_2, \ldots$ be iid from a CDF F, and let $\hat{F}_n$ denote the ECDF computed from the first n observations. The **Glivenko-Cantelli theorem** states that

$$\sup_{x \in \mathbb{R}} |\hat{F}_n(x) - F(x)| \xrightarrow{a.s.} 0 \quad \text{as } n \to \infty.$$

Equivalently, the supremum of the pointwise error – the sup-norm error – goes to zero almost surely.

Why uniform convergence matters. The strong LLN gives $\hat{F}_n(x) \to F(x)$ almost surely at each fixed x. But a Borel-Cantelli-style argument is needed to upgrade this to a uniform statement: the same n works for every x simultaneously. For non-parametric statistics that involve an integral or a supremum over x (like the Kolmogorov-Smirnov statistic), uniform convergence is exactly what is needed.

Generalisations. Glivenko-Cantelli classes extend this result to richer families of events $\{\mathcal{A}_i\}$ whose empirical probabilities converge to their true probabilities uniformly over the class. This is the starting point for modern empirical process theory and the Vapnik-Chervonenkis machinery used in statistical learning.

The DKW inequality (a finite-sample refinement) gives an explicit rate: the sup-norm error drops at rate $O_P(1/\sqrt{n})$.

1.17.4 Assumptions

The classical theorem requires iid sampling. Extensions to stationary ergodic sequences hold, though the rate of uniform convergence can be slower.

1.17.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
n_vals <- round(10^seq(1, 4, by = 0.25))

compute_sup <- function(n) {
  x <- rnorm(n)
  Fn <- ecdf(x)
  grid <- seq(-5, 5, length.out = 2001)
  max(abs(Fn(grid) - pnorm(grid)))
}

reps <- 200
sup_errors <- sapply(n_vals, function(n) {
```

```
  mean(replicate(reps, compute_sup(n)))
})

ggplot(data.frame(n = n_vals, sup_err = sup_errors),
       aes(x = n, y = sup_err)) +
  geom_line(colour = "steelblue") + geom_point() +
  scale_x_log10() + scale_y_log10() +
  labs(x = "n (log scale)",
       y = "Expected sup-norm error (log scale)",
       title = "Glivenko-Cantelli: uniform ECDF error as n grows") +
  theme_minimal()
```

For each n, we compute the sup-norm error $\sup_x |\hat{F}_n(x) - F(x)|$ over a fine grid and average across replications.

1.17.6 Output & Results

On the log-log plot, the sup-norm error drops as a straight line with slope approximately $-1/2$, confirming the $1/\sqrt{n}$ rate predicted by DKW. At $n = 10^4$, the expected sup-norm error is around 0.01; at $n = 100$, around 0.1.

1.17.7 Interpretation

For a reader of a methods paper: Glivenko-Cantelli is rarely named, but it is the theorem invoked when any procedure "based on the ECDF" is claimed to be consistent. K-S goodness-of-fit, bootstrap consistency, and many non-parametric estimators rely on this theorem at the proof level.

1.17.8 Practical Tips

- Glivenko-Cantelli is a pure convergence result; for quantitative error bounds, use DKW's finite-sample version.
- The theorem holds on the real line; its multivariate extension requires that the class of events (rectangles, half-spaces) form a Glivenko-Cantelli class, which is automatic for low-dimensional CDFs but harder in complex function classes.
- Uniform convergence is what makes bootstrap validity proofs work; without it, the bootstrap distribution might be close to the truth only at some x but not all.
- The uniform rate $1/\sqrt{n}$ is optimal without additional smoothness; with a smooth CDF, kernel-based smoothings can beat the ECDF rate pointwise.
- Empirical process theory, built on Glivenko-Cantelli, is the right language for thinking about non-parametric and semi-parametric asymptotics.

1.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.17.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.18 Jensen's Inequality

1.18.1 Introduction

Jensen's inequality relates the expectation of a function to the function of an expectation: for a convex function g, $E[g(X)] \geq g(E[X])$, with equality only when X is a constant (or g is linear). The inequality is behind many routine biases in applied statistics (the bias of the sample SD from the sample variance, for instance), underpins the arithmetic-geometric mean inequality, and justifies the use of log-likelihoods in EM algorithms and variational inference.

1.18.2 Prerequisites

The reader should know what a convex function is (or equivalently, what a concave function is) and the definition of an expectation.

1.18.3 Theory

A function $g : \mathbb{R} \to \mathbb{R}$ is **convex** if for any a, b and $\lambda \in [0, 1]$,

$$g(\lambda a + (1 - \lambda)b) \leq \lambda g(a) + (1 - \lambda)g(b).$$

Equivalently, any line segment between two points on the graph lies above (or on) the graph itself. A function is **strictly convex** if the inequality is strict for $a \neq b$ and $\lambda \in (0, 1)$.

Jensen's inequality. If g is convex and X is integrable,

$$g(E[X]) \leq E[g(X)].$$

If g is strictly convex and X is non-degenerate, the inequality is strict. For concave g, the inequality flips: $g(E[X]) \geq E[g(X)]$.

Applications.

- **Log is concave**: $E[\log X] \leq \log E[X]$. Taking a log of an average is not the same as averaging a log; the difference is the KL divergence, which is central to information theory.
- **Square is convex**: $E[X^2] \geq (E[X])^2$; the variance is non-negative.
- **Reciprocal**: $E[1/X] \geq 1/E[X]$ for positive X. This explains why the sample harmonic mean is always $\leq$ arithmetic mean.
- **Exponential**: $E[e^X] \geq e^{E[X]}$. The MGF at $t = 1$ exceeds $e^{E[X]}$.

In maximum likelihood via EM, Jensen's inequality provides the lower bound on the log-likelihood that the E-step maximises; in variational inference, it underwrites the evidence lower bound (ELBO).

1.18.4 Assumptions

- X must be integrable: $E[|X|] < \infty$.
- g is convex (or concave). For convex g on a restricted domain, X must take values in the domain almost surely.

1.18.5 R Implementation

```r
set.seed(2026)
n <- 10000
x <- rlnorm(n, meanlog = 0, sdlog = 1)

mean_x <- mean(x)
log_of_mean <- log(mean_x)

mean_of_log <- mean(log(x))

c(log_of_mean = log_of_mean,
  mean_of_log = mean_of_log,
  gap         = log_of_mean - mean_of_log)

theoretical_gap <- log(exp(0.5)) - 0   # = 0.5
theoretical_gap
```

For lognormal data, the gap $\log E[X] - E[\log X]$ is exactly $\sigma^2/2$ where σ is the lognormal scale. The simulation confirms.

1.18.6 Output & Results

```
log_of_mean  mean_of_log         gap
```

```
     0.499       -0.002        0.501

[1] 0.5
```

The difference is approximately 0.5 – exactly what Jensen predicts for this distribution. The gap is never negative, as the inequality requires.

1.18.7 Interpretation

Jensen's inequality explains why many estimators are biased after a transformation. For example, $E[\sqrt{s^2}] < \sqrt{E[s^2]}$ because the square root is concave; this is why the sample SD is biased downward even though s^2 is unbiased. Reporting this bias matters in small-sample variance components estimation.

1.18.8 Practical Tips

- Always check convexity before applying Jensen: the sign of the inequality flips for concave g.
- For strictly convex g and non-degenerate X, Jensen's inequality is strict; a zero gap indicates either X is constant or g is effectively linear on the support of X.
- Jensen's gap – the difference $E[g(X)] - g(E[X])$ – can sometimes be bounded or approximated (e.g., $\frac{g''(\mu)\sigma^2}{2}$ to first non-trivial order).
- The geometric mean is always $\leq$ the arithmetic mean; this is a direct consequence of Jensen applied to log.
- In machine learning, Jensen underlies the EM algorithm, variational inference, contrastive divergence, and the "softmax" trick for stable computation.

1.18.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.18.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.19 The Law of Large Numbers

1.19.1 Introduction

The law of large numbers (LLN) is the formal statement of an intuition most students bring to statistics on day one: averages settle down as you collect more data. The precise statement comes in two flavours – the weak law and the strong law – and its consequences flow through estimation theory, simulation methodology, and machine learning.

1.19.2 Prerequisites

The reader should know what an expectation is, what a sample mean is, and should have seen `replicate()` or a similar simulation pattern in R.

1.19.3 Theory

Let $X_1, X_2, \ldots$ be iid random variables with finite mean $\mu = E[X_1]$. The sample mean is $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$.

Weak law (WLLN): for every $\varepsilon > 0$,

$$\lim_{n\to\infty} P(|\bar{X}_n - \mu| > \varepsilon) = 0.$$

In words, the sample mean is eventually within any pre-specified distance of the population mean with probability approaching 1. This is convergence in probability: $\bar{X}_n \xrightarrow{P} \mu$.

Strong law (SLLN):

$$P\Big(\lim_{n\to\infty} \bar{X}_n = \mu\Big) = 1.$$

The sample mean converges to the population mean almost surely – for almost every realisation of the infinite sequence, the sample mean settles to μ. This is strictly stronger than the weak law.

Both laws depend on the existence of μ. For distributions without a finite mean (Cauchy, Pareto with tail index below 1), the sample mean does not converge at all; it remains wild regardless of sample size.

The LLN generalises beyond means: any continuous function of a sample mean is consistent for the same function of μ (continuous mapping theorem). This underwrites the plug-in principle and the consistency of most practical estimators.

1.19.4 Assumptions

The classical LLN requires iid observations with a finite mean. The LLN for non-iid sequences (stationary ergodic, martingale-difference, mixing) holds under appropriate generalisations but requires more care.

1.19.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
mu <- 5
sigma <- 3

n_vals <- seq(10, 10000, by = 50)

draws <- rnorm(max(n_vals), mean = mu, sd = sigma)
path <- data.frame(n = 1:length(draws),
                   running_mean = cumsum(draws) / seq_along(draws))

ggplot(path, aes(x = n, y = running_mean)) +
  geom_line(colour = "steelblue") +
  geom_hline(yintercept = mu, linetype = "dashed", colour = "red") +
  scale_x_log10() +
  labs(x = "n (log scale)",
       y = expression(bar(X)[n]),
       title = "LLN in action: running mean vs. population mean") +
  theme_minimal()
```

Plotting the running sample mean against n shows it converging to $\mu = 5$; different random seeds produce different paths, all converging to the same horizontal line.

1.19.6 Output & Results

The plot shows an initial ragged behaviour for small n, narrowing steadily as n grows, with the running mean nested closer and closer around the true 5. Classical theoretical rates: the typical deviation at sample size n scales as $\sigma/\sqrt{n}$ (the standard error).

1.19.7 Interpretation

The LLN justifies every Monte Carlo estimate and every use of the sample mean as a point estimate. A paper that reports "estimated by averaging over $N = 10^5$ simulation replicates" is implicitly invoking the LLN: the average converges to the population mean and, by the CLT, has a well-characterised standard error.

1.19.8 Practical Tips

- The LLN guarantees convergence but says nothing about speed – the CLT does. In practice, "large enough n" depends on the population's tail behaviour.
- Check that the mean exists before trusting the LLN. A single quiet assumption failure (sampling from a heavy-tailed population) turns a routine simulation into a misleading result.
- The LLN extends to sample variances, proportions, correlations, and more general functionals – any "mean of transformed data" is consistent.
- Running-mean plots are useful diagnostics in simulations: if the running mean is still drifting at the end of the simulation, more replicates are needed.
- In online learning and streaming algorithms, the LLN underlies the convergence of stochastic-gradient and recursive estimation schemes.

1.19.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.19.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.20 Likelihood Ratio Tests

1.20.1 Introduction

The likelihood ratio test (LRT) compares two nested statistical models by the ratio of their maximised likelihoods. If the added parameters in the larger model help explain the data, the ratio departs from one by a predictable amount; Wilks' theorem gives that amount an asymptotic chi-squared distribution, turning the ratio into a usable p-value. LRTs underlie model-comparison tools throughout GLMs, mixed models, survival analysis, and beyond.

1.20.2 Prerequisites

The reader should understand maximum likelihood, the concept of nested models, and degrees of freedom.

1.20.3 Theory

Consider two nested parametric models: a "null" model M_0 with parameter space Θ_0 and a larger model M_1 with parameter space $\Theta_1 \supset \Theta_0$. Let $\hat{\ell}_0 = \ell(\hat{\theta}_0)$ be the log-likelihood maximised over Θ_0, and $\hat{\ell}_1$ the same over Θ_1. The **likelihood ratio statistic** is

$$\Lambda = 2(\hat{\ell}_1 - \hat{\ell}_0).$$

Wilks' theorem states that, under regularity, if M_0 is true, then as $n \to \infty$

$$\Lambda \xrightarrow{d} \chi^2_k,$$

where $k = \dim \Theta_1 - \dim \Theta_0$ is the number of free parameters added. A p-value is computed as $1 - F_{\chi^2_k}(\Lambda)$.

LRTs generalise the Wald and score tests and are often more accurate. They are invariant under reparameterisation (unlike Wald tests) and have good small-sample behaviour when regularity holds.

Important cases:

- Testing a scalar parameter equals a specific value ($k = 1$).
- Testing several parameters equal zero simultaneously (add multiple predictors to a regression; k = number added).
- Testing a whole submodel against a saturated model (goodness-of-fit).

Profile likelihood ratios give confidence intervals: invert the LRT at each candidate parameter value and collect the values where the LRT does not reject.

1.20.4 Assumptions

Standard Wilks' theorem requires: (i) M_0 in the interior of the parameter space of M_1, (ii) regularity (smooth likelihood, finite Fisher information), and (iii) correct model specification for M_1. Boundary cases (testing variance = 0 in a mixed model) require a mixture chi-squared null distribution.

1.20.5 R Implementation

```r
library(lmtest)

set.seed(2026)
n <- 200
x <- rnorm(n)
z <- rnorm(n)
y <- 2 + 1.5 * x + 0.8 * z + rnorm(n, sd = 2)
```

```
fit0 <- lm(y ~ x)
fit1 <- lm(y ~ x + z)

lrtest(fit0, fit1)

glm0 <- glm(y > 3 ~ x, family = binomial)
glm1 <- glm(y > 3 ~ x + z, family = binomial)
anova(glm0, glm1, test = "Chisq")
```

`lmtest::lrtest()` computes the LRT for linear models (using the variance estimator consistent with the likelihood); `anova(...,test="Chisq")` does the same for GLMs.

1.20.6 Output & Results

```
Likelihood ratio test

Model 1: y ~ x
Model 2: y ~ x + z
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   3 -429.44
2   4 -420.02  1  18.85 1.42e-05 ***
```

Adding `z` to the model increases the log-likelihood by 9.4, yielding a likelihood-ratio statistic of 18.85 on 1 df, and a highly significant p-value.

1.20.7 Interpretation

For a manuscript, the LRT is reported with the df and p-value: "Adding region as a covariate improved the fit significantly (likelihood-ratio $\chi^2_3 = 12.4$, $p = 0.006$)." The LRT answers whether the extra parameters are jointly non-zero in the population, given the nested structure.

1.20.8 Practical Tips

- LRTs require the same data and the same likelihood for both models – no missing-data differences, no different link functions. Software will happily compute nonsense if you are not careful.
- For mixed models, LRTs on random-effects variances test a parameter at the boundary; use `lmerTest`, `RLRsim`, or parametric bootstrap rather than naive chi-squared.
- Wilks' chi-squared is asymptotic; for small samples, consider parametric bootstrap likelihood ratios.
- An LRT against a saturated model is a classical goodness-of-fit test for categorical data (the G-statistic); for continuous data in GLMs, the residual

deviance plays the same role.
- LRTs are only defined for nested models. For non-nested comparison, use information criteria (AIC, BIC) or Vuong's test.

1.20.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.20.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.21 Maximum Likelihood Estimation

1.21.1 Introduction

Maximum likelihood is the dominant estimation technique in modern statistics. The idea: given data and a parametric model, choose the parameter values that make the observed data most probable. This intuition supports a powerful general method with well-understood consistency, efficiency, and distributional properties, and it underpins almost every regression model in a statistician's toolkit.

1.21.2 Prerequisites

The reader should know what a probability density is, be comfortable with calculus at the level of derivatives and log transforms, and be able to call `optim()` in R.

1.21.3 Theory

For iid data $X_1, \ldots, X_n$ from a density $f(x \mid \theta)$, the **likelihood function** is

$$L(\theta) = \prod_{i=1}^{n} f(X_i \mid \theta),$$

and the **log-likelihood** is $\ell(\theta) = \sum \log f(X_i \mid \theta)$. The **MLE** is

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta} \ell(\theta).$$

When ℓ is smooth, the MLE satisfies the **score equation** $\frac{\partial \ell}{\partial \theta} = 0$. The score is the gradient of the log-likelihood; its expectation under the true parameter is zero.

Under regularity conditions (smoothness, identifiability, interior optimum, finite Fisher information), the MLE is:

- **Consistent**: $\hat{\theta}_n \xrightarrow{P} \theta_0$.
- **Asymptotically normal**: $\sqrt{n}(\hat{\theta}_n - \theta_0) \xrightarrow{d} \mathcal{N}(0, I(\theta_0)^{-1})$.
- **Asymptotically efficient**: its variance attains the Cramer-Rao lower bound.
- **Equivariant**: for any one-to-one transformation g, $g(\hat{\theta}_{\text{MLE}})$ is the MLE of $g(\theta)$.

Standard errors for the MLE come from the inverse of the observed Fisher information, $\mathcal{J}(\hat{\theta}) = -\partial^2 \ell / \partial\theta\partial\theta^\top$, evaluated at $\hat{\theta}$. Confidence intervals can be based on the normal approximation or, better, on the likelihood ratio, which is invariant under reparameterisation.

1.21.4 Assumptions

The asymptotic theory assumes a well-specified model, finite Fisher information, enough smoothness for the Taylor expansion, and a parameter interior to the parameter space. Boundary parameters (variance at zero, proportion at 0 or 1) and non-identifiable models break the standard theory.

1.21.5 R Implementation

```r
set.seed(2026)
x <- rnorm(200, mean = 5, sd = 2)

nll <- function(par) {
  mu    <- par[1]
  sigma <- par[2]
  if (sigma <= 0) return(Inf)
  -sum(dnorm(x, mean = mu, sd = sigma, log = TRUE))
}

fit <- optim(par = c(0, 1), fn = nll, hessian = TRUE,
             method = "L-BFGS-B", lower = c(-Inf, 1e-6))
mle <- fit$par
se  <- sqrt(diag(solve(fit$hessian)))
```

```
data.frame(parameter = c("mu", "sigma"),
           estimate = mle, se = se,
           ci_low = mle - 1.96 * se,
           ci_high = mle + 1.96 * se)
```

The negative log-likelihood is minimised with `optim()`; standard errors come from the inverse Hessian.

1.21.6 Output & Results

```
  parameter estimate    se  ci_low ci_high
1        mu    5.093 0.143   4.812   5.374
2     sigma    2.022 0.101   1.823   2.221
```

The MLE recovers the true parameters ($\mu = 5$, $\sigma = 2$) with uncertainty quantified by the observed-information-based standard errors.

1.21.7 Interpretation

For a manuscript, the MLE is typically reported as "parameters were estimated by maximum likelihood (Fisher-scoring convergence to 1e-8)", with point estimates, standard errors, and either Wald or profile-likelihood confidence intervals. Information criteria (AIC, BIC) are functions of the maximised likelihood and are reported alongside.

1.21.8 Practical Tips

- Always work on the log scale for numerical stability; products of densities underflow quickly.
- Start the optimiser at sensible initial values – method-of-moments estimates are a good default.
- Parameterise so that the parameter space is unbounded (log-variance instead of variance, logit-probability instead of probability); this makes `optim()` easier and avoids boundary issues.
- For models with many parameters, use `nlm()`, `optim(...,method="BFGS")`, or specialised packages (`bbmle`, `maxLik`, or model-specific tools like `glm()`).
- When the MLE is at a boundary, standard asymptotic inference fails; use bootstrap CIs or specialised boundary-aware procedures instead.

1.21.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.21.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.22 The Method of Moments

1.22.1 Introduction

The method of moments (MoM) is the oldest systematic technique for parameter estimation. It replaces theoretical moments of a distribution with their sample counterparts and solves the resulting equations for the parameters. MoM estimators are typically consistent and easy to derive, making them useful both as standalone estimators and as starting values for more refined procedures like maximum likelihood.

1.22.2 Prerequisites

The reader should know what a moment of a distribution is ($E[X^k]$) and should be able to compute simple summary statistics in R.

1.22.3 Theory

For a distribution parameterised by $\theta = (\theta_1, \dots, \theta_p)$, the k-th theoretical moment is $\mu_k(\theta) = E[X^k]$, and the k-th sample moment is $\hat{\mu}_k = \frac{1}{n}\sum X_i^k$. The MoM sets

$$\hat{\mu}_k = \mu_k(\theta), \qquad k = 1, \dots, p,$$

and solves for θ. The method generalises to central moments, fractional moments, and functions of moments; the essential idea is that moments are easy to estimate, so any parameter expressible as a function of moments can be estimated by plug-in.

Examples:

- **Gamma**(α, β) with mean α/β and variance α/β^2:

$$\hat{\beta} = \bar{X}/s^2, \qquad \hat{\alpha} = \bar{X}^2/s^2.$$

- **Normal**(μ, σ^2): first moment gives $\hat{\mu} = \bar{X}$; second central moment gives $\hat{\sigma}^2 = \frac{1}{n}\sum(X_i - \bar{X})^2$.
- **Beta**(α, β) with mean $\alpha/(\alpha+\beta)$ and variance $\alpha\beta/[(\alpha+\beta)^2(\alpha+\beta+1)]$: solve the two moment equations for $\hat{\alpha}, \hat{\beta}$.

MoM estimators are consistent under mild conditions (existence of the moments being used and continuity of the inverse mapping from moments to parameters). Their asymptotic distribution is normal, with variance obtainable via the delta method. They are generally less efficient than the MLE, but often much simpler to compute.

1.22.4 Assumptions

The necessary moments must exist; for heavy-tailed distributions like the Cauchy (no finite mean), MoM fails. The mapping from moments to parameters must be solvable, which requires enough moments to identify the parameters uniquely.

1.22.5 R Implementation

```r
set.seed(2026)

x <- rgamma(500, shape = 4, rate = 0.5)

xbar  <- mean(x)
s2    <- var(x)

beta_hat  <- xbar / s2
alpha_hat <- xbar^2 / s2

c(alpha_hat = alpha_hat, beta_hat = beta_hat,
  true_alpha = 4, true_rate = 0.5)
```

We generate 500 draws from Gamma(4, 0.5), compute the first two sample moments, and back out the MoM estimators.

1.22.6 Output & Results

```
 alpha_hat  beta_hat true_alpha  true_rate
  3.91         0.48       4.00       0.50
```

Both estimators are close to the truth. Across many simulated datasets, the bias of both goes to zero and their variances shrink at rate $1/n$.

1.22.7 Interpretation

MoM estimators are useful when a closed-form expression is desirable for reporting, or when MLE requires numerical optimisation that might fail for specific initial values. In many textbook distributions, the MoM and the MLE coincide; in others, MoM is a reasonable starting value for the MLE search.

1.22.8 Practical Tips

- Check that the moments implied by your MoM estimator are finite for all parameter values; otherwise the estimator is undefined.
- For distributions where MoM and MLE differ, MLE is usually preferred for efficiency; MoM is fine for preliminary or exploratory work.
- Generalised method of moments (GMM) extends MoM to systems with more equations than parameters, weighting them optimally; it is the standard estimator in econometrics for instrumental-variable and panel models.
- When the mapping from moments to parameters is not invertible in closed form, use `uniroot()` or `optim()` to solve the equations numerically.
- Report the moments used and the mapping; readers can then verify the calculation and check the code.

1.22.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.22.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.23 Moments and Moment Generating Functions

1.23.1 Introduction

Moments are the descending sequence of expectations $E[X], E[X^2], E[X^3], ...$ that summarise a distribution's location, spread, skewness, kurtosis, and finer shape features. The moment generating function (MGF) packages all moments into a single function; when it exists, the MGF characterises the distribution uniquely. MGFs give elegant proofs of the CLT, the distribution of sums of independent variables, and the closure properties of many distributional families.

1.23.2 Prerequisites

The reader should know what an expectation is and be comfortable with basic calculus.

1.23.3 Theory

The k**-th moment** of X is $\mu_k = E[X^k]$ (provided the integral converges). The **central moments** are $\mu_k^{(c)} = E[(X-\mu)^k]$; $\mu_2^{(c)}$ is the variance, $\mu_3^{(c)}$ feeds skewness, $\mu_4^{(c)}$ feeds kurtosis.

The **moment generating function** is

$$M_X(t) = E[e^{tX}], \qquad \text{for all } t \text{ where the expectation is finite.}$$

If M_X is finite in a neighbourhood of zero, then (i) all moments of X exist, (ii) the k-th moment equals the k-th derivative of M_X at zero, $M_X^{(k)}(0) = \mu_k$, and (iii) M_X determines the distribution uniquely.

Properties.

- For independent X, Y: $M_{X+Y}(t) = M_X(t)M_Y(t)$. MGFs turn sums into products.
- For affine transformations: $M_{aX+b}(t) = e^{bt}M_X(at)$.
- The MGF of a standard normal is $M(t) = e^{t^2/2}$. For the normal $\mathcal{N}(\mu, \sigma^2)$: $M(t) = e^{\mu t + \sigma^2 t^2/2}$.

Limitations. The MGF may not exist for all t – Cauchy's MGF is infinite everywhere except at $t = 0$. For such distributions, the **characteristic function** $\varphi_X(t) = E[e^{itX}]$ is preferred: it always exists and shares most useful properties with the MGF.

MGFs prove the CLT elegantly: the MGF of the standardised mean converges to $e^{t^2/2}$, which is the MGF of the standard normal, and uniqueness of MGFs gives convergence in distribution.

1.23.4 Assumptions

Existence of moments up to order k requires $E[|X|^k] < \infty$. Existence of the MGF in a neighbourhood of zero is a much stronger condition (light-tailed distributions only).

1.23.5 R Implementation

```
set.seed(2026)
n <- 1e6
```

```r
# Moments of a sample from a gamma distribution
x <- rgamma(n, shape = 3, rate = 2)
moments <- sapply(1:4, function(k) mean(x^k))
names(moments) <- paste0("m_", 1:4)
moments

# Theoretical moments: E[X^k] = Gamma(shape + k)/Gamma(shape) * rate^-k
shape <- 3; rate <- 2
theoretical <- sapply(1:4, function(k) {
  gamma(shape + k) / gamma(shape) * rate^(-k)
})
names(theoretical) <- paste0("m_", 1:4)
theoretical

# MGF of gamma at a few t: M(t) = (1 - t/rate)^-shape for t < rate
t_vals <- c(0.1, 0.5, 1.0)
mgf_theoretical <- (1 - t_vals / rate)^(-shape)
mgf_empirical   <- sapply(t_vals, function(t) mean(exp(t * x)))
cbind(t = t_vals, theoretical = mgf_theoretical,
      empirical = mgf_empirical)
```

We compare empirical and theoretical moments and MGF values for a gamma distribution.

1.23.6 Output & Results

```
      m_1       m_2       m_3       m_4
   1.5001    3.0018    8.2566   32.9916

      m_1       m_2       m_3       m_4
   1.5000    3.0000    8.2500   33.0000

     t  theoretical  empirical
[1,] 0.1   1.2167    1.2168
[2,] 0.5   2.3704    2.3707
[3,] 1.0   8.0000    8.0076
```

Empirical and theoretical values agree to three decimal places with one million observations; with smaller samples the agreement is poorer in the higher moments and far tails.

1.23.7 Interpretation

For applied statistics, MGFs appear most often as proof tools rather than quantities to compute. Many closure properties (the normal family is closed under convolution; the gamma family has the same closure with a fixed rate parameter) are memorised via their MGFs.

1.23.8 Practical Tips

- For heavy-tailed distributions, higher moments may not exist; always check that the moments you are computing are finite.
- Characteristic functions always exist; use them when MGFs don't.
- In applied work, the first four moments (mean, variance, skewness, kurtosis) typically suffice; higher moments are noisy and rarely informative.
- The MGF of a sum of independent variables is the product of individual MGFs – a fact that makes many distributional derivations tractable.
- Do not confuse moments with cumulants: the cumulant generating function (log of the MGF) has additive behaviour under sums and is often preferred in theoretical work.

1.23.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.23.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.24 Order Statistics

1.24.1 Introduction

Given a sample $X_1, \ldots, X_n$, the **order statistics** are the sorted values $X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(n)}$. They are the raw material for the median, the quantiles, the range, the IQR, and many non-parametric and robust methods. Their distributional theory provides exact confidence intervals for quantiles and limiting distributions for extremes – tools that analytical approaches to the mean cannot reach.

1.24.2 Prerequisites

The reader should know what a CDF and a PDF are, and should be comfortable with `sort()` and `quantile()` in R.

1.24.3 Theory

For iid X_i with CDF F, the CDF of the k-th order statistic is

$$F_{X_{(k)}}(x) = \sum_{j=k}^{n} \binom{n}{j} F(x)^j [1 - F(x)]^{n-j}.$$

If F has density f, the density of $X_{(k)}$ is

$$f_{X_{(k)}}(x) = \frac{n!}{(k-1)!(n-k)!} F(x)^{k-1} [1 - F(x)]^{n-k} f(x).$$

The joint density of $X_{(i)}$ and $X_{(j)}$ for $i < j$ has a similar combinatorial form.

Uniform case. If $U_i \sim \text{Uniform}(0, 1)$, then $U_{(k)} \sim \text{Beta}(k, n - k + 1)$ with mean $k/(n + 1)$. This is why plotting positions in Q-Q plots use $(k - 0.5)/n$ or $k/(n + 1)$ rather than k/n.

Sample quantiles. For sample size n, the sample p-quantile is usually defined as $X_{(\lceil np \rceil)}$ or an interpolation between adjacent order statistics; R's `quantile()` offers nine definitions via the `type` argument. The sample median is $X_{((n+1)/2)}$ for odd n.

Extremes. The distributions of $X_{(1)}$ and $X_{(n)}$ are of independent interest. For large n, appropriately normalised extremes have limiting distributions (Gumbel, Frechet, or Weibull) given by extreme-value theory.

Confidence intervals for quantiles. Because of the combinatorial form of the CDF of an order statistic, exact distribution-free CIs for a population quantile can be constructed: the interval $[X_{(i)}, X_{(j)}]$ has a known coverage probability for the p-quantile, regardless of F.

1.24.4 Assumptions

The classical theory assumes iid observations; most of the formulas extend directly to exchangeable data. For dependent data, the distribution of order statistics is much more complicated.

1.24.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
n <- 20
p <- 0.75
reps <- 10000
```

```r
sim_q <- replicate(reps, {
  x <- rnorm(n, mean = 50, sd = 10)
  sort(x)[ceiling(p * (n + 1))]
})

true_q <- qnorm(p, mean = 50, sd = 10)
c(empirical_mean = mean(sim_q),
  true           = true_q,
  empirical_SE   = sd(sim_q))

x <- rnorm(n, mean = 50, sd = 10)
sorted_x <- sort(x)
q05 <- binom.test(x = round(p * n), n = n, p = p)
k_lo <- qbinom(0.025, n, p)
k_hi <- qbinom(0.975, n, p) + 1
c(lower = sorted_x[k_lo], upper = sorted_x[min(k_hi, n)])
```

The first block simulates the sampling distribution of the 75th-percentile estimate. The second constructs a distribution-free CI for the population 75th percentile from binomial probabilities applied to ranks.

1.24.6 Output & Results

```
empirical_mean           true   empirical_SE
         56.13          56.74           3.01

lower  upper
54.17  72.81
```

The sample quantile is slightly biased in small samples (a known feature for definitions like `type=7`); the distribution-free CI for the true quantile is wide but valid for any continuous distribution.

1.24.7 Interpretation

Order-statistic-based CIs are useful when the distribution of the data is unknown or clearly non-normal. Reporting "median survival 27 months (distribution-free 95% CI 22 to 31)" is an appropriate summary in survival analysis when the parametric form is in doubt.

1.24.8 Practical Tips

- R's `quantile()` default is `type=7`; specify it explicitly if exact reproducibility across software is important.
- For extreme-value inference (maxima, minima), do not rely on asymptotic normality; use extreme-value theory and the `evd` or `extRemes` packages.

- Distribution-free CIs for quantiles are wide; they are the price of making no distributional assumption.
- When the sample is small ($n < 20$), the set of achievable confidence levels for a distribution-free CI is discrete – you cannot always get exactly 95%.
- Empirical CDFs, constructed from order statistics, are the basis of Kolmogorov-Smirnov tests and many other non-parametric procedures.

1.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.24.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.25 Pivotal Quantities

1.25.1 Introduction

A pivotal quantity – or pivot – is a function of the data and the parameter whose probability distribution does not depend on the parameter. The existence of a pivot is what makes exact confidence intervals possible: if $Q(X, \theta)$ has a known distribution, we can rearrange the probability statement about Q into a probability statement about θ, and that rearrangement is the CI.

1.25.2 Prerequisites

The reader should understand the concepts of confidence intervals, the t-distribution, and the chi-squared distribution.

1.25.3 Theory

A **pivotal quantity** $Q = Q(X_1, \ldots, X_n; \theta)$ satisfies: for every θ, the distribution of Q is the same known distribution. This is stronger than "a test statistic under the null": a pivot's distribution does not depend on θ *for every* value of θ.

Canonical examples (iid data):

- **Normal, known** σ^2: $Z = \sqrt{n}(\bar{X} - \mu)/\sigma \sim \mathcal{N}(0, 1)$.

- **Normal, unknown σ^2**: $T = \sqrt{n}(\bar{X} - \mu)/s \sim t_{n-1}$.
- **Normal variance**: $W = (n-1)s^2/\sigma^2 \sim \chi^2_{n-1}$.
- **Ratio of normal variances**: $F = s_1^2/\sigma_1^2 \cdot \sigma_2^2/s_2^2 \sim F_{n_1-1,n_2-1}$.
- **Uniform**$(0,\theta)$: $\max X_i/\theta \sim \text{Beta}(n,1)$.

Given a pivot Q and its known distribution, pick quantiles $q_{\alpha/2}$ and $q_{1-\alpha/2}$ such that $P(q_{\alpha/2} \leq Q \leq q_{1-\alpha/2}) = 1-\alpha$. Invert the inequalities to isolate θ; the resulting set is a $1-\alpha$ CI with exact coverage.

When no exact pivot exists, **asymptotic pivots** (e.g., $(\hat{\theta}-\theta)/\widehat{\text{SE}}$ which is asymptotically $\mathcal{N}(0,1)$) give CIs with approximate coverage.

1.25.4 Assumptions

Exact pivots depend on distributional assumptions (normality, uniformity). When those fail, the nominal coverage is not guaranteed; bootstrap methods or robust asymptotic pivots are alternatives.

1.25.5 R Implementation

```r
set.seed(2026)

n <- 25
mu <- 5
sigma <- 2

x <- rnorm(n, mu, sigma)

xbar <- mean(x)
s    <- sd(x)
t_crit <- qt(0.975, df = n - 1)

ci_mean <- xbar + c(-1, 1) * t_crit * s / sqrt(n)
ci_mean

chi_l <- qchisq(0.025, df = n - 1)
chi_u <- qchisq(0.975, df = n - 1)
ci_var <- c((n - 1) * s^2 / chi_u, (n - 1) * s^2 / chi_l)
ci_var
```

Two exact CIs are constructed: the t-pivot for μ, and the chi-squared pivot for σ^2.

1.25.6 Output & Results

```
[1] 4.272  6.044     # 95% CI for mu
[1] 2.197  7.115     # 95% CI for sigma^2 (true value 4)
```

Both intervals are built from pivots with exact distributions under the normal model; their coverage is exactly 95% across repeated sampling.

1.25.7 Interpretation

Pivotal quantities are often implicit in textbook CIs. Reporting a t-interval for a mean or a chi-squared interval for a variance is reporting a pivot-based construction. The pivot formalism explains why these CIs have exact coverage under the assumed model – and why they lose coverage when the model is wrong.

1.25.8 Practical Tips

- When an exact pivot exists, prefer it over an asymptotic normal approximation, especially for small n.
- Pivots are rare outside the normal family; for most models, asymptotic pivots or the bootstrap are the practical tools.
- The CI for the variance derived from the chi-squared pivot is asymmetric and not centred at s^2; do not report it as "$s^2 \pm$ margin".
- Pivotal CIs for transformations (e.g., log-odds) can be constructed on the transformed scale and back-transformed; this preserves exact coverage if the pivot is exact.
- Not every test statistic is a pivot – the Wald statistic is not, in general, exactly pivotal, which is why its CIs are approximate.

1.25.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.25.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.26 Population vs. Sample

1.26.1 Introduction

Everything in applied statistics rests on a distinction that sounds obvious until one tries to pin it down: the difference between the population a claim is about

and the sample from which the claim is drawn. A *parameter* summarises the population; a *statistic* summarises the sample. The goal of inference is to use the statistic to say something defensible about the parameter. Confusion between the two is the single most common source of overreach in scientific reporting.

1.26.2 Prerequisites

The reader should know what an average is and should be comfortable with simple R arithmetic. No previous inferential statistics is assumed.

1.26.3 Theory

The **target population** is the set of units about which we want to make a claim: "all adults in Germany", "all HER2-positive breast cancer patients treated at this hospital in 2020", "all spikes recorded from V1 neurons in macaque M during task T". It is defined *first*, in scientific terms, not by what is easy to measure.

The **sampling frame** is the operational list from which units are actually drawn: the register of registered voters; the hospital's electronic health record; the 48 neurons that were successfully isolated. When the sampling frame differs from the target population, the sample may be representative of the frame but not of the intended population – *coverage bias*.

The **sample** is the subset of the frame actually observed. Its summary statistics (mean, proportion, correlation) are estimates of the corresponding population parameters. Inference is the machinery that quantifies how close the statistic is likely to be to the parameter, given the sampling scheme and the observed variability.

By convention, Greek letters denote population parameters and Latin letters denote sample statistics:

- Population mean μ, sample mean $\bar{x}$.
- Population variance σ^2, sample variance s^2.
- Population proportion π, sample proportion $\hat{p}$.
- Population regression coefficient β, estimate $\hat{\beta}$.

1.26.4 Assumptions

Inference about the population from the sample requires that the sampling scheme be *representative* – either by probability sampling (each unit in the frame has a known, positive probability of selection) or by an identified mechanism under which conclusions transfer (e.g., random assignment within a convenience sample, which supports causal but not prevalence claims).

1.26.5 R Implementation

```
library(ggplot2)

set.seed(2026)
population <- rnorm(1e6, mean = 75, sd = 10)
mu    <- mean(population)
sigma <- sd(population)

sample_n  <- 50
one_sample <- sample(population, sample_n)
xbar <- mean(one_sample)
s    <- sd(one_sample)

c(population_mean = mu, sample_mean = xbar,
  population_sd   = sigma, sample_sd   = s)

sampling_distribution <- replicate(5000, mean(sample(population,
 ↪  sample_n)))
quantile(sampling_distribution, c(0.025, 0.975))
```

We treat `population` as a fictional census of one million values. A single sample of $n = 50$ gives a point estimate $\bar{x}$; repeated sampling traces out the sampling distribution of $\bar{x}$.

1.26.6 Output & Results

```
 population_mean      sample_mean  population_sd       sample_sd
       74.99806         76.51094       10.00139         9.47318

     2.5%       97.5%
  73.23143    76.78120
```

The sample mean is close to but not equal to the population mean; the sampling distribution tells us that over repeated sampling, 95% of sample means fall within about ± 1.4 of the truth.

1.26.7 Interpretation

When a paper reports "mean systolic blood pressure in the cohort was 128 mmHg (SD 14)", the reader should understand that this is a *statistic*, and that the population parameter it estimates has some uncertainty, quantified by the standard error or confidence interval. The methods section should state explicitly what population is targeted and through what frame.

1.26.8 Practical Tips

- Define the target population in plain scientific terms before the data arrive. Retrospective definitions invite bias.
- Document the sampling frame, inclusion/exclusion criteria, and selection probabilities. This is what makes a study reproducible.
- If the frame deviates from the target population (convenience sampling in clinical trials, online panels, self-selected respondents), state the deviation and the likely direction of bias.
- Sample statistics are random; reporting only a point estimate without an SE or CI gives a false sense of precision.
- Never confuse "the distribution of the data" with "the sampling distribution of a statistic". The first describes individuals, the second describes averages (or other summaries) across repeated samples.

1.26.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.26.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.27 Sampling Distributions

1.27.1 Introduction

The sampling distribution of a statistic is the distribution of its values across hypothetical repetitions of the sampling process. Every confidence interval, every p-value, and every standard error is a statement about the sampling distribution of some statistic. Understanding this concept makes inference intuitive; ignoring it makes it a ritual of incantations.

1.27.2 Prerequisites

The reader should know what a sample mean is and should be comfortable with histograms and R's `replicate()` function.

1.27.3 Theory

Let $T = T(X_1, \ldots, X_n)$ be a statistic computed from a sample of size n drawn from a population with some distribution. The **sampling distribution** of T is the distribution of values T would take if the sampling process were repeated many times, each time drawing a fresh sample of size n from the same population.

Key facts:

- The sampling distribution is a property of T and of the sampling design, not of any particular observed sample.
- Its mean is the expectation $E[T]$. For unbiased estimators, $E[T]$ equals the population parameter.
- Its standard deviation is called the **standard error** of T: $\mathrm{SE}(T) = \sqrt{\mathrm{Var}(T)}$.
- Its shape depends on the statistic, on n, and on the population distribution. For the sample mean from a population with finite variance, the central limit theorem guarantees approximate normality as n grows.

For the sample mean $\bar{X}$ of n iid observations with mean μ and variance σ^2:

$$E[\bar{X}] = \mu, \qquad \mathrm{SE}(\bar{X}) = \sigma/\sqrt{n}.$$

The standard error tells us the typical distance between $\bar{X}$ and μ – it is the unit in which confidence intervals are measured.

1.27.4 Assumptions

The theoretical sampling distribution derived in most textbooks assumes iid sampling from a population with finite variance. Deviations (heavy tails, dependence, non-iid sampling) change the shape or the formulas; the bootstrap lets us approximate the sampling distribution empirically without these assumptions.

1.27.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
population <- rgamma(1e5, shape = 2, rate = 0.1)
mu <- mean(population)
sigma <- sd(population)

draw_mean <- function(n) mean(sample(population, n))

reps <- 10000
```

```r
means_n10  <- replicate(reps, draw_mean(10))
means_n100 <- replicate(reps, draw_mean(100))

cat(sprintf("n=10  : mean=%.2f, SE=%.2f; theoretical SE=%.2f\n",
            mean(means_n10), sd(means_n10), sigma / sqrt(10)))
cat(sprintf("n=100 : mean=%.2f, SE=%.2f; theoretical SE=%.2f\n",
            mean(means_n100), sd(means_n100), sigma / sqrt(100)))

df <- data.frame(xbar = c(means_n10, means_n100),
                 n = factor(rep(c(10, 100), each = reps)))
ggplot(df, aes(x = xbar, fill = n)) +
  geom_histogram(alpha = 0.6, bins = 50, position = "identity") +
  labs(x = expression(bar(X)), y = "Frequency") +
  theme_minimal()
```

For a skewed gamma population, the sampling distribution of the mean concentrates as n grows, and its spread matches the theoretical standard error.

1.27.6 Output & Results

```
n=10  : mean=19.98, SE=4.42; theoretical SE=4.47
n=100 : mean=20.01, SE=1.41; theoretical SE=1.41
```

Increasing n by a factor of 10 decreases the standard error by $\sqrt{10} \approx 3.16$, exactly as the $1/\sqrt{n}$ rule predicts.

1.27.7 Interpretation

The standard deviation of the sampling distribution – the SE – is what a confidence interval is built from. A 95% CI for the mean is approximately $\bar{x} \pm 1.96 \cdot \text{SE}(\bar{X})$. When a manuscript reports "mean 19.98 (SE 1.41, 95% CI 17.22–22.74)", every number in that summary is derived from the sampling distribution of the statistic.

1.27.8 Practical Tips

- Do not confuse the standard deviation of the data with the standard error of a statistic. The SD describes variability across individuals; the SE describes variability across hypothetical samples of size n.
- When the theoretical sampling distribution is unknown, approximate it by the bootstrap: resample with replacement from the observed sample, compute the statistic, and use the empirical distribution of the resampled statistics as an estimate.
- The sampling distribution depends on n – larger samples yield tighter distributions, as the $1/\sqrt{n}$ rule captures for the mean.

- For statistics other than the mean, the sampling distribution may be harder to derive analytically but is equally well-defined; simulation or bootstrap is the general-purpose tool.
- The t distribution that appears in small-sample inference is the sampling distribution of $(\bar{X} - \mu)/(s/\sqrt{n})$ under normality with unknown variance, not of the mean itself.

1.27.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.27.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.28 Sampling Methods

1.28.1 Introduction

How a sample is drawn matters as much as how big it is. The same 500 observations can support valid inference about a population or produce a misleading summary, depending entirely on the sampling mechanism. Classical sampling theory names a handful of probability designs, each with characteristic strengths, variance properties, and analysis requirements.

1.28.2 Prerequisites

The reader should know the difference between a population and a sample, and should be familiar with random-number generation in R.

1.28.3 Theory

Simple random sampling (SRS) draws n units from a frame of N such that every subset of size n is equally likely. The sample mean is unbiased for the population mean, with standard error $\sigma/\sqrt{n} \cdot \sqrt{1 - n/N}$ (the finite-population correction is negligible for $n \ll N$). SRS is the theoretical baseline against which other designs are compared.

Stratified sampling partitions the population into non-overlapping strata – e.g., by region, age band, or hospital – and draws an SRS from each. The pooled estimator has *lower* variance than SRS whenever the strata are more homogeneous within than the population at large, which is typical. Stratification also guarantees representation of small subgroups that might be missed by chance under SRS.

Cluster sampling draws whole clusters (schools, villages, GP practices) and then includes some or all units within each. Variance is *higher* than SRS for the same n because units within a cluster are correlated; the loss is quantified by the design effect $\text{deff} = 1 + (\bar{m} - 1)\rho$, where $\bar{m}$ is average cluster size and ρ is the intra-cluster correlation. Cluster sampling trades statistical efficiency for dramatically lower data-collection cost.

Systematic sampling picks every k-th unit from a list after a random start. It is equivalent to SRS when the list is in random order; when the list has periodicity matching k, it can give badly biased estimates.

Convenience sampling recruits whatever units are available – internet panels, patients at one hospital, undergraduates in a course. It supports no probability-based inference about a broader population; any generalisation must be argued on non-statistical grounds (mechanism, theory, replication).

1.28.4 Assumptions

Probability designs assume a complete sampling frame and known selection probabilities. Analysis must incorporate design weights ($w_i = 1/\pi_i$) and the design variance; standard software that ignores weighting (a t-test on weighted survey data, say) produces incorrect standard errors.

1.28.5 R Implementation

```
library(sampling)

set.seed(2026)
N <- 10000
pop <- data.frame(
  id     = 1:N,
  stratum = sample(c("A", "B", "C"), N, replace = TRUE, prob = c(0.5,
   ↪  0.3, 0.2)),
  y      = rnorm(N, mean = c(A = 50, B = 60, C = 70)[sample(c("A", "B",
   ↪  "C"), N,
           replace = TRUE, prob = c(0.5, 0.3, 0.2))], sd = 5)
)

srs_idx <- sample(N, 200)
srs <- pop[srs_idx, ]
```

```r
mean(srs$y); sd(srs$y) / sqrt(200)

str_idx <- strata(pop, stratanames = "stratum",
                  size = c(A = 100, B = 60, C = 40),
                  method = "srswor")
strat <- pop[str_idx$ID_unit, ]
tapply(strat$y, strat$stratum, mean)
weighted.mean(tapply(strat$y, strat$stratum, mean),
              w = table(pop$stratum))
```

The code draws a 200-unit SRS and a 200-unit stratified sample from the same population, then computes the design-consistent stratified estimator via weighted averages.

1.28.6 Output & Results

```
[1] 57.41       # SRS mean
[1] 0.533       # SE of SRS mean

     A      B      C
 49.97  59.86  70.22   # stratum means

[1] 56.94       # stratified mean (weighted by stratum population
↪  proportions)
```

Both estimators target the same quantity; the stratified design has systematically smaller variance across repeated sampling.

1.28.7 Interpretation

A methods section should state the sampling design precisely: "Patients were sampled by stratified random sampling, stratified by treatment centre with proportional allocation, from a frame of 4 832 eligible records identified by the inclusion criteria." This lets a reviewer evaluate whether analysis choices respect the design.

1.28.8 Practical Tips

- When using stratified or cluster designs, analyse with the `survey` package (or equivalent) so standard errors incorporate weights and clustering.
- Proportional allocation is a sensible default for stratified sampling; optimal (Neyman) allocation weights strata by $N_h \sigma_h$, which can be used when prior variance estimates are available.
- For cluster designs, report the design effect and the ICC; reviewers will ask.

- Systematic sampling is fine for logistical convenience, but inspect the sampling list for periodicity before using it.
- Convenience samples are not disqualified from publication, but their external validity must be argued explicitly; treating them as if they were random is the root cause of many replication failures.

1.28.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.28.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.29 Scales of Measurement

1.29.1 Introduction

Every variable in a dataset carries a scale of measurement, and the scale dictates which statistics are meaningful to compute. Stevens' 1946 taxonomy – nominal, ordinal, interval, ratio – remains the default vocabulary for this idea, and even when its strict version is criticised, the underlying distinctions are unavoidable in applied work. Computing a mean of jersey numbers is nonsense because jersey numbers are nominal; computing a median of exam grades is sensible because grades are at least ordinal.

1.29.2 Prerequisites

A reader should know what a variable is and should be able to tell numeric data from categorical data in R (as `numeric` vs. `factor` or `character`).

1.29.3 Theory

- **Nominal**: labels without order. Operations: equality. Examples: sex, blood type, country. Meaningful summaries: mode, frequency counts.
- **Ordinal**: ordered labels without meaningful distances. Operations: $=$, $<$, $>$. Examples: Likert scale, disease stage (I–IV), pain rating. Meaningful summaries: median, quantiles, rank correlations.

- **Interval**: ordered, with meaningful distances but no true zero. Operations: $+$, $-$. Examples: temperature in Celsius, calendar dates. Ratios are *not* meaningful: 40 °C is not "twice as hot" as 20 °C.
- **Ratio**: interval with a true zero. Operations: $\times$, $/$. Examples: mass, concentration, time elapsed, counts. Ratios are meaningful: 40 kg is twice 20 kg.

The scale determines permissible summary statistics and inferential procedures. A Pearson correlation requires interval or ratio data; a Spearman correlation extends to ordinal. A t-test operates on (at minimum) interval data, though in practice it is widely applied to Likert-like composites with reasonable robustness.

1.29.4 Assumptions

The taxonomy itself is descriptive rather than inferential; its use rests on the assumption that the chosen scale accurately characterises the variable. In practice, many variables sit ambiguously – is a 5-point Likert item ordinal or interval? Standard practice treats Likert *composite scales* (averages of many items) as interval and *single items* as ordinal.

1.29.5 R Implementation

```
library(ggplot2)

df <- data.frame(
  id      = 1:8,
  country = factor(c("DE", "US", "FR", "DE", "US", "FR", "DE", "US")),
  stage   = factor(c("I", "II", "III", "II", "IV", "I", "III", "II"),
                   levels = c("I", "II", "III", "IV"), ordered = TRUE),
  temp_c  = c(36.5, 37.1, 38.2, 36.9, 37.4, 36.7, 38.0, 37.2),
  mass_kg = c(72.1, 81.3, 65.8, 74.2, 88.7, 61.4, 79.0, 84.5)
)

table(df$country)
median(as.numeric(df$stage))
mean(df$temp_c)
mean(df$mass_kg); df$mass_kg[1] / df$mass_kg[3]
```

Nominal `country` is summarised by frequencies; ordinal `stage` by the median (the raw factor must be coerced to numeric for the median call); interval `temp_c` by the mean; ratio `mass_kg` by the mean and by ratios.

1.29.6 Output & Results

```
DE FR US
 3  2  3
```

```
[1] 2      # median stage (II)
[1] 37.25  # mean temperature in C
[1] 75.875 # mean mass in kg
[1] 1.096  # mass ratio is meaningful for ratio data
```

1.29.7 Interpretation

A paper's methods section should list each variable and its scale, because the scale justifies the summary statistic shown in the descriptive-statistics table. "Disease stage (ordinal) is summarised by the median (IQR); body mass (ratio) by the mean (SD)." This prevents a reader from being surprised by, say, a mean of a four-level ordinal variable dressed up as a continuous measurement.

1.29.8 Practical Tips

- Encode ordinal variables in R as `factor(...,ordered=TRUE)` so that methods that respect ordering (proportional-odds models, Spearman correlation) pick up the order automatically.
- Resist the urge to compute means of ordinal variables for reporting; a mean of disease stages is uninterpretable even when it produces a number.
- Do not confuse a *count* (ratio) with a *code* (nominal): "category = 3" is nominal even though stored as an integer.
- Be careful with variables that look ratio but are not: pH is actually a logarithmic transformation of concentration, so differences on the pH scale correspond to ratios of hydrogen-ion concentration.
- When a variable's scale is ambiguous (Likert items), pre-register your analytical choice – ordinal or interval – so reviewers cannot object after the fact.

1.29.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.29.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.30 Slutsky's Theorem

1.30.1 Introduction

Slutsky's theorem is one of the workhorse results of asymptotic statistics. It says that we can substitute a consistent estimator for a constant in an asymptotic statement, without changing the limiting distribution. This simple-sounding result is what makes the Wald statistic have an asymptotic standard-normal distribution, what justifies using the sample SD in place of the population SD in a t-statistic's denominator, and what underlies countless routine asymptotic arguments in applied work.

1.30.2 Prerequisites

The reader should understand convergence in probability and convergence in distribution.

1.30.3 Theory

Slutsky's theorem. Suppose $Y_n \xrightarrow{d} Y$ and $Z_n \xrightarrow{P} c$, where c is a constant. Then:

- $Y_n + Z_n \xrightarrow{d} Y + c$,
- $Y_n Z_n \xrightarrow{d} cY$,
- $Y_n / Z_n \xrightarrow{d} Y/c$ (provided $c \neq 0$).

The key requirement is that Z_n converges to a *constant*, not to another random variable. Convergence of Z_n to a non-degenerate random variable does not generally justify these manipulations.

Typical use. Consider the Wald statistic for a mean:

$$W_n = \frac{\sqrt{n}(\bar{X}_n - \mu)}{s_n}.$$

We know $\sqrt{n}(\bar{X}_n - \mu)/\sigma \xrightarrow{d} \mathcal{N}(0,1)$ by the CLT, and $s_n \xrightarrow{P} \sigma$ by the LLN applied to the sample variance. Writing

$$W_n = \frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma} \cdot \frac{\sigma}{s_n},$$

Slutsky's theorem gives $W_n \xrightarrow{d} \mathcal{N}(0,1) \cdot 1 = \mathcal{N}(0,1)$.

This is why the Wald statistic with an estimated SE still has a standard-normal limit. The same principle applies to Wald chi-squared statistics, score statistics, and almost every test statistic in asymptotic theory.

1.30.4 Assumptions

Slutsky's theorem requires (i) Z_n converges in probability to a constant, not to a random variable, and (ii) the combined operation (sum, product, quotient) is well-defined. For quotients the constant must not be zero; dividing by a small random denominator can still introduce tail issues at finite n that Slutsky does not warn about.

1.30.5 R Implementation

```r
set.seed(2026)

n <- 500
mu <- 3; sigma <- 2
reps <- 5000

wald_stats <- replicate(reps, {
  x <- rnorm(n, mu, sigma)
  sqrt(n) * (mean(x) - mu) / sd(x)
})

qqnorm(wald_stats, pch = ".")
qqline(wald_stats, col = "red")

c(mean = mean(wald_stats),
  sd   = sd(wald_stats))
```

We simulate the Wald statistic across 5000 replications. Slutsky's theorem predicts the limiting distribution is standard normal; the simulation confirms empirically.

1.30.6 Output & Results

```
    mean       sd
 0.00411   0.9997
```

The empirical mean is near zero and SD near one, confirming standard-normal behaviour. The Q-Q plot falls on the reference line.

1.30.7 Interpretation

Slutsky's theorem is rarely cited by name in applied papers, but its conclusions appear constantly. The phrase "by the CLT, and using the consistency of $\hat{\sigma}$ for σ, the Wald statistic is asymptotically standard normal" is a Slutsky argument compressed into a single sentence.

1.30.8 Practical Tips

- When you see an asymptotic argument that swaps an estimator for a parameter, that is (usually) Slutsky at work.
- Slutsky's theorem does not help when both sequences are non-degenerate; for that, joint convergence and the continuous mapping theorem are needed.
- In finite samples, the Slutsky approximation may be crude. For small n, the t-distribution (which accounts for the estimated denominator more precisely) is preferred over the Wald normal.
- Extensions hold for continuous functions: if $Y_n \xrightarrow{d} Y$ and $Z_n \xrightarrow{P} c$, then $g(Y_n, Z_n) \xrightarrow{d} g(Y, c)$ for continuous g at (y, c).
- The theorem is sometimes stated with Z_n taking values in any fixed dimensional space – useful when the "constant" is a vector or matrix (e.g., an estimated covariance).

1.30.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.30.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.31 The Standard Error

1.31.1 Introduction

The **standard error** of a statistic is the standard deviation of its sampling distribution. It quantifies how much the statistic would vary across hypothetical repetitions of the sampling process. Every confidence interval and every t- or z-statistic is built on an estimate of the standard error, so understanding and correctly computing SEs is central to competent data analysis.

1.31.2 Prerequisites

A reader should know what the sampling distribution is, what a sample standard deviation is, and how to use the `sd()` and `mean()` functions in R.

1.31.3 Theory

For an iid sample $X_1, \dots, X_n$ from a population with variance σ^2, the sample mean $\bar{X}$ has variance σ^2/n and therefore

$$\mathrm{SE}(\bar{X}) = \sigma/\sqrt{n}.$$

Because σ is usually unknown, it is estimated by the sample SD s, giving the plug-in estimator $\widehat{\mathrm{SE}}(\bar{X}) = s/\sqrt{n}$.

For a binary proportion $\hat{p} = k/n$, the standard error under binomial sampling is

$$\mathrm{SE}(\hat{p}) = \sqrt{p(1-p)/n}, \qquad \widehat{\mathrm{SE}}(\hat{p}) = \sqrt{\hat{p}(1-\hat{p})/n}.$$

For a difference in means across two independent samples with variances s_1^2, s_2^2,

$$\widehat{\mathrm{SE}}(\bar{X}_1 - \bar{X}_2) = \sqrt{s_1^2/n_1 + s_2^2/n_2}.$$

For regression coefficients, standard errors come from the diagonal of $\hat{\sigma}^2(X^\top X)^{-1}$; R returns them in `summary(fit)$coefficients`.

For statistics without a closed-form SE (ratios, quantiles, custom summaries), the **bootstrap** gives a general-purpose estimator: resample with replacement B times, compute the statistic for each resample, and use the SD of the B bootstrap statistics as $\widehat{\mathrm{SE}}$.

1.31.4 Assumptions

The formula-based SEs above assume iid sampling and, for the finite-sample versions, enough regularity for the CLT to justify using them in normal-approximation CIs. The bootstrap SE is consistent under weaker conditions but inherits the design: it is accurate only when resampling units are the same as the independent sampling units (so bootstrap at the cluster level for cluster samples).

1.31.5 R Implementation

```r
library(boot)

set.seed(2026)
x <- rnorm(60, mean = 75, sd = 12)

se_mean     <- sd(x) / sqrt(length(x))
se_mean
```

```r
y <- c(rep(1, 24), rep(0, 76))
p_hat <- mean(y)
se_prop <- sqrt(p_hat * (1 - p_hat) / length(y))
se_prop

iqr_stat <- function(d, i) IQR(d[i])
b <- boot(x, statistic = iqr_stat, R = 5000)
se_iqr <- sd(b$t)
se_iqr
```

The first two blocks use the closed-form SEs for the mean and a proportion; the third uses the bootstrap for the IQR, which has no convenient closed form.

1.31.6 Output & Results

```
[1] 1.552        # SE(mean)
[1] 0.0427       # SE(proportion)
[1] 2.11         # bootstrap SE(IQR)
```

Each of these is the estimated typical distance between the sample statistic and the population parameter it estimates.

1.31.7 Interpretation

The standard error is what gets reported alongside the estimate: “mean 75.8 (SE 1.6)” or “proportion 0.24 (95% CI 0.16–0.32)”, where the CI is approximately the estimate plus/minus 1.96 times the SE. A reviewer glancing at the SE can judge whether the study is precise enough to support its claims.

1.31.8 Practical Tips

- Always distinguish the SD (spread of individual values) from the SE (spread of a statistic); papers regularly report the wrong one.
- For proportions near 0 or 1, the normal-approximation SE is poor; prefer Wilson or Clopper-Pearson intervals, which are available via `binom.test()` and the `binom` package.
- For differences between groups, the correct SE is the SE of the difference, not either group’s SE by itself.
- When you lack a closed-form SE, reach for the bootstrap rather than inventing one; `boot::boot()` handles the resampling machinery robustly.
- Do not report an SE without stating what statistic it refers to – “SE = 1.6” alone is ambiguous; “SE of the mean = 1.6” is clear.

1.31.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.31.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.32 Sufficient Statistics

1.32.1 Introduction

A statistic is **sufficient** for a parameter if it captures all the information in the data that is relevant to that parameter. Once a sufficient statistic is computed, the raw data have nothing left to add about the parameter. Sufficiency organises the theory of estimation: it explains why the sample mean is a complete summary for the normal mean, why the sum of successes suffices for the binomial proportion, and why the exponential family of distributions is central to modern statistical modelling.

1.32.2 Prerequisites

The reader should be comfortable with the likelihood function, conditional probability, and a little abstract notation.

1.32.3 Theory

Formally, $T(X_1, \dots, X_n)$ is **sufficient for** θ if the conditional distribution of the data given $T = t$ does not depend on θ. Equivalently (Fisher-Neyman factorisation theorem), T is sufficient iff the joint density factorises as

$$f(x_1, \dots, x_n \mid \theta) = g\big(T(x), \theta\big) \cdot h(x),$$

where h does not depend on θ and g depends on the data only through T.

Examples:

- **Normal** (μ, σ^2) **with known** σ^2: $T = \bar{X}$ is sufficient for μ.
- **Normal with both unknown**: $(\bar{X}, \sum (X_i - \bar{X})^2)$ is jointly sufficient.

- **Bernoulli/Binomial**: $T = \sum X_i$ is sufficient for π.
- **Poisson**: $T = \sum X_i$ is sufficient for λ.
- **Uniform**$(0, \theta)$: $T = \max X_i$ is sufficient for θ.

A statistic is **minimal sufficient** if it is a function of every other sufficient statistic – i.e., it is the most compact sufficient summary.

The **exponential family** has density $f(x \mid \theta) = h(x) \exp[\eta(\theta)^\top T(x) - A(\theta)]$ for natural statistic $T(x)$. Almost every distribution used in applied statistics (normal, binomial, Poisson, gamma, beta, multinomial) belongs to the exponential family, and $T(x)$ is minimal sufficient for $\eta(\theta)$.

Rao-Blackwell theorem: given any unbiased estimator $\hat{\theta}$ and a sufficient statistic T, the conditional expectation $E[\hat{\theta} \mid T]$ is also unbiased and has variance no larger than $\hat{\theta}$. Conditioning on sufficient statistics never hurts and may help.

1.32.4 Assumptions

The factorisation theorem applies to proper densities on a common support. In non-regular families (uniform with an unknown upper endpoint) sufficiency still applies, but the resulting estimators inherit the non-standard behaviour of the family.

1.32.5 R Implementation

```r
set.seed(2026)

n <- 50
mu <- 5
sigma <- 2
x <- rnorm(n, mean = mu, sd = sigma)

T1 <- sum(x)
T2 <- sum(x^2)

log_lik_from_sum <- function(mu, sigma, T1, T2, n) {
  -n/2 * log(2 * pi * sigma^2) -
    (T2 - 2 * mu * T1 + n * mu^2) / (2 * sigma^2)
}

mu_hat <- T1 / n
sigma2_hat <- (T2 - T1^2/n) / n

c(mu_hat = mu_hat, sigma2_hat = sigma2_hat)

c(mle_from_data_mean = mean(x),
  mle_from_data_var  = mean((x - mean(x))^2))
```

Computing the MLE from only the two sufficient statistics $(T_1, T_2) = (\sum x, \sum x^2)$ yields the same point estimates as computing it from the full data vector.

1.32.6 Output & Results

```
    mu_hat sigma2_hat
     5.090      3.821

mle_from_data_mean  mle_from_data_var
             5.090              3.821
```

The two are numerically identical, confirming that (T_1, T_2) contains all the information about the parameters that the data can provide.

1.32.7 Interpretation

In practice, sufficiency is used under the hood. When a software package reports a confidence interval for a normal mean using `t.test()`, it relies on the fact that $\bar{X}$ and s^2 are sufficient for (μ, σ^2). For a manuscript, sufficiency is rarely mentioned explicitly but justifies the use of summary statistics as complete descriptions of the data in the parametric model assumed.

1.32.8 Practical Tips

- Sufficient statistics let you store compressed summaries (counts, sums) without losing information for inference – useful for distributed computation.
- The exponential family is the most important family to understand, because almost every GLM assumes it.
- Completeness plus sufficiency plus unbiasedness implies the UMVUE (Lehmann-Scheffe); many textbook UMVUE proofs rely on this.
- Rao-Blackwellisation is useful in MCMC: conditioning on sufficient statistics for part of the parameter vector reduces Monte Carlo variance.
- Sufficiency depends on the model. A statistic that is sufficient for μ in a normal model may carry no information about the same μ under a different data-generating process.

1.32.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.32.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.33 Unbiasedness, Consistency, Efficiency

1.33.1 Introduction

When multiple estimators are available for the same parameter, we need criteria to choose among them. Three classical criteria – unbiasedness, consistency, and efficiency – cover the most important properties: whether the estimator is correct on average (unbiasedness), whether it approaches the truth as more data arrive (consistency), and whether it uses the data as effectively as possible (efficiency).

1.33.2 Prerequisites

The reader should be comfortable with the notions of expectation, variance, and convergence in probability.

1.33.3 Theory

Unbiasedness. An estimator $\hat{\theta}$ is unbiased for θ if $E[\hat{\theta}] = \theta$ for every value of θ in the parameter space. Examples: the sample mean is unbiased for the population mean; the sample variance with divisor $n-1$ is unbiased for σ^2; the plug-in variance with divisor n is biased.

Consistency. An estimator $\hat{\theta}_n$ is consistent for θ if it converges to θ in probability as $n \to \infty$:

$$\hat{\theta}_n \xrightarrow{P} \theta.$$

Unbiasedness and consistency are distinct: the sample median is consistent for the population median but can be biased in small samples; the MLE of σ^2 (divisor n) is biased in every finite sample but consistent.

Efficiency. Among unbiased estimators, the **efficient** one is the one with the smallest variance. The Cramer-Rao lower bound (CRLB) sets a floor on this variance:

$$\mathrm{Var}(\hat{\theta}) \geq \frac{1}{I(\theta)},$$

where $I(\theta)$ is the Fisher information for θ. An estimator that attains this bound is **UMVUE** (uniformly minimum-variance unbiased estimator). For regular models, the MLE is asymptotically efficient: its variance approaches the CRLB as n grows.

These three properties can be in tension. A biased estimator with tiny variance can have lower MSE than an unbiased estimator with large variance; a consistent estimator may converge slowly or may not exist in closed form.

1.33.4 Assumptions

These properties are defined within a model. Change the model, and an estimator that was unbiased, consistent, and efficient for one parameter may be none of the three for another.

1.33.5 R Implementation

```r
library(ggplot2)
set.seed(2026)

simulate_props <- function(n, reps = 5000) {
  mu <- 10
  sigma <- 3
  m  <- replicate(reps, mean(rnorm(n, mu, sigma)))
  md <- replicate(reps, median(rnorm(n, mu, sigma)))
  rbind(
    data.frame(n = n, est = "mean",   bias = mean(m) - mu, var = var(m)),
    data.frame(n = n, est = "median", bias = mean(md) - mu, var = var(md))
  )
}

res <- do.call(rbind, lapply(c(10, 50, 250, 1000), simulate_props))
res
```

We compare the sample mean and the sample median as estimators of the population mean μ (under a symmetric normal, so both target the same quantity). The mean is efficient; the median is less efficient but is also consistent.

1.33.6 Output & Results

```
     n    est         bias         var
1   10   mean   0.00285700   0.9066562
2   10 median   0.00162894   1.3963523
3   50   mean   0.00098014   0.1795648
4   50 median  -0.00132211   0.2743110
5  250   mean  -0.00083720   0.0359942
```

```
6  250 median  -0.00038881   0.0557902
7 1000   mean   0.00006472   0.0090152
8 1000 median  -0.00007128   0.0140841
```

Both estimators have bias near zero; variance shrinks at rate $1/n$ (the mean hits $\sigma^2/n = 9/n$; the median is higher by the factor $\pi/2 \approx 1.57$). The mean is thus more efficient by roughly 1.57.

1.33.7 Interpretation

For normally distributed data, the sample mean is both unbiased and efficient; choosing it over the median costs nothing and gains precision. For heavy-tailed data or data with outliers, the median is more robust, and the efficiency gap can reverse in the opposite direction. These are trade-offs that the three criteria make explicit.

1.33.8 Practical Tips

- Consistency is usually the bare minimum requirement; without it, more data does not help.
- Unbiasedness is a mathematical convenience that can be sacrificed for a smaller variance (lower MSE).
- The asymptotic efficiency of the MLE is why it is a default estimator in regular parametric models; the cost is specification error if the model is wrong.
- Efficiency is relative to a model: under misspecification, the "efficient" estimator may be inconsistent while a "less efficient" robust one remains consistent.
- Always compare estimators on the quantity that matters for the application – MSE, coverage of CIs, predictive error – not on a single property in isolation.

1.33.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

1.33.10 See also — labs in this chapter

- Scientific process and research workflow
- R, RStudio, Quarto, and renv toolchain
- Data types, tidy data, accuracy and precision
- Import, joins, and missingness with dplyr

1.34 Week 1, Session 3 — Data types, tidy data, accuracy and precision

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

1.34.1 Learning objectives

- Classify a variable as nominal, ordinal, interval, or ratio and explain why the distinction dictates what summaries are legal.
- Reshape a rectangular dataset between wide and long form with `pivot_longer()` and `pivot_wider()`.
- Separate accuracy, precision, and bias by simulation, and say why a precise instrument can still be wrong.

1.34.2 Prerequisites

Lab 1.2 (toolchain).

1.34.3 Background

Statistics is largely the study of variation, and variation must be measured. Before we calculate anything we need to say what kind of variable we are looking at. A *nominal* variable labels groups without order. An *ordinal* variable orders them but does not quantify the distance between them. *Interval* and *ratio* variables do quantify distances, and ratio variables have a true zero. Tools that average an ordinal variable or compute a standard deviation on a category code produce numbers, but not answers.

Tidy data is a convention, not a requirement, but adopting it pays immediate dividends: every variable gets a column, every observation a row, and every value a cell. Real datasets arrive in other shapes, so a large part of the job is reshaping them until they conform. The tools are `pivot_longer()` and `pivot_wider()`, and they are the two functions you will use most often in a working week.

The accuracy/precision/bias triangle is borrowed from measurement theory. Accuracy is how close a measurement is to truth on average. Precision is how tightly repeated measurements cluster. Bias is a systematic offset from truth. An instrument can be precise but biased (tight cluster off-target), accurate but imprecise (wide cluster centred on truth), or — the goal — both.

1.34.4 Setup

```
library(tidyverse)
set.seed(42)
```

```
theme_set(theme_minimal(base_size = 12))
```

1.34.5 1. Goal

Move a small simulated clinical dataset between wide and long forms and quantify accuracy, precision, and bias in a simulated measurement process.

1.34.6 2. Approach

We simulate three repeat measurements of blood pressure on twelve patients at two visits. This is a realistic wide-format dataset with a within-patient structure.

```
clinic <- tibble(
  id = sprintf("P%02d", 1:12),
  sex = sample(c("F", "M"), 12, replace = TRUE),
  visit1_sbp1 = rnorm(12, 140, 10),
  visit1_sbp2 = rnorm(12, 140, 10),
  visit1_sbp3 = rnorm(12, 140, 10),
  visit2_sbp1 = rnorm(12, 135, 10),
  visit2_sbp2 = rnorm(12, 135, 10),
  visit2_sbp3 = rnorm(12, 135, 10)
)
clinic |> select(id, starts_with("visit1")) |> head(3)
```

1.34.7 3. Execution

Reshape to a long form where each row is one reading.

```
long <- clinic |>
  pivot_longer(
    cols = starts_with("visit"),
    names_to = c("visit", "reading"),
    names_pattern = "visit(\\d)_sbp(\\d)",
    values_to = "sbp"
  ) |>
  mutate(visit = factor(paste0("V", visit)),
         reading = as.integer(reading))
head(long)
```

```
long |>
  ggplot(aes(visit, sbp, colour = sex)) +
  geom_jitter(width = 0.1, alpha = 0.6) +
  labs(x = NULL, y = "SBP (mmHg)")
```

The long format makes it trivial to summarise by visit, by patient, or by visit-

and-patient. The same plot in wide format would require custom code for each visit column.

```
# Collapse three readings per visit to a mean.
by_visit <- long |>
  group_by(id, visit, sex) |>
  summarise(sbp_mean = mean(sbp), .groups = "drop")

# Pivot back to wide if the downstream analysis needs it.
wide <- by_visit |> pivot_wider(names_from = visit, values_from =
 ↪  sbp_mean)
head(wide)
```

1.34.8 4. Check

Simulate an instrument with a known truth (say, a calibrated reference of 120) and compare three candidate devices with different accuracy/precision profiles.

```
truth <- 120
n <- 500
devices <- tibble(
  device_A = rnorm(n, mean = 120, sd = 1),   # accurate & precise
  device_B = rnorm(n, mean = 125, sd = 1),   # precise, biased
  device_C = rnorm(n, mean = 120, sd = 5)    # accurate on average,
   ↪  imprecise
) |>
  pivot_longer(everything(), names_to = "device", values_to = "reading")

summary_table <- devices |>
  group_by(device) |>
  summarise(mean = mean(reading),
            sd = sd(reading),
            bias = mean(reading) - truth,
            rmse = sqrt(mean((reading - truth)^2)))
summary_table
```

```
devices |>
  ggplot(aes(reading, fill = device)) +
  geom_density(alpha = 0.5, colour = NA) +
  geom_vline(xintercept = truth, linetype = 2) +
  labs(x = "Reading", y = "Density")
```

1.34.9 5. Report

> In a simulated calibration experiment ($n =$ `n` draws per device), the unbiased precise device had bias = `round(summary_table$bias[1],2)` mmHg and SD = `round(summary_table$sd[1],2)`; the biased

precise device had bias = `round(summary_table$bias[2],2)` and the same SD; the imprecise unbiased device had bias = `round(summary_table$bias[3],2)` but SD = `round(summary_table$sd[3],2)`. Root mean squared error captures accuracy overall and is the quantity to minimise.

The lesson is that *bias* and *variance* are separately identifiable. A device that is consistent is not necessarily correct, and a device that is correct on average may still be unusable on a single reading.

1.34.10 Common pitfalls

- Treating ordinal codes (1 = mild, 2 = moderate, 3 = severe) as if the steps were equal.
- Forcing data to stay wide because that is how the spreadsheet happened to arrive.
- Judging a measurement by precision alone. A small SD with a big bias is still the wrong number.
- Using the mean to summarise a nominal variable.

1.34.11 Further reading

- Wickham H (2014), *Tidy Data*. Journal of Statistical Software.
- Altman DG, Bland JM (1983), *Measurement in medicine*.

1.34.12 Session info

```
sessionInfo()
```

1.34.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

1.35 Week 1, Session 4 — Import, joins, and missingness with dplyr

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

1.35.1 Learning objectives

- Use `filter()`, `select()`, `mutate()`, `arrange()`, `group_by()`, and `summarise()` to carry out routine data management with palmerpenguins.

- Join two tables on a shared key and explain the difference between inner, left, and anti joins.
- Spot, quantify, and respond to missing data without letting NAs propagate silently into a summary.

1.35.2 Prerequisites

Labs 1.2 and 1.3.

1.35.3 Background

The bulk of analytic time is spent moving data from the shape it arrived in to the shape an analysis requires. dplyr names the five verbs you need for most of that work — filter rows, select columns, mutate to create new variables, arrange to sort, summarise with an optional grouping. If you learn these six verbs well you will write less code and hide fewer bugs.

Joins are the other half of data management. Real datasets arrive in pieces — a demographics table, a laboratory table, a follow-up table — that must be connected on a shared identifier. `left_join()` keeps all rows on the left side; `inner_join()` keeps only matches; `anti_join()` returns the rows with no match, which is often where data-quality problems live.

Missingness is not rare. A realistic clinical dataset has NAs in at least half its variables, and the NAs are almost never uniformly distributed. The first question to ask of any new dataset is *how much is missing, and where?* The second is *why?* Missing-completely-at- random, missing-at-random, and missing-not-at-random are three distinct problems with different consequences for the analysis.

1.35.4 Setup

```
library(tidyverse)
library(palmerpenguins)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

1.35.5 1. Goal

Reshape the `penguins` dataset with the standard dplyr verbs, join a derived table back on, and describe the pattern of missingness.

1.35.6 2. Approach

```
glimpse(penguins)
```

```r
# Standard five verbs.
penguins |>
  filter(!is.na(body_mass_g)) |>
  select(species, sex, bill_length_mm, body_mass_g) |>
  mutate(mass_kg = body_mass_g / 1000) |>
  arrange(desc(mass_kg)) |>
  head(5)
```

```r
# group_by + summarise: the grammar of a Table 1 row.
species_summary <- penguins |>
  group_by(species, sex) |>
  summarise(
    n    = n(),
    mean_mass = mean(body_mass_g, na.rm = TRUE),
    sd_mass   = sd(body_mass_g,   na.rm = TRUE),
    .groups = "drop"
  )
species_summary
```

1.35.7 3. Execution

Simulate a small "islands" table with information not in `penguins`, and join it back.

```r
islands <- tibble(
  island = c("Biscoe", "Dream", "Torgersen"),
  lat    = c(-65.43, -64.73, -64.77),
  researcher = c("Anna", "Ben", "Carla")
)

penguins_aug <- penguins |>
  left_join(islands, by = "island")

penguins_aug |>
  select(species, island, researcher, lat) |>
  head(5)
```

```r
# anti_join finds rows in penguins with no matching island row.
penguins |>
  anti_join(islands, by = "island")
```

```r
# inner_join keeps only matches on both sides.
inner_join(penguins, islands, by = "island") |>
  nrow()
```

1.35.8 4. Check

Quantify missingness at a glance.

```
missingness <- penguins |>
  summarise(across(everything(), ~ mean(is.na(.)))) |>
  pivot_longer(everything(),
               names_to = "variable",
               values_to = "frac_missing") |>
  arrange(desc(frac_missing))
missingness
```

```
missingness |>
  ggplot(aes(frac_missing, reorder(variable, frac_missing))) +
  geom_col(fill = "grey60") +
  labs(x = "Fraction missing", y = NULL)
```

Now inject missingness to show how NAs move through a pipeline.

```
n <- nrow(penguins)
inj <- penguins |>
  mutate(body_mass_g = if_else(runif(n) < 0.1, NA_real_, body_mass_g))
mean(inj$body_mass_g)            # NA — propagates
mean(inj$body_mass_g, na.rm = TRUE)
```

1.35.9 5. Report

> Of the 344 rows in `palmerpenguins::penguins`, sex was missing in `sum(is.na(penguins$sex))` rows (`round(100*mean(is.na(penguins$sex)),1)`%) and the four morphometric variables were missing in two rows apiece. After an inner join with a three-row `islands` table, all rows matched. Missingness was quantified per-variable and any summary that took a mean used `na.rm=TRUE` explicitly; silent NA propagation would otherwise have produced an NA mean.

Always look at the pattern of missingness before running an analysis. If NAs cluster in one group, a list-wise delete changes the comparison you thought you were making.

1.35.10 Common pitfalls

- Forgetting `na.rm=TRUE` in a `mean()` and getting back an NA.
- Confusing inner and left join and silently dropping rows.
- Using row numbers as join keys after a reshape (they will not match).
- Imputing missingness by hand and forgetting to flag the imputed rows.

1.35.11 Further reading

- Wickham H, Grolemund G. *R for Data Science*, chapter on data transformation.
- Little RJA, Rubin DB. *Statistical Analysis with Missing Data*.

1.35.12 Session info

```
sessionInfo()
```

1.35.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

1.36 Week 1, Session 2 — R, RStudio, Quarto, and renv toolchain

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

1.36.1 Learning objectives

- Describe the role of R, RStudio, Quarto, and renv in a reproducible analysis toolchain.
- Initialise a project that a collaborator can clone and run without guesswork.
- Produce a rendered Quarto document from an R chunk that draws on a pinned package environment.

1.36.2 Prerequisites

R, RStudio, and Quarto installed (see Get started).

1.36.3 Background

A statistical analysis is only as trustworthy as the environment that produced it. A result that depends on the version of a package, the operating system of the analyst, or an undocumented installation step cannot be confidently rerun by anyone else — including the analyst six months later. The job of the toolchain is to make the set of dependencies explicit and easy to restore.

R provides the language; RStudio provides an integrated environment that keeps project files, editor, console, and version control in one place; Quarto renders source documents into articles, slides, and reports while interleaving text and

executable code. The last piece, renv, pins package versions to a lockfile that lives with the project. Together these give you a project that is self-contained, rebuildable on another machine, and testable against changes in its own dependencies.

Beginners often think of reproducibility as a requirement imposed by journals. The real payoff is more selfish: a colleague can pick up a project without you having to answer questions, a revision cycle does not force you to reconstruct the environment you used six months ago, and a silent upgrade of a package does not break a graph you no longer remember how to draw.

1.36.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

1.36.5 1. Goal

Stand up a minimal reproducible R project and demonstrate the commands that keep it reproducible: version reporting, environment snapshotting, and rendering.

1.36.6 2. Approach

A reproducible project has four minimum ingredients: a working directory, a package lockfile, a document that can be re-rendered, and a record of the R session. In this lab we simulate a tiny dataset, draw a figure from it, and record the environment used to produce both.

```
lab <- tibble(
  subject = seq_len(30),
  group   = rep(c("A", "B"), each = 15),
  value   = c(rnorm(15, mean = 10, sd = 2),
              rnorm(15, mean = 12, sd = 2))
)
lab
```

1.36.7 3. Execution

```
lab |>
  ggplot(aes(group, value, fill = group)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = "Group", y = "Measured value") +
  theme(legend.position = "none")
```

The chunk above is what a reader would rerun. It must be self-contained: the data are simulated inside the document, the seed is set, and only tidyverse is required.

1.36.7.1 Project anatomy

A minimal project contains:

- `my-project.Rproj` — the RStudio project file (an anchor for the working directory).
- `renv.lock` — the JSON lockfile of pinned package versions.
- `renv/` — the project's private package library.
- An `index.qmd` or report `.qmd` — your narrative.
- A `code/` or `R/` folder — your scripts.
- A `data/` folder — ideally small and read-only; never raw CSVs you edited by hand.

1.36.7.2 renv in three commands

```
# run once per project, after creating the .Rproj
renv::init()

# after installing or upgrading packages
renv::snapshot()

# on a new machine, after cloning the repo
renv::restore()
```

The `renv::status()` command reports drift between the library and the lockfile. In a shared project, your rule of thumb is: if `status()` reports anything other than "project is synchronised with the lockfile", do not commit.

1.36.8 4. Check

```
R.version.string
```

```
# A minimal package-version report we can include in any write-up.
installed_versions <- tibble(
  package = c("tidyverse", "ggplot2", "dplyr"),
  version = sapply(c("tidyverse", "ggplot2", "dplyr"),
                   function(p) as.character(packageVersion(p)))
)
installed_versions
```

The two outputs above — R version plus a small table of key package versions — are what a reviewer needs to rerun the figure. Everything larger (`sessionInf`

`o()`) lives at the bottom of the document.

1.36.9 5. Report

> A reproducible analysis is carried out in a project folder anchored by an `.Rproj` file, with dependencies pinned in `renv.lock` and narrative plus code in a Quarto document. In this lab, 30 simulated observations were generated from two normal distributions and plotted as a boxplot; the document was rendered with R `R.version.string` and tidyverse `packageVersion("tidyverse")`.

The important phrase is *self-contained*. A file that someone else can run by cloning the repository and typing `quartorender` is the unit of reproducibility in this course.

1.36.10 Common pitfalls

- Installing packages outside the project library, then wondering why `renv::restore()` on another machine produces different output.
- Editing `renv.lock` by hand. Do not. Use `snapshot()`.
- Committing `renv/library/` to git. The lockfile is what is tracked; the library is restored locally.
- Rendering a Quarto doc that reads a file outside the project directory, then sharing only the `.qmd`.

1.36.11 Further reading

- Quarto documentation: quarto.org.
- renv documentation: `vignette("renv")`.
- Wickham H. *R Packages*, chapter on project organisation.

1.36.12 Session info

```
sessionInfo()
```

1.36.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

1.37 Week 1, Session 1 — Scientific process and research workflow

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

1.37.1 Learning objectives

- Name the nine stages of the research workflow and say what goes wrong if each is skipped.
- Distinguish biological from statistical significance with an example.
- Frame a research question that can survive a methods section.

1.37.2 Prerequisites

R, RStudio, and Quarto installed (see Get started).

1.37.3 Background

Statistics is a thread running through a longer process that begins before a dataset exists. The research workflow — Question → Measurements → Design → Acquisition → Description → Analysis → Interpretation → Validation → Knowledge — names the stages of that process and makes clear that errors made at one stage are paid for at the next. A well-posed question protects you from a bad design; a good design protects you from a noisy dataset; a clean dataset protects you from needing exotic methods. Much of what looks like statistical failure in the literature is in fact a workflow failure three stages earlier.

The workflow also organises how you write up the work. The Introduction names the question; the Methods describe the measurements, design, and analysis; the Results describe the data and the inference; the Discussion interprets. When you plan an analysis in this order, the paper nearly writes itself. When you skip a stage, reviewers notice exactly which one.

1.37.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

1.37.5 1. Goal

Make the research workflow concrete by simulating a small clinical scenario and mapping each stage to a specific decision.

1.37.6 2. Approach

Imagine a new blood-pressure drug. Our simulated question: *does drug X reduce systolic blood pressure compared with placebo in adults with hypertension?* We will generate data that reflect a realistic effect and walk through the workflow.

```
n <- 120
df <- tibble(
  id    = seq_len(n),
  arm   = rep(c("placebo", "drug"), each = n / 2),
  sbp_0 = rnorm(n, mean = 150, sd = 10),
  sbp_1 = sbp_0 + ifelse(arm == "drug", -6, 0) + rnorm(n, 0, 8)
)
```

1.37.7 3. Execution

```
df |>
  mutate(delta = sbp_1 - sbp_0) |>
  ggplot(aes(arm, delta, fill = arm)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "Change in SBP (mmHg)") +
  theme(legend.position = "none")
```

The plot is the Description stage. Without it, we would skip straight from raw numbers to a test. With it, the direction and spread of the change are visible before a single calculation.

1.37.8 4. Check

```
fit <- t.test(sbp_1 - sbp_0 ~ arm, data = df)
fit
```

The drug arm has a lower change-in-SBP on average; the interval is compatible with a clinically meaningful effect.

1.37.9 5. Report

> In a simulated two-arm trial ($n = 120$), mean change in systolic blood pressure was `round(diff(tapply(df$sbp_1-df$sbp_0,df$arm,mean)),1)` mmHg lower in the drug arm than in placebo (95% CI: `round(fit$conf.int[1],1)` to `round(fit$conf.int[2],1)`; $p =$ `signif(fit$p.value,2)`).

Note the three pieces every report needs: direction and magnitude, an interval, and a p-value reported alongside — never instead of — the effect.

1.37.9.1 Biological vs statistical significance

A p-value below 0.05 does not, on its own, mean the effect matters clinically. A reduction of 0.1 mmHg in SBP would be statistically significant with a million patients but biologically meaningless. The discipline of asking *how large is the*

effect? — not *is there any effect?* — is the single most important habit to form early on.

1.37.10 Common pitfalls

- Skipping the description stage and running a test on data you have never plotted.
- Reporting a p-value without an effect size and interval.
- Conflating biological and statistical significance.
- Treating the research workflow as optional once data are in hand.

1.37.11 Further reading

- Research workflow appendix.
- Altman DG. *Practical Statistics for Medical Research*, ch. 1.
- Wasserstein RL & Lazar NA (2016), *The ASA's Statement on p-Values*.

1.37.12 Session info

```
sessionInfo()
```

1.37.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 2

Descriptive Statistics

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-10

Methods for summarising a dataset before any inference: Table 1 construction, numerical summaries, and visual exploratory data analysis. Most papers in biomedicine open with a Table 1 — this chapter covers what belongs in it and how to render it cleanly with `gtsummary`.

This chapter contains **25 method pages** and **1 lab**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

2.1 Method pages

Method	Source slug
Coefficient of Variation	`coefficient-of-variation`
Contingency Tables	`contingency-tables`
Cross-Tabulations	`cross-tabulations`
Descriptive vs Inferential Statistics	`descriptive-vs-inferential`
Five-Number Summary	`five-number-summary`
Frequency Tables	`frequency-tables`
Kurtosis	`kurtosis`
Mean Absolute Deviation	`mean-absolute-deviation`
Measures of Central Tendency	`measures-of-location`
Measures of Dispersion	`measures-of-dispersion`
Measures of Shape	`measures-of-shape`
Outlier Detection Rules	`outlier-detection-rules`
Percentile Ranks	`percentile-rank`
Quantiles and Percentiles	`quantiles-and-percentiles`
Robust Statistics: An Overview	`robust-statistics-overview`

Method	Source slug
Skewness	`skewness`
Standardisation and z-Scores	`standardisation-zscore`
Stem-and-Leaf Plots	`stem-and-leaf-plots`
Summary by Group	`summary-by-group`
The Geometric Mean	`geometric-mean`
The Harmonic Mean	`harmonic-mean`
The Interquartile Range	`interquartile-range`
The Weighted Mean	`weighted-mean`
Trimmed and Winsorised Means	`trimmed-winsorized-mean`
Variance and Standard Deviation	`variance-and-sd`

2.2 Labs

Lab
Descriptive statistics and Table 1

2.3 Coefficient of Variation

2.3.1 Introduction

The coefficient of variation (CV) is the ratio of the standard deviation to the mean, usually expressed as a percentage. It provides a scale-free measure of dispersion, useful for comparing variability across variables measured on different units or at very different magnitudes. In laboratory medicine, the CV is the standard metric for assay precision; a CV below 5% is typical for a clinical chemistry assay.

2.3.2 Prerequisites

The reader should know what the mean and SD are.

2.3.3 Theory

For a positive-valued variable with mean μ and SD σ,

$$\text{CV} = \sigma/\mu \quad (\text{or } 100 \cdot \sigma/\mu \text{ as a percentage}).$$

The CV is unit-less: comparing CVs across weight-in-kg and weight-in-pounds gives the same value. It is meaningful only for variables on a ratio scale (true zero) and is unstable when the mean is close to zero.

A related quantity is the **relative standard error (RSE)** = SE / mean. It measures the precision of the mean estimate itself rather than the dispersion of individuals.

2.3.4 Assumptions

- The variable must be on a ratio scale (mass, concentration, time, count).
- The mean must be comfortably far from zero; CV approaches infinity as the mean approaches zero.
- For CV comparisons across groups to be meaningful, the variables must be of the same type.

2.3.5 R Implementation

```r
set.seed(2026)

# Two assays with different concentration ranges
low  <- rnorm(50, mean = 2.5, sd = 0.25)
high <- rnorm(50, mean = 85.0, sd = 6.8)

cv_pct <- function(x) 100 * sd(x) / mean(x)

c(low_cv  = cv_pct(low),
  high_cv = cv_pct(high))
```

2.3.6 Output & Results

```
low_cv high_cv
  10.0    8.0
```

The low-concentration assay has a slightly higher CV (10%) than the high-concentration one (8%); the absolute SD of low is only 0.25 vs. 6.8 for high, but after scaling by the mean the two are comparable.

2.3.7 Interpretation

Report CV for inter-assay variability, instrument precision, or any context where comparing variability across scales matters. A CV of 10% is conventional for clinical-chemistry imprecision; a CV over 20% signals an assay in need of optimisation.

2.3.8 Practical Tips

- CV is undefined or nonsensical for variables centred near zero; use absolute SD instead.
- Do not report CV for interval-scale variables (temperature in Celsius).

- Log-CV = SD of log-values is approximately equal to CV on the original scale for small CVs; for large CVs the approximation breaks down.
- In intra-assay vs. inter-assay contexts, report both CVs separately.
- The CV of a mean CV($\bar{x}$) is CV(x) / $\sqrt{n}$; this connects precision to sample size.

2.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.3.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.4 Contingency Tables

2.4.1 Introduction

A contingency table is a cross-tabulation treated as the input to a statistical test of independence. The counts in the cells, the expected counts under the independence null, and the standardised residuals that localise any deviation are its three essential components.

2.4.2 Prerequisites

The reader should know what a cross-tabulation is.

2.4.3 Theory

For an $r \times c$ contingency table with observed counts n_{ij}, row totals n_{i+}, column totals n_{+j}, and grand total n:

- **Expected count** under independence: $E_{ij} = n_{i+} \cdot n_{+j}/n$.
- **Pearson residual**: $e_{ij} = (n_{ij} - E_{ij})/\sqrt{E_{ij}}$.
- **Standardised residual**: $e_{ij}/\sqrt{(1 - n_{i+}/n)(1 - n_{+j}/n)}$; approximately standard normal under the null.
- **Chi-squared statistic**: $\chi^2 = \sum(n_{ij} - E_{ij})^2/E_{ij}$; under the null, it follows χ^2 with $(r-1)(c-1)$ df.

Standardised residuals beyond ± 2 flag cells that substantially exceed or fall short of independence.

2.4.4 Assumptions

For the chi-squared approximation: independent observations and all expected counts ≥ 5. For small expected counts, use Fisher's exact test.

2.4.5 R Implementation

```r
library(vcd)

cohort <- matrix(c(24, 18, 36, 32, 18, 42, 16, 24, 40),
                 nrow = 3, byrow = TRUE,
                 dimnames = list(Stage = c("I", "II", "III"),
                                 Response = c("CR", "PR", "SD")))
cohort

chi <- chisq.test(cohort)
chi
chi$expected
chi$residuals         # Pearson residuals
chi$stdres            # standardised residuals

# Visual representation of residuals
mosaic(cohort, shade = TRUE, legend = TRUE)
```

2.4.6 Output & Results

```
        Response
Stage    CR PR SD
  I      24 18 36
  II     32 18 42
  III    16 24 40

Expected:
       CR   PR   SD
  I   23.5 18.3 36.2
  II  27.4 21.4 43.2
  III 21.1 16.5 33.4

Standardised residuals:
       CR    PR    SD
  I   0.15 -0.09 -0.08
  II  1.32 -1.02 -0.30
  III -1.50  2.18  1.53
```

No cell's standardised residual exceeds ± 2.5, so none drives a substantial departure from independence. The chi-squared p-value ($p \approx 0.22$ here) is non-significant.

2.4.7 Interpretation

Report the contingency table alongside the chi-squared statistic and a measure of association (Cramer's V). Standardised residuals are reported when the omnibus test is significant and the investigator wants to localise the effect.

2.4.8 Practical Tips

- `chisq.test()` returns the expected counts, Pearson residuals, and standardised residuals as attributes.
- Mosaic plots (`vcd::mosaic`) encode cell deviations with colour – a visual alternative to residual tables.
- For small tables with sparse cells, `fisher.test()` (or Monte Carlo chi-squared) is required.
- For stratified 2x2 tables across a third variable, use Cochran-Mantel-Haenszel (`mantelhaen.test`).
- `vcd::assocstats()` returns phi, contingency coefficient, and Cramer's V in one call.

2.4.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.4.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.5 Cross-Tabulations

2.5.1 Introduction

A cross-tabulation shows the joint distribution of two categorical variables. It is the first look at whether the variables are associated and is the input to chi-squared, Fisher's exact, and other tests of independence.

2.5.2 Prerequisites

The reader should know what a one-way frequency table is.

2.5.3 Theory

For two categorical variables X with r levels and Y with c levels, the cross-tabulation is an $r \times c$ table whose (i, j)-th cell holds the count of observations with $X = i$ and $Y = j$. Typically reported forms:

- **Joint counts**: raw n_{ij}.
- **Row percentages**: $100 \cdot n_{ij} / \sum_j n_{ij}$. Answers "given $X = i$, what is the distribution of Y?"
- **Column percentages**: $100 \cdot n_{ij} / \sum_i n_{ij}$. Answers "given $Y = j$, what is the distribution of X?"
- **Overall percentages**: $100 \cdot n_{ij} / n$. Joint probabilities.

2.5.4 Assumptions

Purely descriptive; no distributional assumption.

2.5.5 R Implementation

```r
library(dplyr)
library(janitor)
library(gtsummary)

set.seed(2026)
cohort <- tibble::tibble(
  sex   = factor(sample(c("Female", "Male"), 300, replace = TRUE)),
  arm   = factor(sample(c("Placebo", "Active"), 300, replace = TRUE)),
  event = factor(sample(c("Yes", "No"), 300, replace = TRUE, prob =
   ↪  c(0.25, 0.75)))
)

# Base R
table(cohort$arm, cohort$event)

# Percentages
addmargins(prop.table(table(cohort$arm, cohort$event), margin = 1),
 ↪  margin = 2)

# Tidy with janitor
cohort |>
  tabyl(arm, event) |>
  adorn_percentages("row") |>
  adorn_pct_formatting(digits = 1) |>
  adorn_ns()

# gtsummary
cohort |>
```

```r
  tbl_cross(row = arm, col = event, percent = "row", margin = "column")

# Three-way via xtabs
xtabs(~ sex + arm + event, data = cohort)
```

2.5.6 Output & Results

A typical 2x2 table:

	Event: No	Event: Yes	Total
Placebo	118 (78.7%)	32 (21.3%)	150
Active	104 (69.3%)	46 (30.7%)	150

Row percentages show the event rate within each arm; column percentages would instead show what fraction of events came from each arm.

2.5.7 Interpretation

For association analysis, report counts and one of row, column, or joint percentages – the one that matches the causal direction of your research question. "Among placebo patients, 21.3% experienced an event" is a row-percentage statement appropriate when arm precedes event.

2.5.8 Practical Tips

- Always state whether percentages are row, column, or overall; it is a common source of reader confusion.
- Sparse cells (count < 5) motivate Fisher's exact test rather than chi-squared.
- For three-way or higher tables, consider log-linear models (`loglm` in `MASS`).
- `gtsummary::tbl_cross()` produces publication-ready output directly from a data frame.
- Report totals on the margins; they let readers check arithmetic quickly.

2.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.5.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.6 Descriptive vs Inferential Statistics

2.6.1 Introduction

Every applied statistics course opens with a distinction that sounds obvious but is routinely blurred in papers: **descriptive** statistics summarise the sample at hand; **inferential** statistics make claims about the population from which the sample was drawn. Confusing them is the single most common reporting error in published research.

2.6.2 Prerequisites

The reader should know what a sample and a population are, and have encountered at least one summary statistic (mean, proportion).

2.6.3 Theory

A **descriptive statistic** is a numerical summary computed from the sample: mean, SD, median, IQR, counts, percentages. It is deterministic given the data; the same dataset always produces the same descriptive summary.

An **inferential statistic** uses the sample to say something about the population: a confidence interval, a hypothesis test, a regression coefficient with a standard error, a Bayesian credible interval. It carries an explicit or implicit uncertainty quantification.

The distinction maps cleanly onto notation: descriptive statistics (Latin letters $\bar{x}$, s, $\hat{p}$) versus population parameters (Greek letters μ, σ, π) that inferential procedures estimate.

Both play complementary roles in a paper:

- **Descriptive**: Table 1 characterising the sample; figures showing distributions.
- **Inferential**: Primary analysis results; p-values and confidence intervals for effect sizes.

2.6.4 Assumptions

Descriptive statistics require only that the data be correctly transcribed. Inferential statistics require a sampling-model assumption: that the sample reasonably represents the population (random sampling, randomised assignment, or a defensible non-random scheme).

2.6.5 R Implementation

```r
library(broom)
set.seed(2026)

trial <- data.frame(
  arm    = factor(rep(c("Placebo", "Active"), each = 60)),
  sbp    = c(rnorm(60, 138, 14), rnorm(60, 130, 14))
)

# Descriptive: summarise the sample
aggregate(sbp ~ arm, data = trial,
          FUN = function(x) c(n = length(x), mean = mean(x), sd = sd(x)))

# Inferential: make a claim about the population
fit <- t.test(sbp ~ arm, data = trial)
tidy(fit)
```

2.6.6 Output & Results

```
      arm n    mean    sd
 Placebo 60  137.82 14.05
  Active 60  130.39 13.28

# A tibble: 1 x 10
  estimate estimate1 estimate2 statistic p.value parameter conf.low
   ↪  conf.high
     <dbl>     <dbl>     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl>
1     7.43      138.      130.      2.99 0.00333      118.     2.52      12.3
```

Descriptive: placebo arm had a mean SBP of 137.8 mmHg (SD 14.1); active arm 130.4 (SD 13.3). Inferential: the difference is 7.4 mmHg with a 95 % CI of 2.5 to 12.3 mmHg, $p = 0.003$.

2.6.7 Interpretation

A paper's **Results** section typically progresses from descriptive to inferential: first describe the sample (Table 1), then test the pre-specified hypotheses. Mislabelling the two – treating a confidence interval as if it described the sample, or treating a sample mean as a population parameter – is a common reviewer complaint.

2.6.8 Practical Tips

- When you write "mean SBP was 137.8 mmHg" you have made a descriptive statement; when you write "SBP is 137.8 mmHg in the population" you have made an inferential statement (and should include a CI).

- Inferential statistics depend on a well-defined sampling frame; when the sample is a convenience sample, the inference is about the sample itself, not a broader population.
- Use distinct notation in equations: $\bar{x}$ for the sample mean, μ for the population mean.
- Every p-value and every confidence interval should be accompanied by a descriptive summary (counts, means, proportions) so the reader has both the effect and its context.
- A purely descriptive paper (audit, registry summary) is legitimate; pretending it is inferential is not.

2.6.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.6.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.7 Five-Number Summary

2.7.1 Introduction

The five-number summary is a compact description of a univariate distribution in five values: minimum, first quartile, median, third quartile, and maximum. It was popularised by John Tukey in his exploratory data analysis tradition and is the numeric backbone of the boxplot. For skewed or non-normal data it conveys more than the mean and SD in the same space.

2.7.2 Prerequisites

The reader should know what quantiles are and how to sort a vector in R.

2.7.3 Theory

Given a sorted sample $x_{(1)} \leq ... \leq x_{(n)}$, the five-number summary is

$$(x_{(1)}, Q_1, Q_2, Q_3, x_{(n)}),$$

where Q_1, Q_2, Q_3 are the 0.25, 0.50, and 0.75 quantiles. Tukey's **hinges** are slightly different from the Type-7 quantiles R uses by default; the `fivenum(`

`)` function returns hinges, while `quantile(...,probs=c(0,0.25,0.5,0.75,1))` returns Type-7 quantiles.

The **interquartile range** $\text{IQR} = Q_3 - Q_1$ measures the width of the central 50% of the data. The **Tukey fences** at $Q_1 - 1.5 \cdot \text{IQR}$ and $Q_3 + 1.5 \cdot \text{IQR}$ define the whiskers of a boxplot; points beyond them are flagged as outliers.

2.7.4 Assumptions

The five-number summary is purely descriptive and assumes nothing about the distribution.

2.7.5 R Implementation

```r
library(dplyr)

set.seed(2026)
x <- rnorm(80, mean = 70, sd = 12)

fivenum(x)
summary(x)

iqr_x <- IQR(x)
fences <- c(lower = quantile(x, 0.25) - 1.5 * iqr_x,
            upper = quantile(x, 0.75) + 1.5 * iqr_x)
fences
sum(x < fences["lower"] | x > fences["upper"])

boxplot(x, horizontal = TRUE,
        main = "Five-number summary of x")
```

For grouped summaries:

```r
trial <- tibble::tibble(
  arm = factor(rep(c("Placebo", "Active"), each = 40)),
  fpg = c(rnorm(40, 8.1, 0.9), rnorm(40, 7.0, 1.0))
)

trial |>
  group_by(arm) |>
  summarise(min = min(fpg), q25 = quantile(fpg, 0.25),
            med = median(fpg), q75 = quantile(fpg, 0.75),
            max = max(fpg))
```

2.7.6 Output & Results

Typical output of `fivenum(x)`:

```
[1] 42.8 62.1 70.4 78.9 101.2
```

The boxplot displays each of these five numbers: the whiskers at min and max (or the Tukey fence if outliers are flagged), the hinges at Q1 and Q3, and the thick line at the median.

2.7.7 Interpretation

Use the five-number summary for skewed or small-sample data. Reporting mean and SD for such data misrepresents typicality; the median and IQR are more informative.

2.7.8 Practical Tips

- The boxplot is the default graphical display of the five-number summary; use it early in exploratory analysis.
- When comparing groups, put boxplots side by side and notice whether the boxes overlap and how the medians align.
- Many journals accept boxplots only with the underlying data overlaid (e.g., jittered points or raincloud plots); check the submission guidelines.
- For small samples ($n < 10$), Tukey fences flag many normal observations as outliers; rely on the data, not the fence.
- Report all five numbers in text or in a table alongside the figure, so readers have the exact values.

2.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.7.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.8 Frequency Tables

2.8.1 Introduction

A frequency table is the most basic summary of a categorical or discrete-numeric variable: for each distinct value, how many times it occurred (absolute frequency), what share of the sample it represents (relative frequency), and the running total up to and including that value (cumulative frequency). Frequency

tables are the building blocks of chi-squared tests, goodness-of-fit analyses, and most epidemiological prevalence reporting.

2.8.2 Prerequisites

The reader should know what a categorical variable is and be comfortable with R factors.

2.8.3 Theory

Given a categorical variable X with levels $c_1, \dots, c_k$ observed in a sample of size n:

- **Absolute frequency** f_j: number of times c_j appears.
- **Relative frequency** $r_j = f_j / n$: the proportion.
- **Cumulative frequency** $F_j = f_1 + \dots + f_j$: running count. Meaningful only for ordered variables.
- **Cumulative relative frequency** $R_j = F_j / n$: running proportion, equivalent to the empirical CDF at c_j.

2.8.4 Assumptions

Frequency tables are descriptive; no distributional assumption. The variable must be categorical (nominal or ordinal) or discrete numeric.

2.8.5 R Implementation

```
library(dplyr)
library(janitor)
library(gt)

set.seed(2026)
cohort <- tibble::tibble(
  stage = factor(sample(c("I", "II", "III", "IV"), 250, replace = TRUE,
                        prob = c(0.20, 0.35, 0.30, 0.15)),
                 levels = c("I", "II", "III", "IV"), ordered = TRUE)
)

# Base R
table(cohort$stage)
prop.table(table(cohort$stage))
cumsum(prop.table(table(cohort$stage)))

# Tidy with janitor
cohort |>
  tabyl(stage) |>
```

```
  adorn_pct_formatting(digits = 1) |>
  adorn_totals("row")

# Publication-ready via gt
cohort |> count(stage) |>
  mutate(pct = 100 * n / sum(n),
         cum_pct = cumsum(pct)) |>
  gt() |>
  fmt_number(columns = c(pct, cum_pct), decimals = 1)
```

2.8.6 Output & Results

```
  I  II III  IV
 53  87  77  33

    I    II   III    IV
 0.21  0.35  0.31  0.13

    I    II   III    IV
 0.21  0.56  0.87  1.00
```

For the ordered stages, the cumulative column shows that 87% of patients are stage III or earlier.

2.8.7 Interpretation

Report frequency tables for every categorical variable in a manuscript's Table 1. Include both counts and percentages: "Stage III: 77 (30.8%)". For ordered variables, the cumulative column often tells the story more clearly than the raw counts.

2.8.8 Practical Tips

- Preserve factor level ordering when reporting; ordered factors in R maintain the sequence automatically.
- Report the denominator used for percentages (total sample or column total) to avoid confusion.
- For sparse levels, consider collapsing categories; report both the original and collapsed table.
- Missing values: `table()` excludes NAs by default; use `useNA="always"` to include them.
- Publication tables benefit from `gtsummary::tbl_summary()` which handles both frequencies and continuous variables.

2.8.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.8.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.9 The Geometric Mean

2.9.1 Introduction

The geometric mean is the arithmetic mean on the log scale, back-transformed to the original scale. It is the appropriate summary for variables that combine multiplicatively rather than additively: antibody titres, concentrations, growth rates, investment returns. For log-normal data, the geometric mean is the median and is thus a natural measure of typicality.

2.9.2 Prerequisites

The reader should know the arithmetic mean and be comfortable with natural logarithms.

2.9.3 Theory

For positive values $x_1, \ldots, x_n$:

$$\mathrm{GM}(x) = \Big(\prod x_i\Big)^{1/n} = \exp\!\Big(\frac{1}{n}\sum \log x_i\Big).$$

The second form is numerically stable and matches how the geometric mean is computed in practice. Key properties:

- $\mathrm{GM}(x) \le \bar{x}$, with equality only when all x_i are equal (AM-GM inequality, a consequence of Jensen).
- For log-normal data, the geometric mean equals the population median.
- The ratio of the geometric mean to the arithmetic mean is a simple measure of skewness.

2.9.4 Assumptions

- All values must be strictly positive; a single zero or negative value renders the geometric mean undefined.

- For inference (confidence intervals), log-normality of the data is typically assumed.

2.9.5 R Implementation

```r
library(psych)

set.seed(2026)
titres <- 2^sample(4:12, 50, replace = TRUE)  # antibody titres in 2^n
 ↪  form

mean(titres)
median(titres)
psych::geometric.mean(titres)
exp(mean(log(titres)))

# 95% CI for the geometric mean
n <- length(titres)
log_x <- log(titres)
log_mean <- mean(log_x)
log_se   <- sd(log_x) / sqrt(n)
exp(log_mean + c(-1, 1) * qt(0.975, n - 1) * log_se)
```

2.9.6 Output & Results

```
[1] 1157         # arithmetic mean
[1] 512          # median
[1] 588          # geometric mean
[1] 588          # (equivalent computation)

[1] 441 782      # 95% CI for GM
```

The geometric mean (588) sits close to the median (512) and well below the arithmetic mean (1157). This is typical of log-normal or multiplicative data: the arithmetic mean is pulled up by the upper tail.

2.9.7 Interpretation

Report the geometric mean for titres, viral loads, drug concentrations, and any other multiplicatively combining variable. In a paper: "the geometric mean antibody titre was 588 (95% CI 441-782)".

2.9.8 Practical Tips

- Non-positive values require imputation or exclusion; adding a small constant (e.g., 0.5) before log is common but distorts small values.

- The geometric mean of ratios is the exponent of the mean of log-ratios; reporting log-scale effects is cleaner.
- For growth rates, the geometric mean equals the arithmetic mean of $(1+r_i)$ values minus 1.
- In meta-analysis of ratios, log transformation + normal approximation is the standard pipeline.
- When data are centred around 1 (e.g., hazard ratios), the difference between AM and GM is small.

2.9.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.9.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.10 The Harmonic Mean

2.10.1 Introduction

The harmonic mean is the reciprocal of the arithmetic mean of reciprocals. It is appropriate when averaging rates, speeds, or ratios where the denominator is fixed – for example, when a traveller covers equal distances at different speeds and you want the average speed. In machine learning, the harmonic mean of precision and recall gives the F1 score.

2.10.2 Prerequisites

The reader should know the arithmetic mean and be comfortable taking reciprocals in R.

2.10.3 Theory

For positive values $x_1, \ldots, x_n$,

$$\mathrm{HM}(x) = \frac{n}{\sum 1/x_i}.$$

The harmonic mean is always less than or equal to the geometric mean, which is less than or equal to the arithmetic mean: HM $\leq$ GM $\leq$ AM, with equality when all x_i are equal.

The F1 score in binary classification is the harmonic mean of precision and recall:

$$F_1 = \frac{2 \cdot P \cdot R}{P + R} = \frac{1}{\frac{1}{2}\left(\frac{1}{P} + \frac{1}{R}\right)} \cdot (\text{rescaled}).$$

2.10.4 Assumptions

All values must be strictly positive. Negative values or zeros make the harmonic mean undefined or infinite.

2.10.5 R Implementation

```r
library(psych)

speeds <- c(40, 60, 80)  # km/h over three equal distances
psych::harmonic.mean(speeds)
mean(speeds)                  # incorrect for average speed over equal
 ↪  distances

# F1 score
precision <- 0.80
recall    <- 0.60
f1 <- 2 * precision * recall / (precision + recall)
f1

# Harmonic mean is appropriate for average rates
n <- 3
hm_speed <- n / sum(1 / speeds)
hm_speed
```

2.10.6 Output & Results

```
[1] 55.38       # harmonic mean of speeds
[1] 60          # arithmetic mean (incorrect for equal-distance travel)
[1] 0.686       # F1 score with P=0.80, R=0.60
```

Travelling equal distances at 40, 60, 80 km/h gives an average speed of 55.4 km/h, not 60 km/h. The F1 of 0.69 splits the difference between P and R, penalising the lower value more than the arithmetic mean would.

2.10.7 Interpretation

Use the harmonic mean when averaging rates with a fixed numerator or denominator (e.g., average speed when distance is constant). Use the F1 form when balancing two normalised scores where both values matter.

2.10.8 Practical Tips

- Zero or negative inputs break the harmonic mean; guard against them.
- The harmonic mean weights small values more heavily; a single small observation drives the result down.
- In meta-analysis of rates per unit time, harmonic-mean sample size is sometimes used in effective-sample-size calculations.
- The F-beta score generalises F1: $F_\beta = (1+\beta^2)PR/(\beta^2 P + R)$, with $\beta = 2$ emphasising recall and $\beta = 0.5$ emphasising precision.
- Reporting a harmonic mean without explanation is rare; specify why it is the appropriate summary in the methods.

2.10.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.10.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.11 The Interquartile Range

2.11.1 Introduction

The interquartile range (IQR) is the distance between the first and third quartiles, $Q_3 - Q_1$. It measures the width of the central 50% of the distribution and is robust to outliers, since extreme values do not affect the quartiles themselves. The IQR appears in boxplots, Tukey fences, and the median-IQR convention for reporting skewed biomedical variables.

2.11.2 Prerequisites

The reader should know what quantiles and quartiles are, and be comfortable with R's `quantile()` function.

2.11.3 Theory

For a sample, Q_1 and Q_3 are the 0.25 and 0.75 quantiles of the empirical CDF. The IQR is

$$\text{IQR} = Q_3 - Q_1.$$

Tukey's hinges differ slightly from R's default quantiles; `fivenum()` returns the hinges and `IQR()` uses `quantile()` with `type=7`.

For a normal distribution with SD σ, the IQR equals $2 \cdot 0.6745 \cdot \sigma \approx 1.349\sigma$. Equivalently, dividing the IQR by 1.349 gives a rough, robust estimate of the SD.

The **Tukey fences** at $Q_1 - 1.5 \cdot \text{IQR}$ and $Q_3 + 1.5 \cdot \text{IQR}$ define the whiskers of a standard boxplot; points beyond are flagged as outliers.

2.11.4 Assumptions

The IQR is purely descriptive and requires no distributional assumption.

2.11.5 R Implementation

```r
set.seed(2026)
x <- rnorm(80, mean = 75, sd = 12)

quantile(x, probs = c(0.25, 0.75))
IQR(x)

# Tukey fences
q1 <- quantile(x, 0.25); q3 <- quantile(x, 0.75)
fences <- c(lower = q1 - 1.5 * IQR(x),
            upper = q3 + 1.5 * IQR(x))
fences
sum(x < fences[1] | x > fences[2])

# Robust SD from IQR (approximately)
IQR(x) / 1.349
sd(x)
```

2.11.6 Output & Results

```
   25%      75%
 66.4     83.1

[1] 16.7       # IQR

lower  upper
 41.4  108.1

[1] 12.4       # SD from IQR (robust)
[1] 11.9       # sample SD
```

The two SD estimates agree closely for approximately normal data; divergence signals heavy tails or outliers.

2.11.7 Interpretation

Report the IQR alongside the median for skewed data. A typical biomedical manuscript uses "median 6.8 days (IQR 4.2-10.1 days)" to summarise length-of-stay or similar variables.

2.11.8 Practical Tips

- The boxplot visually displays the IQR as the box; whiskers extend to the Tukey fences or to the nearest non-outlier data point.
- For very small samples ($n < 10$), the IQR is unstable; report the order statistics directly.
- The IQR is the scale estimator implicit in the Tukey boxplot; it is one of the most interpretable dispersion measures for audiences less familiar with SDs.
- In grouped summaries, compare IQRs across groups alongside medians; equal medians with different IQRs indicate different spreads at the same centre.
- Do not confuse IQR (dispersion) with inter-quartile ratio (Q_3/Q_1) occasionally used in log-skewed contexts.

2.11.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.11.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.12 Kurtosis

2.12.1 Introduction

Kurtosis is the fourth standardised moment of a distribution and quantifies tail heaviness relative to the normal. High kurtosis means a few observations contribute disproportionately to the variance (the "fat tails" of finance). Low kurtosis means the distribution has fewer extreme values than the normal. Kurtosis matters because many estimators and tests assume normal-like tails; heavy tails break asymptotic approximations and inflate Type I error.

2.12.2 Prerequisites

The reader should know what mean, SD, and skewness are, and should be comfortable computing summary statistics in R.

2.12.3 Theory

The population kurtosis is

$$\gamma_2 = E\left[\left(\frac{X-\mu}{\sigma}\right)^4\right].$$

For a standard normal distribution, $\gamma_2 = 3$. The convention is to report **excess kurtosis** $\gamma_2 - 3$, where 0 indicates normal-like tails, positive indicates heavier tails (leptokurtic), and negative indicates lighter tails (platykurtic).

Classical values:

- Normal: excess kurtosis = 0.
- Uniform: excess kurtosis = -1.2 (light tails).
- Logistic: excess kurtosis = 1.2.
- t-distribution with ν df: excess kurtosis = $6/(\nu-4)$ for $\nu > 4$. At $\nu = 5$ the excess kurtosis is 6; at $\nu = 10$ it is 1.
- Exponential: excess kurtosis = 6.
- Laplace: excess kurtosis = 3.

2.12.4 Assumptions

Kurtosis as a descriptive statistic has no assumption. As an estimator of a population parameter, it requires finite fourth moments.

2.12.5 R Implementation

```
library(moments)

set.seed(2026)
norm_sample <- rnorm(5000)
t4_sample   <- rt(5000, df = 4)
unif_sample <- runif(5000)

c(normal  = kurtosis(norm_sample) - 3,
  t_df4   = kurtosis(t4_sample) - 3,
  uniform = kurtosis(unif_sample) - 3)

moments::anscombe.test(t4_sample)
```

2.12.6 Output & Results

```
  normal    t_df4  uniform
   -0.04     6.12    -1.21
```

The t-distribution with 4 df has massively heavier tails than normal, as predicted. The uniform distribution has lighter tails. Anscombe's test on the t-sample gives $p < .001$, formally rejecting normal kurtosis.

2.12.7 Interpretation

Report excess kurtosis alongside skewness whenever distribution shape matters. A value near zero is "normal-like"; a value above 1 is a heavy-tailed distribution for which outlier-sensitive methods (mean, SD, OLS) may perform poorly.

2.12.8 Practical Tips

- Kurtosis is sensitive to outliers; a single extreme value can inflate it dramatically.
- Formal normality tests (Shapiro-Wilk, Jarque-Bera) implicitly test kurtosis and skewness jointly.
- Heavy tails call for robust methods: median, MAD, robust regression, rank-based tests.
- Leptokurtic distributions sometimes motivate using a Student-t rather than normal likelihood in regression (see `brms::student`).
- Different software sometimes report kurtosis with/without the "-3" correction; always check whether excess or raw kurtosis is being reported.

2.12.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.12.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.13 Mean Absolute Deviation

2.13.1 Introduction

The mean absolute deviation (MAD) is the average distance of observations from a measure of central tendency. It uses absolute values instead of squares, making

it less sensitive to outliers than the standard deviation. In the modern literature, the term "MAD" almost always means the **median absolute deviation from the median**, a robust scale estimator.

2.13.2 Prerequisites

The reader should know what a mean, median, and SD are, and how to compute them in R.

2.13.3 Theory

Two versions exist.

MAD about the mean: $\text{MAD}_{\text{mean}} = \frac{1}{n}\sum |x_i - \bar{x}|$. More stable than the SD under heavy-tailed distributions but still sensitive to outliers because it uses the mean as the centre.

MAD about the median (modern default): $\text{MAD} = \text{median}|x_i - \text{median}(x)|$. Highly robust; breakdown point of 50%. When multiplied by 1.4826, it is a consistent estimator of the SD for normal data.

R's `mad()` function computes MAD about the median with the 1.4826 scaling constant by default.

2.13.4 Assumptions

MAD is purely descriptive; no distributional assumption.

2.13.5 R Implementation

```r
set.seed(2026)
x <- c(rnorm(50, mean = 100, sd = 10), 500)  # one outlier

sd(x)
mad(x, constant = 1.4826)
mad(x, constant = 1)
mean(abs(x - mean(x)))
```

2.13.6 Output & Results

```
[1] 56.4      # SD inflated by outlier
[1] 9.78      # MAD estimator of SD (robust)
[1] 6.60      # raw MAD (unscaled)
[1] 22.4      # MAD about the mean (less robust)
```

The classical SD is nearly six times the robust MAD estimator because a single extreme point at 500 dominates the sum of squares. The MAD about the median is essentially unaffected.

2.13.7 Interpretation

Report MAD when the data contain outliers or when robustness to contamination is desirable. In biomedical chemistry, MAD is the default measure of analytical imprecision for certain platforms.

2.13.8 Practical Tips

- `mad()` defaults: `constant=1.4826`, `center=median`. Always check whether this is what you want.
- For a normal sample, MAD $\times$ 1.4826 approximates the SD; the two agree closely at moderate n.
- The MAD is a robust estimator but not robust to the estimator of the *centre*: use the median, not the mean, as the centre.
- For positive-skewed data (lab values), a raw MAD on the original scale can still be informative; log-transform the data first if the skew is severe.
- Do not confuse the MAD with the variance; they are in the same units as the data (unlike the variance, which is squared units).

2.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.13.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.14 Measures of Dispersion

2.14.1 Introduction

A measure of central tendency tells you where the data are centred; a measure of dispersion tells you how far they spread around that centre. Two samples can share the same mean yet behave differently in every downstream calculation because their variability is different. This tutorial covers the five dispersion measures used routinely in applied work, explains when each is appropriate, and shows how to compute them in R.

2.14.2 Prerequisites

The reader should know what the mean and median are, and should be comfortable computing them on an R vector.

2.14.3 Theory

For a sample $x_1, \ldots, x_n$:

- **Range** $= \max x_i - \min x_i$. Simplest, most fragile: uses only two values, is dominated by outliers, has no information about the bulk of the distribution.
- **Variance** $s^2 = \frac{1}{n-1}\sum(x_i - \bar{x})^2$. The average squared deviation from the mean. Additive over independent sums. Divisor $n-1$ gives an unbiased estimator of the population variance; divisor n gives the MLE under normality (biased).
- **Standard deviation** $s = \sqrt{s^2}$. Same units as the data. The SD of a normally distributed variable corresponds to the 68-95-99.7 rule.
- **Median absolute deviation (MAD)** $= \text{median}|x_i - \text{median}(x)|$. Robust to outliers; breakdown point of 50%. Multiplied by 1.4826 to estimate the SD of a normal distribution.
- **Interquartile range (IQR)** $= Q_3 - Q_1$. The width of the central 50% of the data. Used in the Tukey boxplot fence for outlier identification ($\pm 1.5 \cdot \text{IQR}$).

The SD is the default for approximately normal data. For skewed or contaminated data, the MAD and IQR are far more stable. Reporting the range is informative only for small, clean samples or for summarising audit ranges.

2.14.4 Assumptions

All dispersion measures are purely descriptive; they do not require any distributional assumption. Their *interpretation*, however, depends on the distribution: an SD of 10 on a normal distribution implies 68% of values within ± 10 of the mean, while an SD of 10 on a heavy-tailed distribution does not.

2.14.5 R Implementation

```
library(dplyr)
library(robustbase)

set.seed(2026)

df <- tibble::tibble(
  symmetric    = rnorm(200, mean = 50, sd = 10),
  right_skewed = rlnorm(200, meanlog = log(50), sdlog = 0.6)
)

summarise_dispersion <- function(x) {
  tibble::tibble(
    n        = length(x),
    range    = max(x) - min(x),
```

```
    variance = var(x),
    sd       = sd(x),
    iqr      = IQR(x),
    mad      = mad(x, constant = 1.4826),
    mad_raw  = mad(x, constant = 1)
  )
}

dispersion_tbl <- df |>
  tidyr::pivot_longer(everything(), names_to = "variable", values_to =
↪  "value") |>
  group_by(variable) |>
  summarise(
    summarise_dispersion(value),
    .groups = "drop"
  )

dispersion_tbl

Qn_scale <- robustbase::Qn(df$right_skewed)
Sn_scale <- robustbase::Sn(df$right_skewed)
c(Qn = Qn_scale, Sn = Sn_scale)
```

The `mad()` function defaults to a scaling constant of 1.4826, which makes it consistent with the SD for normal data. `robustbase::Qn()` and `Sn()` are alternative robust scale estimators with higher efficiency than MAD.

2.14.6 Output & Results

Typical output:

variable	n	range	variance	sd	iqr	mad
symmetric	200	57.3	98.5	9.93	13.2	10.1
right_skewed	200	260.2	1143.2	33.8	34.5	23.7

For the symmetric normal sample, SD and MAD are nearly identical as expected under normality. For the right-skewed log-normal sample, the SD (33.8) is larger than the MAD (23.7), because the SD is pulled up by the right tail.

2.14.7 Interpretation

Report dispersion appropriate to the distribution shape:

- **Approximately normal data**: mean (SD).
- **Skewed or contaminated data**: median (IQR) or median (MAD).

- Never report mean (SD) for a distribution whose histogram is obviously asymmetric; the summary gives a misleading impression of typicality.

2.14.8 Practical Tips

- Plot the histogram before choosing the summary; numerical dispersion alone can disguise bimodality or heavy tails.
- For small samples, IQR is preferred over range: a single extreme value dominates the range.
- The SD and variance are measured in the original units squared (variance) or original units (SD); do not mix them in a single summary.
- When reporting a coefficient of variation (CV = SD/mean), ensure the mean is far from zero; CV is meaningless for variables centred near zero.
- In R, `sd()` uses the $n-1$ divisor; the MLE version is `sqrt(mean((x-mean(x))^2))`.

2.14.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.14.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.15 Measures of Central Tendency

2.15.1 Introduction

A measure of central tendency is a single number that summarises "where the data are". The three classical choices – arithmetic mean, median, and mode – are not interchangeable: each answers a slightly different question, and each is appropriate in a different context. Choosing the right one matters because readers of your results will form mental pictures of the data that the number implies.

2.15.2 Prerequisites

The reader should know the difference between a numeric, an ordered categorical, and a nominal categorical variable, and should be able to read a histogram. Familiarity with R vectors is assumed.

2.15.3 Theory

The **arithmetic mean** of observations $x_1, \ldots, x_n$ is $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$. It is the centre of mass of the data and minimises the sum of squared deviations from any candidate centre. The mean uses all observations, but it is sensitive to outliers: a single extreme value can drag it arbitrarily far from the bulk of the data.

The **median** is the value that separates the lower half of the data from the upper half: $x_{((n+1)/2)}$ for odd n, the average of the two middle values for even n. The median minimises the sum of absolute deviations. Unlike the mean, it is robust to outliers: moving the largest observation to infinity leaves the median unchanged.

The **mode** is the most frequently occurring value. For continuous data, the mode is typically defined from a kernel density estimate rather than from the raw values, because ties are unlikely. The mode is the only measure of central tendency that makes sense for nominal categorical variables.

The **trimmed mean** computes the mean after removing a proportion α of the smallest and largest observations – a compromise between the efficiency of the mean and the robustness of the median. A 10% trimmed mean removes the bottom and top 10% of the observations; a 50% trimmed mean is the median.

2.15.4 Assumptions

These are descriptive statistics, not tests, so they do not carry distributional assumptions. They do carry interpretive assumptions: the mean is most useful when the distribution is roughly symmetric; the median is preferred for skewed distributions and for ordinal data; the mode is preferred for categorical data and for multimodal distributions where a single centre is misleading.

2.15.5 R Implementation

```r
library(DescTools)
library(psych)

x <- c(2.1, 2.4, 2.5, 2.7, 2.9, 3.1, 3.3, 3.4, 3.7, 42.0)

mean(x)
median(x)
mean(x, trim = 0.1)
Mode(x)
psych::geometric.mean(x)
psych::harmonic.mean(x)
```

The vector `x` is a small sample with a single outlier at 42. Running the above shows a mean of 6.71, a median of 3.00, and a 10% trimmed mean of 2.94. The

geometric mean (3.44) and harmonic mean (3.15) are close to the median rather than the mean, since both give less weight to the outlier.

2.15.6 Output & Results

For a skewed or contaminated distribution, the three classical measures of central tendency can differ substantially:

Statistic	Value
Mean	6.71
Median	3.00
10% trimmed mean	2.94
Geometric mean	3.44
Harmonic mean	3.15

The mean is pulled far from the bulk of the data by the single outlier; every other measure stays close to the main cluster.

2.15.7 Interpretation

For a manuscript, the choice of statistic should be justified in the methods section. A paper reporting the "mean serum creatinine" in a cohort where several dialysis patients are included should either exclude those patients explicitly or report the median with IQR. Never mix conventions across papers in a series: if you reported medians in one table you should not swap to means in the next.

2.15.8 Practical Tips

- Plot the data before choosing the statistic. A histogram or boxplot will show immediately whether the mean is a sensible summary.
- Report the median and IQR (not the mean and SD) for variables with strong right skew: lab values, income, incubation times.
- For small samples ($n < 10$), any central tendency measure is uncertain; consider bootstrapping to get a confidence interval for the statistic of choice.
- Do not report the mode for continuous data unless it comes from a density estimate – the sample mode of continuous data is the lowest value in the tied set, which is meaningless.

2.15.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.15.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.16 Measures of Shape

2.16.1 Introduction

Central tendency and dispersion alone do not describe a distribution completely: two samples with the same mean and SD can look very different if one is symmetric and the other skewed, or if one has thin tails and the other heavy tails. Skewness and kurtosis are the conventional third- and fourth-moment summaries that capture these shape features.

2.16.2 Prerequisites

The reader should know what the mean, variance, and standard deviation are, and should understand what a histogram shows.

2.16.3 Theory

The **skewness** of a distribution measures its asymmetry:

$$g_1 = \frac{1}{n} \sum \left(\frac{x_i - \bar{x}}{s} \right)^3 .$$

Positive skewness indicates a right-tailed distribution (mean > median); negative skewness indicates a left tail. For a normal distribution, $g_1 = 0$. Absolute values greater than about 1 are considered substantial.

The **kurtosis** measures tail heaviness:

$$g_2 = \frac{1}{n} \sum \left(\frac{x_i - \bar{x}}{s} \right)^4 .$$

For a normal, $g_2 = 3$. By convention, **excess kurtosis** is reported as $g_2 - 3$: zero means normal-like tails, positive means heavier (leptokurtic), negative means lighter (platykurtic). High kurtosis signals that a few observations contribute a large share of the variance; in finance this is the "fat tails" that break many standard models.

Both statistics are moment-based and therefore sensitive to outliers. In small samples, their estimates are noisy.

2.16.4 Assumptions

Skewness and kurtosis are descriptive and require no distributional assumption. Their asymptotic standard errors assume independent observations with finite sixth moment for skewness and eighth for kurtosis.

2.16.5 R Implementation

```
library(moments)
library(psych)

set.seed(2026)

symmetric   <- rnorm(5000, mean = 0, sd = 1)
right_skewed <- rlnorm(5000, meanlog = 0, sdlog = 0.6)
heavy_tailed <- rt(5000, df = 4)

shape_summary <- function(x, label) {
  data.frame(
    variable = label,
    skewness = moments::skewness(x),
    kurtosis = moments::kurtosis(x),
    excess_kurtosis = moments::kurtosis(x) - 3
  )
}

rbind(
  shape_summary(symmetric,    "Normal"),
  shape_summary(right_skewed, "Log-normal"),
  shape_summary(heavy_tailed, "t, df=4")
)

psych::describe(data.frame(symmetric, right_skewed, heavy_tailed),
                skew = TRUE, ranges = FALSE)
```

2.16.6 Output & Results

variable	skewness	kurtosis	excess kurtosis
Normal	0.01	3.02	0.02
Log-normal	1.95	9.96	6.96
t, df=4	-0.02	7.41	4.41

The normal sample has skewness and excess kurtosis near zero. The log-normal

is strongly right-skewed and heavy-tailed. The t-distribution with 4 df is symmetric but very heavy-tailed.

2.16.7 Interpretation

Report shape statistics alongside location and dispersion for any distribution whose histogram is not obviously normal. In a methods section: "Serum CRP was right-skewed (skewness = 1.8, excess kurtosis = 3.2), so log-transformation was applied before parametric testing."

2.16.8 Practical Tips

- Skewness and kurtosis are sensitive to outliers; a single extreme point can drive g_1 past 1.
- In small samples ($n < 30$), estimates are noisy; do not over-interpret small deviations from zero.
- For normality assessment, prefer graphical methods (Q-Q plots) over numerical skewness and kurtosis thresholds.
- Different software use slightly different bias corrections; `moments::skewness` uses the g_1 form, `psych::describe` uses the G_1 form with a small-sample correction.
- Log transformation often reduces both skewness and excess kurtosis in right-skewed biomedical data.

2.16.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.16.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.17 Outlier Detection Rules

2.17.1 Introduction

Outlier detection is a diagnostic step, not a cleaning step: flagged points should be investigated, not automatically removed. Several rules produce flags from different premises; each has a different trade-off between sensitivity and robustness. This page covers the common univariate rules and their assumptions.

2.17.2 Prerequisites

The reader should know what mean, median, SD, and IQR are.

2.17.3 Theory

- **1.5 x IQR fence (Tukey)**: observations outside $[Q_1 - 1.5 \cdot \text{IQR},\ Q_3 + 1.5 \cdot \text{IQR}]$ are mild outliers; $3 \cdot \text{IQR}$ marks extremes. Robust and distribution-free.
- **Z-score (3-sigma)**: $|z_i| > 3$ where $z_i = (x_i - \bar{x})/s$. Sensitive to the outliers it is trying to flag (masking).
- **Hampel identifier**: $|x_i - \text{median}| > 3 \cdot \text{MAD}$. Robust variant of the z-score that uses median and MAD.
- **Grubbs' test**: formal test of the most extreme value under normality. Gives a p-value but is sensitive to its own normality assumption.

Masking and swamping are two pitfalls: masking occurs when true outliers inflate the SD and hide themselves; swamping occurs when one outlier drags neighbours across the threshold, falsely flagging them.

2.17.4 Assumptions

- IQR fence and Hampel: no distributional assumption; robust.
- Z-score: approximately normal distribution.
- Grubbs' test: normality and a single suspected outlier.

2.17.5 R Implementation

```r
library(outliers)

set.seed(2026)
x <- c(rnorm(48, mean = 50, sd = 8), 95, 110)

# IQR fence
q <- quantile(x, c(0.25, 0.75))
iqr <- IQR(x)
fences <- c(q[1] - 1.5 * iqr, q[2] + 1.5 * iqr)
which(x < fences[1] | x > fences[2])

# Z-score
z <- (x - mean(x)) / sd(x)
which(abs(z) > 3)

# Hampel
h <- (x - median(x)) / mad(x)
which(abs(h) > 3)
```

```
# Grubbs' test for the single most extreme
grubbs.test(x)
```

2.17.6 Output & Results

```
IQR fence flags:   [49, 50]
z-score flags:     [50]
Hampel flags:      [49, 50]
Grubbs: G = 2.81, p = 0.03
```

The IQR and Hampel rules identify both extreme values; the z-score, inflated by those same values, identifies only the most extreme. Grubbs' test rejects the null that the extreme point is from the same distribution.

2.17.7 Interpretation

Report which rule you applied and how many observations it flagged. "Two observations exceeded the Tukey 1.5 x IQR upper fence; both were verified as accurate data-entry values and retained in the primary analysis" is a defensible sentence.

2.17.8 Practical Tips

- Flag, investigate, decide. Never delete silently.
- For skewed data, IQR fences still flag normal-range observations as outliers; consider log-transformation first.
- For multivariate outliers, use Mahalanobis distance; for regression diagnostics, use Cook's distance.
- Grubbs' test assumes normality; for non-normal data prefer the IQR or Hampel rule.
- Report sensitivity analyses that re-run the primary analysis with and without flagged observations; the conclusion should be robust.

2.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.17.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.18 Percentile Ranks

2.18.1 Introduction

A percentile rank answers "what fraction of observations fall at or below this value?" It converts an absolute measurement to a relative position in the sample distribution. In paediatrics, growth charts give a child's weight percentile; in cognitive testing, an IQ score comes with a percentile rank; in bioinformatics, a gene's expression is often reported as a percentile across a reference panel.

2.18.2 Prerequisites

The reader should know what a quantile is and be comfortable with sorted vectors in R.

2.18.3 Theory

For a sample $x_1, \ldots, x_n$, the percentile rank of a value y is

$$\mathrm{PR}(y) = 100 \cdot \frac{\#\{x_i \le y\}}{n}.$$

Several refinements exist: the "mid-rank" form counts half of any ties; the "plotting position" form uses $(i - a)/(n + 1 - 2a)$ for various a (e.g., Blom's $a = 3/8$).

For a single value, computing the percentile rank is the inverse of computing a quantile: if the 75th percentile is $q_{0.75}$, then $\mathrm{PR}(q_{0.75}) \approx 75$.

2.18.4 Assumptions

Purely descriptive; assumes only that the reference distribution is appropriate to the value being ranked.

2.18.5 R Implementation

```
library(dplyr)

set.seed(2026)
growth_chart <- rnorm(1000, mean = 15.0, sd = 2.2)   # reference weights
 ↪  (kg)
patient      <- 17.8                                 # a patient's weight

pr_patient <- 100 * mean(growth_chart <= patient)
pr_patient

# Percentile ranks for all observations in a cohort
```

```
cohort <- data.frame(weight = rnorm(30, 15.5, 2.0))
cohort$rank    <- rank(cohort$weight)
cohort$pr      <- 100 * cohort$rank / nrow(cohort)
cohort$pr_mid  <- 100 * (rank(cohort$weight) - 0.5) / nrow(cohort)
head(cohort)

# ECDF equivalent
ec <- ecdf(growth_chart)
100 * ec(patient)
```

2.18.6 Output & Results

```
[1] 89.4     # 17.8 kg is at the 89th percentile of the reference
```

```
  weight rank    pr pr_mid
1  13.29    1   3.3   1.67
2  14.15    6  20.0  18.33
3  15.01   11  36.7  35.00
```

2.18.7 Interpretation

Report percentile ranks in growth, development, and norming contexts. In a clinical note: “body weight 17.8 kg (89th percentile for age/sex)”. The percentile tells the reader how unusual the value is relative to the reference.

2.18.8 Practical Tips

- Percentile ranks depend on the reference sample; always state which reference was used (national norms, internal cohort, etc.).
- Ties are handled by `rank()` via averaging, minimum, or maximum; specify `ties.method` when the convention matters.
- For small reference samples, percentile ranks are unstable; use smoothing (LMS method) for growth chart applications.
- A percentile rank of exactly 100 is usually not reported; “above the 99th percentile” is the conventional phrasing for the sample maximum.
- Do not confuse percentile rank (position in a distribution) with percentile (value at a specified position).

2.18.9 For Reviewers

Use the shared reviewer checklist, together with this method’s assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript’s application of the method.

2.18.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.19 Quantiles and Percentiles

2.19.1 Introduction

A quantile is a cut-point that divides a distribution by cumulative probability: the p-quantile is the value below which a proportion p of the data lies. The median is the 0.5-quantile; quartiles are the 0.25, 0.50, and 0.75 quantiles; percentiles are quantiles rescaled to the 0-100 range. Quantiles underlie boxplots, five-number summaries, reference ranges in lab medicine, and non-parametric confidence intervals.

2.19.2 Prerequisites

The reader should know what a cumulative distribution function is and be comfortable with sorted vectors in R.

2.19.3 Theory

For a sample $x_1, \ldots, x_n$ sorted as $x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n)}$, the sample p-quantile is not uniquely defined when $p \cdot (n+1)$ is not an integer. R offers nine definitions, selected via the `type` argument of `quantile()`:

- `type=1`: inverse of the empirical CDF, giving a step-function quantile.
- `type=4`: linear interpolation of the empirical CDF (SAS default).
- `type=6`: linear interpolation based on plotting positions (Minitab, SPSS).
- `type=7`: R's default; linear interpolation between order statistics with no offset.
- `type=8`: approximately unbiased for continuous distributions (Hyndman-Fan recommendation).

The different types give slightly different numeric values in small samples; they converge as n grows. Type 7 is the default in R's `quantile()`, but reproducibility across software requires specifying the type.

2.19.4 Assumptions

Quantiles are purely descriptive; they require no distributional assumption.

2.19.5 R Implementation

```r
library(dplyr)

set.seed(2026)
```

```r
x <- rnorm(60, mean = 75, sd = 10)

# Default quantile types
quantile(x, probs = c(0.25, 0.50, 0.75))

# Compare five R types
sapply(c(1, 4, 6, 7, 8), function(t) quantile(x, probs = 0.25, type = t))

# Five-number summary via fivenum
fivenum(x)

# Deciles (10th, 20th, ..., 90th percentiles)
quantile(x, probs = seq(0.1, 0.9, by = 0.1))

# Quantiles by group
diabetes <- tibble::tibble(
  arm = factor(rep(c("Placebo", "Active"), each = 40)),
  hba1c = c(rnorm(40, 7.4, 0.6), rnorm(40, 6.8, 0.7))
)
diabetes |> group_by(arm) |>
  summarise(q25 = quantile(hba1c, 0.25),
            q50 = quantile(hba1c, 0.50),
            q75 = quantile(hba1c, 0.75))
```

2.19.6 Output & Results

```
25% 50% 75%
71.2 74.8 80.6

   1    4    6    7    8
72.3 71.8 71.0 71.2 71.3
```

The five types differ by less than 2 units at $n = 60$. For `diabetes`, the active arm's quartiles are shifted lower than placebo's, as expected.

2.19.7 Interpretation

Report quantiles when the distribution is skewed, ordinal, or otherwise not well-summarised by a mean: median with IQR (25th and 75th percentiles) is the standard form. In reference-range contexts (clinical chemistry), the 2.5 and 97.5 percentiles define the central 95% "normal range".

2.19.8 Practical Tips

- Always specify `type` in `quantile()` when reproducing results across software.

- The boxplot uses Type 2 (`fivenum()`) for its hinges; this differs from Type 7 quantiles.
- Quantiles respect monotonic transformations: `quantile(log(x),0.5)==log(quantile(x,0.5))` (exactly for the median, approximately for others depending on `type`).
- For very small samples ($n < 10$), quantile estimates are unstable; report the order statistics directly instead.
- Empirical quantiles are biased estimators of population quantiles; use Harrell-Davis (`Hmisc::hdquantile`) for a smoother estimator.

2.19.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.19.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.20 Robust Statistics: An Overview

2.20.1 Introduction

A robust statistic is one that remains useful when the data deviate from the idealised model – heavy tails, outliers, skewed contamination. Classical estimators like the mean and SD are efficient under normality but break down under even modest contamination. Robust alternatives sacrifice a little efficiency for dramatic stability, and are the default in many applied fields where data quality cannot be guaranteed.

2.20.2 Prerequisites

The reader should know the mean, median, SD, and MAD. Familiarity with outlier concepts helps.

2.20.3 Theory

Two key concepts from robustness theory:

Breakdown point. The largest fraction of contaminated observations an estimator can tolerate before giving an arbitrary answer. The arithmetic mean has breakdown 0: a single extreme point can push it arbitrarily far. The median has

breakdown 0.5: up to half the sample can be contaminated before the median is affected.

Influence function. Measures the infinitesimal effect of adding a single observation at x on the estimator. The arithmetic mean has a linear, unbounded influence function; the median has a bounded (step-shaped) influence function.

Robust estimators aim for high breakdown point and bounded influence function, typically with moderate loss of efficiency at the normal model.

Robust alternatives to classical summaries:

Classical	Robust	Breakdown	Efficiency at normal
Mean	Median	0.5	64%
Mean	20% trimmed mean	0.2	~95%
Mean	Huber M-estimator	0	~95%
SD	MAD	0.5	37%
SD	Qn, Sn	0.5	~82%
OLS	Robust regression (`rlm`, `lmrob`)	0 to 0.5	95%

2.20.4 Assumptions

Robustness is an explicit relaxation of distributional assumptions; it asks how the estimator behaves under deviation, not whether deviation is present.

2.20.5 R Implementation

```r
library(robustbase)
library(MASS)

set.seed(2026)
clean <- rnorm(100, mean = 50, sd = 10)
contaminated <- c(clean[1:95], rnorm(5, mean = 200, sd = 5))

classical <- c(mean = mean(contaminated), sd = sd(contaminated))
robust    <- c(median = median(contaminated),
               mad    = mad(contaminated),
               Qn     = Qn(contaminated),
               Sn     = Sn(contaminated))

rbind(classical = c(classical, NA, NA), robust)

# Robust regression example
n <- 80
x <- rnorm(n)
y <- 2 + 1.5 * x + rnorm(n)
```

```r
y[c(5, 10, 15)] <- y[c(5, 10, 15)] + c(15, -15, 20)  # inject outliers

coef(lm(y ~ x))
coef(rlm(y ~ x))          # M-estimator
coef(lmrob(y ~ x))        # MM-estimator with high breakdown
```

2.20.6 Output & Results

```
      mean    sd   median  mad    Qn    Sn
 57.84   32.9   50.28  10.21  10.45 10.72
```

Classical mean is pulled to 57.8 by the five extreme values; every robust estimator remains near the true 50.

For the regression: OLS slope is 1.23 (pulled by outliers); `rlm` gives 1.47 (close to truth of 1.5); `lmrob` gives 1.48.

2.20.7 Interpretation

Use robust statistics as the default in biomedical work where outliers or heavy tails are typical. A robust summary plus a classical one can be reported side by side; divergence between them signals contamination that warrants investigation.

2.20.8 Practical Tips

- MM-estimators (`lmrob`) combine high breakdown with high efficiency and are the modern robust-regression default.
- Robust methods are not a license to ignore outliers; always investigate before reporting.
- For heavy-tailed data, a Student-t likelihood (`brms` or `heavy` package) is an alternative to robust M-estimators.
- Robust confidence intervals are usually narrower than a non-robust CI on contaminated data, reflecting the estimator's stability.
- Specialised packages: `robustbase`, `WRS2` (Wilcox), `quantreg` (quantile regression), `MASS::rlm`.

2.20.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.20.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.21 Skewness

2.21.1 Introduction

Skewness is the third standardised moment of a distribution; it quantifies how asymmetric the distribution is around its mean. Zero skewness corresponds to perfect symmetry; positive skewness indicates a long right tail; negative skewness indicates a long left tail. Many biomedical variables (lab values, incubation times, survival times) are right-skewed; income and wealth distributions are classically right-skewed with very long tails.

2.21.2 Prerequisites

The reader should know what a mean and SD are, and should have seen a few histograms.

2.21.3 Theory

The population skewness is

$$\gamma_1 = E\left[\left(\frac{X-\mu}{\sigma}\right)^3\right].$$

Several sample estimators exist with different small-sample biases. The classical moment estimator is $g_1 = m_3/m_2^{3/2}$ where $m_k = \frac{1}{n}\sum(x_i - \bar{x})^k$. SAS/SPSS/`psych` use a bias-corrected form $G_1 = \sqrt{n(n-1)}/(n-2) \cdot g_1$. Interpretation: $|g_1| < 0.5$ is approximately symmetric, $0.5 < |g_1| < 1$ is moderately skewed, $|g_1| > 1$ is highly skewed.

2.21.4 Assumptions

As a descriptive statistic, skewness has no assumptions. As an estimator, it requires finite third moments, which fails for heavy-tailed distributions like the Cauchy.

2.21.5 R Implementation

```r
library(moments)

set.seed(2026)
sym   <- rnorm(2000, 0, 1)
```

```r
rskew <- rlnorm(2000, meanlog = 0, sdlog = 0.7)
lskew <- -rlnorm(2000, meanlog = 0, sdlog = 0.7)

c(symmetric = skewness(sym),
  right     = skewness(rskew),
  left      = skewness(lskew))

# log transform reduces right skewness
skewness(log(rskew))
```

2.21.6 Output & Results

```
symmetric      right       left
    0.013      2.134     -2.090
```

Log-transforming the right-skewed sample drops skewness from 2.1 to approximately 0; this is the standard remedy for biomedical lab values.

2.21.7 Interpretation

Report skewness for any variable whose histogram departs from symmetry. When $|g_1| > 1$, the mean is pulled away from the centre of mass; the median is a better point summary, or the variable should be transformed.

2.21.8 Practical Tips

- Log transformation is the default remedy for right-skewed positive data.
- Visual inspection beats numerical thresholds; a histogram makes the shape immediately visible.
- Skewness in small samples is noisy; avoid thresholds below $n = 30$.
- The Pearson skewness coefficient $3(\text{mean} - \text{median})/\text{SD}$ is a robust alternative.
- For formal tests of skewness, use D'Agostino's test (`moments::agostino.test`).

2.21.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.21.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.22 Standardisation and z-Scores

2.22.1 Introduction

A z-score expresses each observation as its distance from the mean in units of the standard deviation. Standardisation is often applied before clustering, PCA, ridge/lasso regression, and any procedure whose output is unit-sensitive. For data with outliers, a robust variant using the median and MAD is safer.

2.22.2 Prerequisites

The reader should know what the mean, SD, median, and MAD are.

2.22.3 Theory

The classical z-score for observation x_i is

$$z_i = (x_i - \bar{x})/s.$$

After standardisation, the sample mean is 0 and the sample SD is 1. The values are dimensionless and comparable across variables measured on different scales.

The **robust z** uses the median and scaled MAD:

$$z_i^{\text{robust}} = (x_i - \text{median}(x))/(1.4826 \cdot \text{MAD}(x)).$$

It reduces to the classical z for normal data but resists the inflation of the scale estimator by outliers.

Standardisation is not the same as normalisation. **Min-max normalisation** rescales to $[0, 1]$: $(x_i - \min(x))/(\max(x) - \min(x))$. Each has its use; z-scores preserve relative distances better.

2.22.4 Assumptions

Purely algebraic; no distributional assumption. The choice between classical and robust z depends on whether outliers are present.

2.22.5 R Implementation

```r
library(robustbase)

set.seed(2026)
x <- c(rnorm(50, mean = 100, sd = 10), 200, 250)

z_classical <- scale(x)
```

```
z_robust    <- (x - median(x)) / (1.4826 * mad(x, constant = 1))

range(z_classical)
range(z_robust)

cohort <- data.frame(
  age  = rnorm(100, 58, 12),
  sbp  = rnorm(100, 130, 18),
  crp  = rlnorm(100, log(3), 0.6)
)
z_cohort <- scale(cohort)
colMeans(z_cohort)
apply(z_cohort, 2, sd)
```

2.22.6 Output & Results

```
range(z_classical): -1.69   6.11
range(z_robust):    -2.02  15.04
```

The robust z flags the outlier at 250 as > 15 SDs from the median, while the classical z only gives 6. Classical z understates the deviation because the extreme point has inflated the sample SD.

2.22.7 Interpretation

Report z-scores when comparing observations on different scales. Biological examples: centred and scaled gene-expression values across samples; cognitive-test scores scaled to a reference population.

2.22.8 Practical Tips

- `scale()` returns a matrix; wrap with `as.data.frame()` when using downstream tidyverse functions.
- Save the `center` and `scale` attributes to apply the same standardisation to new test data – critical in predictive modelling.
- For skewed variables, log-transform before standardising.
- The robust z is a simple and effective first-pass outlier filter: $|z^{\text{robust}}| > 3$ flags candidates.
- Do not standardise the outcome in a regression if interpretability is desired; standardise predictors if comparing coefficients.

2.22.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.22.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.23 Stem-and-Leaf Plots

2.23.1 Introduction

The stem-and-leaf plot, introduced by John Tukey, is a compact display that shows the distribution of a small sample while preserving every data value. Each observation is split into a "stem" (leading digits) and a "leaf" (trailing digit); stems are listed vertically and leaves are written next to each stem in order. For samples up to about 100 observations, the resulting ASCII plot is a poor man's histogram that can be typed into a notebook.

2.23.2 Prerequisites

The reader should know what a histogram is.

2.23.3 Theory

Given observations, pick a stem width (10s, 100s, etc.) and list each observation by its stem and leaf. For example, values 12, 15, 18, 22, 25, 29, 34 with tens-digit stems:

```
1 | 2 5 8
2 | 2 5 9
3 | 4
```

The back-to-back variant puts two groups on either side of the stem, permitting visual comparison. Modern software rarely uses stem-and-leaf for publication figures, but it remains a quick inspection tool in R's console.

2.23.4 Assumptions

Purely descriptive, requires nothing except numeric data.

2.23.5 R Implementation

```r
set.seed(2026)
x <- round(rnorm(40, mean = 100, sd = 15))
stem(x)
```

```r
# With scale change
stem(x, scale = 0.5)
stem(x, scale = 2)

# Back-to-back
library(aplpack)
x <- rnorm(30, 100, 15)
y <- rnorm(30, 110, 12)
stem.leaf.backback(x, y)
```

2.23.6 Output & Results

```
  The decimal point is 1 digit(s) to the right of the |

   7 | 3
   8 | 0447
   9 | 01135789
  10 | 0111345899
  11 | 0245578
  12 | 34
```

Each leaf on the 10-row represents one observation: 101, 101, 101, 103, 104, etc. The display is itself a histogram with the same-height bars rotated 90 degrees.

2.23.7 Interpretation

Use stem-and-leaf plots for quick eyeballing in R's console; they reveal outliers, skew, and multimodality without generating a figure file.

2.23.8 Practical Tips

- For samples over 100, the plot becomes cluttered; use a histogram.
- `scale` in `stem()` adjusts the stem width; experiment to find the right granularity.
- Back-to-back plots are a useful low-overhead alternative to side-by-side boxplots.
- Stem-and-leaf is rarely used in journal articles today; it is a Tukey EDA tradition, not a reporting convention.
- In teaching contexts, it remains a useful way to introduce distribution shape before histograms.

2.23.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.23.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.24 Summary by Group

2.24.1 Introduction

Split-apply-combine is the workhorse pattern of applied data analysis: split the data by a grouping variable, apply a summary function to each group, and combine the results. It produces the per-group means, medians, counts, and other statistics that populate Table 1 of most scientific papers.

2.24.2 Prerequisites

The reader should know what the mean, median, and SD are, and be comfortable with R's `dplyr` pipe syntax.

2.24.3 Theory

For a dataset with n observations and a grouping variable g taking values $g_1, \dots, g_k$, split-apply-combine computes

$$s_j = f(\{x_i : g_i = g_j\})$$

for each group j, where f is the summary function (mean, median, quantile, any aggregator). The returned object has k rows, one per group.

R implements this via base `aggregate()`, `tapply()`, and `by()`, and via the modern `dplyr::group_by()|>summarise()` and `data.table` equivalents.

2.24.4 Assumptions

Purely descriptive; no distributional assumption. The grouping variable should have enough observations per group for the summary to be meaningful (a mean based on $n = 2$ is noisy).

2.24.5 R Implementation

```
library(dplyr)
library(data.table)
```

```r
library(gtsummary)

set.seed(2026)
trial <- tibble::tibble(
  arm      = factor(rep(c("Placebo", "Active"), each = 75)),
  sex      = factor(sample(c("F", "M"), 150, replace = TRUE)),
  age      = round(rnorm(150, 58, 12)),
  hba1c    = round(c(rnorm(75, 7.3, 0.7), rnorm(75, 6.9, 0.8)), 1),
  event    = rbinom(150, 1, 0.15)
)

# dplyr
trial |>
  group_by(arm) |>
  summarise(
    n         = n(),
    age_mean  = mean(age),
    age_sd    = sd(age),
    hba1c_med = median(hba1c),
    hba1c_iqr = IQR(hba1c),
    event_rate = mean(event)
  )

# Multi-variable grouping
trial |>
  group_by(arm, sex) |>
  summarise(across(c(age, hba1c), list(mean = mean, sd = sd)),
            .groups = "drop")

# data.table
dt <- as.data.table(trial)
dt[, .(n = .N, age_mean = mean(age), hba1c_median = median(hba1c)),
   by = .(arm)]

# Publication-ready
trial |>
  tbl_summary(by = arm, include = c(age, sex, hba1c, event),
              statistic = list(all_continuous() ~ "{mean} ({sd})"))
```

2.24.6 Output & Results

```
# A tibble: 2 x 6
  arm       n age_mean age_sd hba1c_med hba1c_iqr event_rate
  <fct> <int>    <dbl>  <dbl>     <dbl>     <dbl>      <dbl>
1 Placebo  75     57.9   12.1       7.3       0.9     0.173
2 Active   75     57.8   11.8       6.9       1.1     0.120
```

The two arms differ in HbA1c (medians 7.3 vs. 6.9) while being comparable in age and sex distribution – the typical pattern in a Table 1 balance check.

2.24.7 Interpretation

Report grouped summaries for every important subgrouping in a study: treatment arms, sex, study site, risk stratum. The `gtsummary::tbl_summary(by=...)` call produces a fully formatted Table 1 with p-values on request.

2.24.8 Practical Tips

- Always include `n` per group so readers know the denominator.
- Handle missing values explicitly: `summarise(mean(x,na.rm=TRUE))` or prefilter.
- For many summary statistics per variable, `across(everything(),list(mean=mean,sd=sd))` is concise.
- `data.table` is faster for very large datasets; the syntax differs but the logic is identical.
- Report median + IQR for skewed variables and mean + SD for approximately normal ones; mix the two when the Table 1 covers both types.

2.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.24.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.25 Trimmed and Winsorised Means

2.25.1 Introduction

A trimmed mean drops a proportion of observations from each tail before computing the mean. A Winsorised mean replaces those observations with the nearest non-trimmed values and then averages. Both estimators compromise between the arithmetic mean (efficient but sensitive to outliers) and the median (robust but inefficient). They are useful when the data are heavy-tailed but you want to retain most of the sample's information.

2.25.2 Prerequisites

The reader should know what the arithmetic mean and the median are.

2.25.3 Theory

For a sample of size n and a trim proportion $\alpha \in [0, 0.5)$:

- **Trimmed mean**: sort the data, discard the smallest and largest $\lfloor n\alpha \rfloor$ observations, and average the rest.
- **Winsorised mean**: replace the smallest $\lfloor n\alpha \rfloor$ values with the $(\lfloor n\alpha \rfloor + 1)$-th smallest, and similarly at the upper end; then compute the mean.

Special cases: $\alpha = 0$ gives the arithmetic mean; $\alpha = 0.5$ gives the median (exactly for odd n, almost for even n). A 10% or 20% trim is a common compromise.

Trimmed means have a positive breakdown point (proportional to α): a trim of 20% tolerates up to 20% contamination without being affected. Winsorised means have the same breakdown but retain the influence of the replaced values through their "pinning" behaviour.

2.25.4 Assumptions

These estimators are descriptive; no distributional assumption. Their *efficiency* depends on the distribution: the 20% trimmed mean is nearly as efficient as the arithmetic mean under normality but much more robust under heavy tails.

2.25.5 R Implementation

```r
library(psych)
library(DescTools)

set.seed(2026)
x <- c(rnorm(48, mean = 50, sd = 8), 200, 250)  # two outliers

mean(x)
median(x)
mean(x, trim = 0.10)
mean(x, trim = 0.20)
DescTools::Winsorize(x, probs = c(0.10, 0.90)) |> mean()
```

2.25.6 Output & Results

```
[1] 56.62       # arithmetic mean (pulled up by outliers)
[1] 50.30       # median
[1] 51.82       # 10% trimmed mean
```

```
[1] 50.78       # 20% trimmed mean
[1] 51.50       # 10% Winsorised mean
```

The trimmed and Winsorised means align closely with the median, demonstrating their robustness. The arithmetic mean is dragged up by the two extreme points.

2.25.7 Interpretation

Report a trimmed or Winsorised mean when outliers are present, their exclusion is justified, and you want a more efficient summary than the median. Explicitly state the trim proportion.

2.25.8 Practical Tips

- The 10% trim is a useful default; 20% is stronger but loses efficiency.
- Don't let the trim serve as a clean-up for genuine heterogeneity; investigate the outliers first.
- Inference for trimmed means requires a Yuen test (`WRS2::yuen`) rather than the usual t-test.
- Winsorising is preferred when you want to retain the sample size but soften the influence of extremes.
- Combined with robust scale (MAD, Sn), the trimmed mean / SD pair gives a fully robust location / spread summary.

2.25.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.25.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.26 Variance and Standard Deviation

2.26.1 Introduction

Variance and standard deviation are the most widely used dispersion measures in applied statistics. The variance is the mean squared deviation from the mean; the SD is its square root, returning to the original units. Both underlie the normal distribution, the central limit theorem, regression, and every parametric test that refers to "sigma".

2.26.2 Prerequisites

The reader should know the mean and be comfortable squaring, summing, and taking square roots in R.

2.26.3 Theory

For a sample $x_1, \ldots, x_n$:

- **Population variance** (when the data *are* the whole population): $\sigma^2 = \frac{1}{n}\sum(x_i - \mu)^2$.
- **Sample variance** (when the data are a sample): $s^2 = \frac{1}{n-1}\sum(x_i - \bar{x})^2$.

The $n-1$ divisor (Bessel's correction) makes s^2 an unbiased estimator of σ^2. Using n would systematically underestimate σ^2 because $\bar{x}$ is itself estimated from the same data.

The sample **standard deviation** is $s = \sqrt{s^2}$. Even though s^2 is unbiased, s is biased downward because the square root is concave (Jensen's inequality). The bias shrinks like $1/n$ and is negligible for $n > 30$; an unbiased estimator exists (`c_4` correction in SPC) but is rarely used in applied work.

R's `var()` and `sd()` use the $n-1$ divisor by default. To get the MLE version, compute `mean((x-mean(x))^2)`.

2.26.4 Assumptions

Variance and SD are descriptive; no distributional assumption is required. For inferential use (the standard error of the mean is $s/\sqrt{n}$), the usual iid assumption applies.

2.26.5 R Implementation

```r
set.seed(2026)
x <- rnorm(60, mean = 100, sd = 15)

var(x)
sd(x)

# MLE (biased) version with divisor n
mean((x - mean(x))^2)
sqrt(mean((x - mean(x))^2))

# Bias of s: theoretical factor c_4(n) for normal data
c4 <- function(n) sqrt(2 / (n - 1)) * gamma(n / 2) / gamma((n - 1) / 2)
sd_unbiased <- sd(x) / c4(length(x))
sd_unbiased
```

2.26.6 Output & Results

```
[1] 210.5       # s^2 with divisor n-1
[1]  14.5       # s
[1] 207.0       # MLE variance
[1]  14.4       # MLE SD (biased)
[1]  14.6       # bias-corrected SD
```

At $n = 60$, the difference between biased and unbiased SD is a fraction of a percent, irrelevant for applied work.

2.26.7 Interpretation

Report variance when the scale of the squared units is interpretable (e.g., variance of portfolio returns). Report SD when the scale of the original units is desired. In a paper: "mean 100.2 (SD 14.5)" is universally readable.

2.26.8 Practical Tips

- Always specify which divisor you mean when reporting; R's default is $n-1$, which is the statistician's convention.
- For very small samples, the biased-vs-unbiased distinction matters; for most applied work (n > 30) it does not.
- The SD is in the same units as the data; the variance is in units squared. Don't mix them in a single report.
- When a variance is reported without an associated mean, check whether it is actually a standard error (which is smaller by $\sqrt{n}$).
- For skewed data, the SD is not a useful summary; use the IQR instead.

2.26.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.26.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.27 The Weighted Mean

2.27.1 Introduction

A weighted mean is a generalisation of the arithmetic mean in which each observation carries a positive weight that reflects its importance, reliability, or inclu-

sion probability. Weighted means appear in survey sampling (weights are reciprocals of inclusion probabilities), in meta-analysis (weights are inverse-variance), and in weighted regression.

2.27.2 Prerequisites

The reader should know the arithmetic mean and be comfortable with the R function `weighted.mean()`.

2.27.3 Theory

Given observations $x_1, \dots, x_n$ with positive weights $w_1, \dots, w_n$, the weighted mean is

$$\bar{x}_w = \frac{\sum w_i x_i}{\sum w_i}.$$

When all weights are equal, this reduces to the arithmetic mean. The weights rescale each observation's contribution. In inverse-variance weighting, $w_i = 1/s_i^2$, which gives more precise observations more influence.

The weighted variance is

$$s_w^2 = \frac{\sum w_i (x_i - \bar{x}_w)^2}{\sum w_i - 1}$$

(using the frequency-weight convention). For survey weights, a different denominator is used.

2.27.4 Assumptions

- Weights must be positive.
- The choice of weight reflects a real asymmetry (inclusion probability, precision, repetition count); arbitrary weighting is meaningless.

2.27.5 R Implementation

```
library(survey)

# Meta-analysis: four studies with mean and SE
studies <- data.frame(
  mean = c(1.20, 0.95, 1.40, 1.10),
  se   = c(0.15, 0.25, 0.10, 0.20)
)
studies$w <- 1 / studies$se^2
```

```
pooled_mean <- weighted.mean(studies$mean, studies$w)
pooled_se   <- sqrt(1 / sum(studies$w))
c(pooled_mean = pooled_mean, pooled_se = pooled_se)

# Survey weights: 200 respondents, sample weight = 1 / inclusion
 ↪  probability
set.seed(2026)
resp <- data.frame(
  sbp = rnorm(200, 132, 15),
  w   = sample(c(0.5, 1.0, 2.0), 200, replace = TRUE)
)
weighted.mean(resp$sbp, resp$w)
```

2.27.6 Output & Results

```
pooled_mean    pooled_se
      1.236        0.075
```

The inverse-variance weighted mean is 1.24 with SE 0.08 – more precise than the simple average of 1.16 that would ignore the study-level precision.

2.27.7 Interpretation

Meta-analytic pooling gives study-level weights that reflect how much each study contributes to the combined evidence. Survey weights make the computed mean estimate the population mean rather than the sample mean. Always report which weights were used.

2.27.8 Practical Tips

- Negative or zero weights are invalid; check the input.
- For survey data, use the `survey` package (`svymean`, `svydesign`) rather than `weighted.mean()` to get the correct SE accounting for design.
- Inverse-variance weighting is optimal under random-effects meta-analysis only when τ^2 is zero; otherwise weights should use $w_i = 1/(s_i^2 + \tau^2)$.
- When weights are frequencies (integer counts), the weighted mean equals the ordinary mean of the expanded dataset.
- Reporting "the weighted mean" without specifying the weights is ambiguous; include the weight definition in methods.

2.27.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The

links above support a structured review; they do not certify the example or a manuscript's application of the method.

2.27.10 See also — labs in this chapter

- Descriptive statistics and Table 1

2.28 Week 2, Session 1 — Descriptive statistics and Table 1

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

2.28.1 Learning objectives

- Match the right summary statistic to the right variable type — means for symmetric continuous, medians for skewed, counts and percentages for categorical.
- Build a publication-ready Table 1 with `gtsummary::tbl_summary()`.
- Defend the choice of summary with a quick visual check.

2.28.2 Prerequisites

Labs 1.3 through 1.5.

2.28.3 Background

Table 1 is the first real table of almost every clinical paper. It describes the sample: who was in the study, in how many groups, with what characteristics, and — if the study has arms — how balanced those arms are at baseline. A Table 1 that is built thoughtlessly hides an unbalanced comparison; a Table 1 that is built well anchors the rest of the paper.

The right summary depends on the variable. A continuous variable that is roughly symmetric is summarised by its mean and standard deviation; one that is skewed or has heavy tails is summarised by its median and interquartile range. A categorical variable is summarised by counts and percentages. Reporting the median of a balanced continuous variable is not wrong, but it is unusual; reporting the mean of a skewed variable is often misleading.

gtsummary packages these choices into a declarative API. You say which variables to include and which grouping variable to stratify by, and the table is built with sensible defaults. When you need non-standard behaviour, you override per-variable.

2.28.4 Setup

```
library(tidyverse)
library(palmerpenguins)
library(gtsummary)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

2.28.5 1. Goal

Produce a Table 1 for the `penguins` dataset stratified by species, and back the choice of summary with a diagnostic plot.

2.28.6 2. Approach

```
dat <- penguins |>
  drop_na(species, sex, bill_length_mm, bill_depth_mm,
          flipper_length_mm, body_mass_g)
```

2.28.7 3. Execution

Pick the right summary per variable. First, eyeball the shapes.

```
dat |>
  pivot_longer(c(bill_length_mm, bill_depth_mm,
                 flipper_length_mm, body_mass_g),
               names_to = "variable", values_to = "value") |>
  ggplot(aes(value, fill = species)) +
  geom_histogram(bins = 20, alpha = 0.6, colour = "grey30",
                 position = "identity") +
  facet_wrap(~ variable, scales = "free") +
  labs(x = NULL, y = "Count")
```

All four continuous variables are approximately symmetric within species, so mean (SD) is defensible. If a variable were skewed, we would switch to median (IQR).

```
t1 <- dat |>
  select(species, sex, bill_length_mm, bill_depth_mm,
         flipper_length_mm, body_mass_g) |>
  tbl_summary(
    by = species,
    statistic = list(
      all_continuous() ~ "{mean} ({sd})",
      all_categorical() ~ "{n} ({p}%)"
    ),
```

```
    digits = all_continuous() ~ 1,
    label = list(
      bill_length_mm    ~ "Bill length (mm)",
      bill_depth_mm     ~ "Bill depth (mm)",
      flipper_length_mm ~ "Flipper length (mm)",
      body_mass_g       ~ "Body mass (g)",
      sex               ~ "Sex"
    )
  ) |>
  add_overall() |>
  add_p()
t1
```

`add_p()` will run a t-test or ANOVA on continuous variables and a chi-square or Fisher's exact test on categorical, with sensible defaults for sample size.

2.28.8 4. Check

Validate the gtsummary output against a hand computation.

```
dat |>
  group_by(species) |>
  summarise(
    n = n(),
    mean_mass = mean(body_mass_g),
    sd_mass   = sd(body_mass_g),
    .groups = "drop"
  )
```

The group means and standard deviations must match the numbers in the body-mass row of the Table 1.

2.28.9 5. Report

> Table 1 describes `nrow(dat)` penguins from three species in the Palmer Archipelago. Body mass was highest in Gentoo (`round(mean(dat$body_mass_g[dat$species=="Gentoo"]),0)` ± `round(sd(dat$body_mass_g[dat$species=="Gentoo"]),0)` g) and lower in Adelie and Chinstrap. All four continuous variables were approximately symmetric within species and are summarised as mean (SD); sex was approximately balanced within each species.

Table 1 is not a tool for hypothesis testing. The p-values in the rightmost column describe the data, not the biology. A statistically-significant difference between arms at baseline in a randomised trial is, by construction, a chance finding.

2.28.10 Common pitfalls

- Reporting a mean for a skewed variable.
- Using Table 1 as a place to "prove balance" with p-values in a randomised trial.
- Suppressing the missing-data row. Always report it.
- Forgetting that `tbl_summary` drops NAs by default; check the count at the bottom of each column.

2.28.11 Further reading

- Sjoberg DD et al. *Reproducible summary tables with the gtsummary package*. R Journal.
- Altman DG. *Practical Statistics for Medical Research*, chapter on describing data.

2.28.12 Session info

```
sessionInfo()
```

2.28.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 3

Probability Theory

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-10

The probability machinery the rest of the book leans on: Bayes' rule, common discrete and continuous distributions, the law of total probability, Q-Q diagnostics, and density estimation. The chapter is short on theory and long on R-side intuition.

This chapter contains **40 method pages** and **3 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

3.1 Method pages

Method	Source slug
Bayes' Theorem	`bayes-theorem`
Conditional Distributions	`conditional-distributions`
Conditional Probability	`conditional-probability`
Convolutions of Distributions	`convolutions`
Copulas: An Introduction	`copulas-introduction`
Discrete vs Continuous Variables	`discrete-vs-continuous`
Expectation of a Random Variable	`expectation`
Independence	`independence`
Joint Distributions	`joint-distributions`
Kolmogorov's Axioms of Probability	`kolmogorov-axioms`
Law of Total Probability	`law-of-total-probability`
Marginal Distributions	`marginal-distributions`
Random Variables	`random-variables`
Sample Space and Events	`sample-space-events`
Student's t-Distribution	`t-distribution`

Method	Source slug
The Bernoulli Distribution	`bernoulli-distribution`
The Beta Distribution	`beta-distribution`
The Binomial Distribution	`binomial-distribution`
The Chi-Squared Distribution	`chi-squared-distribution`
The Correlation Coefficient	`correlation-coefficient`
The Cumulative Distribution Function	`cumulative-distribution-function`
The Exponential Distribution	`exponential-distribution`
The F Distribution	`f-distribution`
The Gamma Distribution	`gamma-distribution`
The Geometric Distribution	`geometric-distribution`
The Hazard Function	`hazard-function`
The Hypergeometric Distribution	`hypergeometric-distribution`
The Log-Normal Distribution	`lognormal-distribution`
The Multinomial Distribution	`multinomial-distribution`
The Multivariate Normal Distribution	`multivariate-normal`
The Negative Binomial Distribution	`negative-binomial-distribution`
The Normal Distribution	`normal-distribution`
The Poisson Distribution	`poisson-distribution`
The Probability Density Function	`probability-density-function`
The Probability Mass Function	`probability-mass-function`
The Survival Function	`survival-function`
The Uniform Distribution	`uniform-distribution`
The Weibull Distribution	`weibull-distribution`
Transformations of Random Variables	`transformations-of-rv`
Variance and Covariance	`variance-covariance-of-rv`

3.2 Labs

Lab
Probability, conditional probability, Bayes' theorem
Discrete distributions: Bernoulli, binomial, Poisson
Continuous distributions and Q-Q plots

3.3 Bayes' Theorem

3.3.1 Introduction

Bayes' theorem is the rule for inverting conditional probabilities: given $P(B \mid A)$, it tells us $P(A \mid B)$. It is the mathematical bridge between diagnostic sensitivity and positive predictive value, between forward simulation and inverse

inference, between a likelihood function and a posterior distribution. Every Bayesian analysis, every clinical prediction, and every probabilistic classifier is an application of Bayes' theorem in disguise.

3.3.2 Prerequisites

Conditional probability and the law of total probability.

3.3.3 Theory

For events A and B with $P(B) > 0$,

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}.$$

When $\{A_i\}$ forms a partition of the sample space, the denominator expands by the law of total probability:

$$P(A_i \mid B) = \frac{P(B \mid A_i)\, P(A_i)}{\sum_j P(B \mid A_j)\, P(A_j)}.$$

In the Bayesian vocabulary:

- $P(A)$ is the **prior**: what was believed before observing B.
- $P(B \mid A)$ is the **likelihood**: how likely B is under each hypothesis.
- $P(A \mid B)$ is the **posterior**: updated belief about A after observing B.
- $P(B)$ is the **marginal likelihood** or "evidence".

Posterior $\propto$ likelihood $\times$ prior is the single most important slogan in Bayesian statistics.

3.3.4 Assumptions

- $P(B) > 0$; otherwise conditioning is undefined.
- The prior $P(A)$ must be specified; choosing it is where domain knowledge enters.

3.3.5 R Implementation

```r
library(ggplot2)

# Diagnostic test: prior = prevalence, likelihood = sensitivity and
 ↪  specificity
sensitivity <- 0.92
specificity <- 0.97
prevalences <- seq(0.001, 0.5, length.out = 100)
```

```
ppv <- sensitivity * prevalences /
       (sensitivity * prevalences + (1 - specificity) * (1 - prevalences))

df <- data.frame(prevalence = prevalences, ppv = ppv)

ggplot(df, aes(prevalence, ppv)) +
  geom_line(colour = "#2A9D8F", linewidth = 1) +
  geom_vline(xintercept = 0.01, linetype = "dashed", colour = "#F4A261") +
  geom_vline(xintercept = 0.30, linetype = "dashed", colour = "#F4A261") +
  labs(x = "Disease prevalence",
       y = "Positive predictive value",
       title = "PPV depends strongly on prevalence (Bayes' theorem)") +
  theme_minimal()

c(prevalence_001 = ppv[which.min(abs(prevalences - 0.01))],
  prevalence_030 = ppv[which.min(abs(prevalences - 0.30))])
```

3.3.6 Output & Results

```
prevalence_001 prevalence_030
        0.2368         0.9290
```

At 1% prevalence, a positive test on a 92%-sensitive, 97%-specific assay gives only 24% probability of disease. At 30% prevalence, the same test yields a 93% PPV. The test characteristics are unchanged; the posterior depends crucially on the prior.

3.3.7 Interpretation

The prevalence dependence of PPV is the canonical real-world application of Bayes' theorem. Reporting "our test has 92% sensitivity and 97% specificity" is meaningless without the prevalence of the intended use population; the clinically relevant quantity is PPV.

3.3.8 Practical Tips

- Never cite sensitivity / specificity in isolation when discussing diagnostic utility; always give the PPV and NPV at the target prevalence.
- Bayes' theorem scales trivially to many hypotheses via the partition form; for continuous parameter spaces, the sums become integrals.
- Prior choice is contentious in some Bayesian applications; weakly informative priors (Cauchy, normal with wide scale) are standard defaults.
- Updating sequentially: the posterior after observation 1 becomes the prior for observation 2. Chains of updates are equivalent to a single update with all observations.

- The **likelihood ratio** $P(B \mid A)/P(B \mid A^c)$ converts the prior odds of A to the posterior odds multiplicatively – often a cleaner formulation than raw probabilities.

3.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.3.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.4 The Bernoulli Distribution

3.4.1 Introduction

The Bernoulli distribution models a single binary trial: success vs. failure, disease vs. healthy, click vs. no click. It is the simplest non-trivial distribution and the building block of the binomial, geometric, and negative binomial distributions.

3.4.2 Prerequisites

Random variables and basic probability.

3.4.3 Theory

A random variable X is **Bernoulli(p)** if it takes values in $\{0, 1\}$ with

$$P(X = 1) = p, \qquad P(X = 0) = 1 - p.$$

PMF: $p_X(x) = p^x(1-p)^{1-x}$ for $x \in \{0, 1\}$.

Moments:

- $E[X] = p$.
- $\mathrm{Var}(X) = p(1-p)$, maximised at $p = 0.5$ (value 0.25).
- $E[X^k] = p$ for every $k \geq 1$ (since $X^k = X$).

The sum of n iid Bernoulli(p) variables is **Binomial**(n, p).

MLE of p from an iid sample $X_1, \ldots, X_n$: $\hat{p} = \bar{X}$, the sample proportion.

3.4.4 Assumptions

Independence across trials and a common probability p. Violations of either produce overdispersion (if p varies) or correlation (if trials are dependent).

3.4.5 R Implementation

```r
set.seed(2026)

p <- 0.3
n <- 10000

# Sample from Bernoulli(p) via rbinom with size = 1
X <- rbinom(n, size = 1, prob = p)

# Sample mean is the MLE of p
c(p_hat = mean(X), p_true = p)

# Sample variance approximates p(1 - p)
c(sample_var = var(X), theoretical = p * (1 - p))

# 95% Wald confidence interval for p
se_p <- sqrt(mean(X) * (1 - mean(X)) / n)
c(lower = mean(X) - 1.96 * se_p, upper = mean(X) + 1.96 * se_p)

# Wilson (better for small p or small n)
binom::binom.confint(sum(X), n, conf.level = 0.95, methods = "wilson")
```

3.4.6 Output & Results

```
    p_hat    p_true
   0.3002     0.3000

sample_var theoretical
   0.2101      0.2100

   lower    upper
  0.2912   0.3092

method    x    n    mean    lower    upper
Wilson 3002 10000  0.3002   0.2912   0.3094
```

Empirical mean is $0.3002 \approx p$; empirical variance is $0.2101 \approx p(1-p) = 0.21$; Wald and Wilson CIs agree in this range.

3.4.7 Interpretation

In applied work, the Bernoulli is the underlying distribution of every binary outcome: adverse event, responder status, screen-positive. Reporting usually aggregates to proportions across n independent Bernoullis – i.e., to the binomial framework.

3.4.8 Practical Tips

- For small samples or extreme p (near 0 or 1), use Wilson or Clopper-Pearson CIs instead of Wald.
- Overdispersion (variance exceeding $p(1-p)$) in grouped data signals unobserved heterogeneity; consider beta-binomial or mixed-effects logistic regression.
- `rbinom(n,size=1,prob=p)` is the direct way to simulate Bernoullis; `sample(0:1,n,replace=TRUE,prob=c(1-p,p))` is equivalent.
- The Bernoulli likelihood is the foundation of logistic regression; MLE of p from regression coefficients requires the logit link.
- Two dependent Bernoullis require a specification of their joint distribution – usually via a copula or a 2x2 joint PMF.

3.4.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.4.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.5 The Beta Distribution

3.5.1 Introduction

The beta distribution is the flexible family of probabilities on the unit interval $[0, 1]$. It is the natural distribution for proportions, success probabilities, and measurements bounded between two fixed limits. In Bayesian inference, it is the conjugate prior for the binomial and Bernoulli likelihoods.

3.5.2 Prerequisites

Continuous distributions, Bayesian inference basics.

3.5.3 Theory

A beta random variable with shape parameters $a > 0$ and $b > 0$ has PDF

$$f(x) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}, \qquad 0 \le x \le 1,$$

where $B(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$ is the beta function.

Moments:

- $E[X] = a/(a+b)$.
- $\mathrm{Var}(X) = ab/[(a+b)^2(a+b+1)]$.

Special cases:

- $a = b = 1$: Uniform$(0,1)$.
- $a = b = 0.5$: Jeffreys prior for the binomial probability.
- $a, b > 1$: unimodal density with mode $(a-1)/(a+b-2)$.
- $a < 1$ and $b < 1$: bimodal (poles at 0 and 1).

Conjugacy: given a Bernoulli/Binomial likelihood with success probability θ and a Beta(a,b) prior on θ, the posterior after observing k successes in n trials is Beta$(a+k, b+n-k)$.

3.5.4 Assumptions

The variable must be bounded between 0 and 1. For probabilities near 0 or 1, the distribution can have a pole; use care when simulating.

3.5.5 R Implementation

```r
curve(dbeta(x, 2, 5),  from = 0, to = 1, col = "steelblue", lwd = 2,
      main = "Beta densities", ylab = "f(x)")
curve(dbeta(x, 5, 2),  add = TRUE, col = "#F4A261", lwd = 2)
curve(dbeta(x, 0.5, 0.5), add = TRUE, col = "#6A4C93", lwd = 2)
curve(dbeta(x, 2, 2),  add = TRUE, col = "#2A9D8F", lwd = 2)
legend("top", legend = c("Beta(2,5)", "Beta(5,2)", "Beta(0.5,0.5)",
 ↪  "Beta(2,2)"),
       col = c("steelblue", "#F4A261", "#6A4C93", "#2A9D8F"), lwd = 2)

# Moments
a <- 2; b <- 5
c(mean = a / (a + b), var = a * b / ((a + b)^2 * (a + b + 1)))
```

```r
X <- rbeta(1e5, a, b)
c(emp_mean = mean(X), emp_var = var(X))

# Bayesian update example
prior_a <- 2; prior_b <- 2
k <- 15; n <- 20
post_a <- prior_a + k; post_b <- prior_b + n - k
c(posterior_mean = post_a / (post_a + post_b),
  posterior_95_CI_lower = qbeta(0.025, post_a, post_b),
  posterior_95_CI_upper = qbeta(0.975, post_a, post_b))
```

3.5.6 Output & Results

```
   mean      var
 0.2857   0.0255

 emp_mean   emp_var
   0.286     0.025

posterior_mean posterior_95_CI_lower posterior_95_CI_upper
         0.708                 0.504                 0.871
```

Empirical moments match theoretical. The Bayesian update with a Beta(2,2) prior and 15 successes in 20 trials gives a posterior mean of 0.71 with a 95% credible interval of (0.50, 0.87).

3.5.7 Interpretation

Beta distributions appear everywhere in Bayesian analysis of proportions. Reporting "posterior probability of success 0.71 (95% credible interval 0.50 to 0.87)" is a beta-based statement.

3.5.8 Practical Tips

- For weakly informative priors, Beta(1, 1) (uniform) or Beta(0.5, 0.5) (Jeffreys) are common defaults.
- Very large a and b correspond to strongly informative priors; check the prior predictive distribution to verify plausibility.
- Beta regression (`betareg`) models proportions in $(0, 1)$ as a function of covariates.
- Values exactly at 0 or 1 are problematic; transform via $(x(n-1)+0.5)/n$ or use zero-one-inflated beta.
- For a quick-look sanity check, a beta with $a = b$ is symmetric and centred at 0.5, scaling by $(a + b)$ as a concentration parameter.

3.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.5.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.6 The Binomial Distribution

3.6.1 Introduction

The binomial distribution counts the number of successes in n independent trials, each with the same probability p. It is the most common discrete distribution in applied statistics: responder counts in a trial, positive test results in a screen, correct classifications in an evaluation.

3.6.2 Prerequisites

Bernoulli trials, basic combinatorics.

3.6.3 Theory

If $X = \sum_{i=1}^{n} B_i$ where B_i are iid Bernoulli(p), then $X \sim \text{Binomial}(n, p)$.

PMF: $P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$ for $k = 0, 1, \ldots, n$.

Moments:

- $E[X] = np$.
- $\text{Var}(X) = np(1-p)$.

Normal approximation: $\text{Binomial}(n, p) \approx \mathcal{N}(np, np(1-p))$ when $np \geq 10$ and $n(1-p) \geq 10$. Continuity-corrected: $P(X \leq k) \approx \Phi\big((k + 0.5 - np)/\sqrt{np(1-p)}\big)$.

Poisson approximation: for large n and small p with $np = \lambda$ moderate, $\text{Binomial}(n, p) \approx \text{Poisson}(\lambda)$.

Relationship to other distributions:

- Sum of independent binomials with same p: binomial with sum of n's.

- Conditional on total successes, two independent binomials have a hypergeometric distribution of one.

3.6.4 Assumptions

- Fixed number of trials n.
- Independent trials.
- Constant success probability p.

Violations produce overdispersion (for varying p) or correlation (for dependent trials).

3.6.5 R Implementation

```r
library(ggplot2)

n <- 20; p <- 0.35

k <- 0:n
pmf <- dbinom(k, n, p)

data.frame(k, pmf = round(pmf, 4))

# Moments
c(mean = n * p, var = n * p * (1 - p))
mean(rbinom(1e5, n, p)); var(rbinom(1e5, n, p))

# Normal approximation
x <- seq(0, n, length.out = 400)
norm_approx <- dnorm(x, mean = n * p, sd = sqrt(n * p * (1 - p)))

ggplot() +
  geom_col(data = data.frame(k, pmf), aes(k, pmf), fill = "#2A9D8F") +
  geom_line(data = data.frame(x, norm_approx), aes(x, norm_approx),
            colour = "#F4A261", linewidth = 1) +
  labs(x = "k", y = "probability",
       title = "Binomial(20, 0.35) with normal approximation") +
  theme_minimal()

# Interval probability and quantile
pbinom(10, n, p)        # P(X <= 10)
qbinom(0.025, n, p)     # 2.5th percentile
```

3.6.6 Output & Results

```
    k     pmf
```

```
1   0 0.0020
2   1 0.0219
...
20 20 0.0000

    mean     var
     7.0     4.55

[1] 0.9468     # P(X <= 10)
[1]      3     # 2.5th percentile
```

The normal approximation tracks the PMF closely at $n = 20$, $p = 0.35$ where $np = 7$ and $n(1 - p) = 13$.

3.6.7 Interpretation

Binomial reasoning pervades applied statistics: one-proportion tests, two-proportion tests, logistic regression, and every chi-squared-style comparison. Reporting a proportion with a Wilson 95% CI is reporting a binomial inference.

3.6.8 Practical Tips

- For small n or extreme p, use the exact binomial test / Clopper-Pearson interval instead of normal-based inference.
- Overdispersion relative to binomial signals either varying p across trials or correlation among trials; the beta-binomial is a common remedy.
- `dbinom(k,n,p,log=TRUE)` gives log-probabilities, useful for likelihoods with many observations.
- Compound inferences (e.g., comparing two binomial proportions) have their own CIs; prefer score intervals (`prop.test`) over Wald.
- In R the order of arguments is `dbinom(x,size,prob)` where `size` is n and `prob` is p; mixing them up is a common bug.

3.6.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.6.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.7 The Chi-Squared Distribution

3.7.1 Introduction

The chi-squared distribution arises as the sum of squared independent standard normal random variables. It is the sampling distribution of the sample variance under normality, the reference distribution for goodness-of-fit and contingency tests, and a key ingredient in the F distribution used throughout ANOVA.

3.7.2 Prerequisites

Normal distribution, basic calculus.

3.7.3 Theory

If $Z_1, \dots, Z_k$ are iid standard normal, then

$$Q = \sum_{i=1}^{k} Z_i^2 \sim \chi^2_k.$$

The parameter k is the **degrees of freedom**. The PDF is

$$f(x) = \frac{1}{2^{k/2}\Gamma(k/2)} x^{k/2-1} e^{-x/2}, \qquad x > 0.$$

Moments: $E[Q] = k$, $\mathrm{Var}(Q) = 2k$.

Sum property: if $Q_1 \sim \chi^2_{k_1}$ and $Q_2 \sim \chi^2_{k_2}$ are independent, then $Q_1 + Q_2 \sim \chi^2_{k_1+k_2}$.

Sample variance: for iid normal $X_1, \dots, X_n$ with variance σ^2,

$$\frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1}.$$

This is the basis of confidence intervals for a normal variance.

Large-df behaviour: χ^2_k is approximately normal for large k (by the CLT applied to the sum of squared normals).

3.7.4 Assumptions

For the sample-variance result: iid normal observations. For the goodness-of-fit approximation in chi-squared tests: independent observations and large enough expected cell counts.

3.7.5 R Implementation

```r
k <- 5

# PDF and CDF
x <- seq(0, 20, length.out = 400)
plot(x, dchisq(x, df = k), type = "l", col = "#2A9D8F", lwd = 2,
     main = paste("Chi-squared,", k, "df"), ylab = "f(x)")

# Moments
c(theoretical_mean = k, theoretical_var = 2 * k,
  empirical_mean   = mean(rchisq(1e5, k)),
  empirical_var    = var(rchisq(1e5, k)))

# Verify: sum of k squared standard normals
n <- 1e5
Q_direct <- rchisq(n, df = k)
Q_from_normals <- rowSums(matrix(rnorm(n * k), n, k)^2)
c(mean_direct = mean(Q_direct), mean_from_normals = mean(Q_from_normals))

# Confidence interval for a variance
set.seed(2026)
X <- rnorm(30, mean = 0, sd = 2)
s2 <- var(X)
n <- length(X)
chi_l <- qchisq(0.025, n - 1); chi_u <- qchisq(0.975, n - 1)
c((n - 1) * s2 / chi_u, (n - 1) * s2 / chi_l)
```

3.7.6 Output & Results

```
theoretical_mean  theoretical_var  empirical_mean  empirical_var
             5.0             10.0           5.003           9.98

mean_direct mean_from_normals
      5.005             5.004

[1] 2.542 7.206
```

A 95% CI for the true variance (which is 4) is (2.54, 7.21), which captures the true value. The sum-of-squared-normals construction produces the expected mean of 5.

3.7.7 Interpretation

Chi-squared tests of independence, goodness-of-fit, and variance homogeneity all reference the chi-squared distribution. Reporting a test statistic value like

"$\chi^2_3 = 9.5$, $p = 0.023$" is reporting a chi-squared-distributed statistic with its p-value.

3.7.8 Practical Tips

- Chi-squared confidence intervals for variance are highly sensitive to the normality assumption.
- For large df, use the normal approximation: $\chi^2_k \approx \mathcal{N}(k, 2k)$, or more accurately $\sqrt{2\chi^2_k} \approx \mathcal{N}(\sqrt{2k-1}, 1)$.
- Chi-squared tests with small expected counts are poorly approximated; use exact tests or Monte Carlo.
- Non-central chi-squared (with non-centrality parameter λ) governs power calculations for chi-squared tests.
- The square of a t_ν is $F_{1,\nu}$, not chi-squared; do not conflate.

3.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.7.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.8 Conditional Distributions

3.8.1 Introduction

A conditional distribution describes how one random variable behaves given a specific value (or values) of another. Conditional expectations are what regression models estimate; conditional variances quantify residual uncertainty. Together they are the language of predictive statistics.

3.8.2 Prerequisites

Joint distributions, marginal distributions, conditional probability.

3.8.3 Theory

For continuous joint density $f_{X,Y}$ with marginal $f_Y > 0$, the **conditional density of X given $Y = y$** is

$$f_{X|Y}(x \mid y) = \frac{f_{X,Y}(x, y)}{f_Y(y)}.$$

Analogous expression for discrete variables using PMFs.

Conditional expectation of X given $Y = y$:

$$E[X \mid Y = y] = \int x f_{X|Y}(x \mid y)\, dx.$$

As a function of y, $E[X \mid Y]$ is itself a random variable.

Tower property (law of total expectation):

$$E[X] = E[E[X \mid Y]].$$

Conditional variance: $\mathrm{Var}(X \mid Y = y) = E[X^2 \mid Y = y] - (E[X \mid Y = y])^2$.

Law of total variance: $\mathrm{Var}(X) = E[\mathrm{Var}(X \mid Y)] + \mathrm{Var}(E[X \mid Y])$. Within-group variance plus between-group variance.

For a bivariate normal with correlation ρ: $X \mid Y = y \sim \mathcal{N}(\mu_X + \rho\sigma_X\sigma_Y^{-1}(y - \mu_Y), \sigma_X^2(1 - \rho^2))$.

3.8.4 Assumptions

$f_Y(y) > 0$ (or the conditional probability is defined via an alternate limiting procedure); $E[|X|] < \infty$ for the conditional expectation to exist.

3.8.5 R Implementation

```r
library(mvtnorm)
set.seed(2026)

# Bivariate normal
mu <- c(0, 0); Sigma <- matrix(c(1, 0.7, 0.7, 1), 2, 2)
rho <- Sigma[1, 2] / sqrt(prod(diag(Sigma)))
XY <- rmvnorm(5e4, mean = mu, sigma = Sigma)

# Conditional on Y near y0, distribution of X
y0 <- 1.0; window <- 0.05
X_given_Y <- XY[abs(XY[, 2] - y0) < window, 1]
```

```
c(n_window    = length(X_given_Y),
  emp_mean    = mean(X_given_Y),
  theor_mean  = mu[1] + rho * (y0 - mu[2]),
  emp_sd      = sd(X_given_Y),
  theor_sd    = sqrt(Sigma[1, 1] * (1 - rho^2)))

# Tower property: E[X] = E[E[X | Y]]
c(E_X = mean(XY[, 1]),
  tower = mean(mu[1] + rho * (XY[, 2] - mu[2])))

# Law of total variance
var_total <- var(XY[, 1])
var_within <- Sigma[1, 1] * (1 - rho^2)
var_between <- rho^2 * Sigma[1, 1]
c(total = var_total,
  sum_of_parts = var_within + var_between)
```

3.8.6 Output & Results

```
n_window   emp_mean theor_mean     emp_sd   theor_sd
  4094        0.693      0.700      0.708      0.714

       E_X       tower
     0.003      -0.001

       total sum_of_parts
       1.000        1.000
```

Empirical conditional moments match theoretical; tower property and law of total variance hold exactly (up to Monte Carlo error).

3.8.7 Interpretation

Regression is the enterprise of estimating $E[Y \mid X = x]$ as a function of x. When a manuscript reports "for every 1-unit increase in x, y increases by β", it is interpreting a slope of the conditional expectation.

3.8.8 Practical Tips

- Conditional distributions require the conditioning event to have positive probability (or a limiting argument for continuous variables).
- $E[Y \mid X]$ is a function of X and therefore a random variable; $E[Y \mid X = x]$ is a specific number.
- The conditional expectation is the best predictor of Y from X under squared-error loss (Hilbert projection).

- Jensen's inequality extends to conditional expectations: $E[g(X) \mid Y] \geq g(E[X \mid Y])$ for convex g.
- In high-dimensional conditioning, direct computation is intractable; regression models (linear, generalised, spline-based) estimate $E[Y \mid X]$ directly.

3.8.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.8.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.9 Conditional Probability

3.9.1 Introduction

Conditional probability quantifies how information about one event updates the probability of another. It is the lens through which every diagnostic test, every Bayesian update, and every hazard calculation is viewed. The rules are simple; the interpretations are subtle and recurrently misunderstood in applied research.

3.9.2 Prerequisites

Basic probability, sample spaces, and set operations.

3.9.3 Theory

For events A and B with $P(B) > 0$, the **conditional probability of A given B** is

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}.$$

Conceptually, conditioning restricts attention to the subset of the sample space where B occurred, and then asks: among those outcomes, what fraction are in A?

Rearranging gives the **product (multiplication) rule**:

$$P(A \cap B) = P(A \mid B)\, P(B) = P(B \mid A)\, P(A).$$

The product rule extends to multiple events via the chain rule:

$$P(A_1 \cap \ldots \cap A_n) = P(A_1)\, P(A_2 \mid A_1)\, P(A_3 \mid A_1 \cap A_2) \cdots$$

Conditioning preserves the axioms: fixing B, the function $P(\cdot \mid B)$ is itself a valid probability measure on the conditional sample space. All standard properties (complement rule, inclusion-exclusion) hold with P replaced by $P(\cdot \mid B)$.

3.9.4 Assumptions

- $P(B) > 0$. Conditioning on a zero-probability event is not defined in the elementary setting; measure-theoretic extensions exist but are beyond scope.
- The original probability space is fixed; conditioning does not change Ω, only the measure restricted to B.

3.9.5 R Implementation

We use a 2x2 population table to demonstrate conditioning mechanically.

```
set.seed(2026)

# Simulated screening-test data: 10 000 patients
# True disease status and test result
n <- 10000
disease_prev <- 0.05
sensitivity  <- 0.90
specificity  <- 0.95

patients <- data.frame(
  disease = rbinom(n, 1, disease_prev)
) |>
  within({
    test <- ifelse(disease == 1,
                   rbinom(length(disease), 1, sensitivity),
                   rbinom(length(disease), 1, 1 - specificity))
  })

tab <- table(patients$test, patients$disease,
             dnn = c("test", "disease"))
tab
```

```r
p_disease_given_test_pos <- tab["1", "1"] / sum(tab["1", ])
p_disease_given_test_pos

p_test_pos_given_disease <- tab["1", "1"] / sum(tab[, "1"])
p_test_pos_given_disease

p_joint <- tab["1", "1"] / n
c(joint = p_joint,
  formula = p_test_pos_given_disease * mean(patients$disease == 1))
```

3.9.6 Output & Results

```
       disease
test        0    1
     0   9030   48
     1    472  450

[1] 0.4881    # P(disease | test+)
[1] 0.9036    # P(test+ | disease)

joint   formula
0.0450    0.0450
```

The product rule is verified: $P(\text{test}+ \cap \text{disease}) = P(\text{test}+ \mid \text{disease}) \cdot P(\text{disease})$. And $P(\text{disease} \mid \text{test}+) = 0.49$ – far below the sensitivity of 0.90, because disease prevalence is low.

3.9.7 Interpretation

The final ratio – 49% positive predictive value despite 90% sensitivity – is the classic base-rate lesson in clinical testing. In a manuscript: "positive predictive value was 49%, reflecting the low prevalence of disease in the screened population despite a sensitive test."

3.9.8 Practical Tips

- $P(A \mid B) \neq P(B \mid A)$ in general; this is the prosecutor's fallacy when confused in legal and clinical settings.
- Always check that the conditioning event has positive probability; in small samples, conditional probabilities based on empty strata are undefined.
- When interpreting a conditional probability, state the conditioning event explicitly: "the probability of disease *given that the test is positive*" is clearer than "the positive predictive value" for non-specialist readers.
- Chain the product rule when computing joint probabilities of many events: $P(A \cap B \cap C) = P(A)P(B \mid A)P(C \mid A \cap B)$.

- Conditioning changes the denominator: moving from marginal to conditional probability requires dividing by $P(B)$.

3.9.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.9.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.10 Convolutions of Distributions

3.10.1 Introduction

The distribution of a sum of independent random variables is their **convolution**. Many standard distribution families are closed under convolution: sums of independent normals are normal, sums of independent Poissons are Poisson, sums of independent gammas (with the same rate) are gamma. This closure is what makes convolution a core operation in probability.

3.10.2 Prerequisites

Joint distributions, independence, integration.

3.10.3 Theory

For independent X and Y with densities f_X and f_Y, the density of $Z = X + Y$ is

$$f_Z(z) = \int_{-\infty}^{\infty} f_X(x) f_Y(z - x)\, dx.$$

The integral is the **convolution** $f_X * f_Y$. For discrete PMFs, the sum replaces the integral.

Characteristic functions convert convolutions to products: $\varphi_{X+Y}(t) = \varphi_X(t)\varphi_Y(t)$. This makes convolutions of many distributions analytically tractable via Fourier-analytic methods.

Closure examples:

- $\mathcal{N}(\mu_1, \sigma_1^2) + \mathcal{N}(\mu_2, \sigma_2^2) \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$ (independent).
- $\text{Poisson}(\lambda_1) + \text{Poisson}(\lambda_2) \sim \text{Poisson}(\lambda_1 + \lambda_2)$ (independent).
- $\text{Gamma}(\alpha_1, \beta) + \text{Gamma}(\alpha_2, \beta) \sim \text{Gamma}(\alpha_1 + \alpha_2, \beta)$ (same rate; independent).
- $\chi^2_{n_1} + \chi^2_{n_2} \sim \chi^2_{n_1+n_2}$ (independent).
- $\text{Binomial}(n_1, p) + \text{Binomial}(n_2, p) \sim \text{Binomial}(n_1 + n_2, p)$ (same p; independent).

Note that the Cauchy family is also closed under sums: sums of independent Cauchys are Cauchy – a consequence of the stable distribution family to which Cauchy belongs.

3.10.4 Assumptions

Independence is essential; the convolution formula fails for dependent X and Y.

3.10.5 R Implementation

```r
set.seed(2026)

# Normal + normal is normal
X <- rnorm(1e5, mean = 0, sd = 1)
Y <- rnorm(1e5, mean = 0, sd = 1)
Z <- X + Y
c(emp_mean = mean(Z), emp_sd = sd(Z),
  theoretical_mean = 0, theoretical_sd = sqrt(2))

# Poisson + Poisson is Poisson
X_pois <- rpois(1e5, lambda = 3)
Y_pois <- rpois(1e5, lambda = 2)
Z_pois <- X_pois + Y_pois
c(emp_mean = mean(Z_pois),
  emp_var  = var(Z_pois),
  theoretical_mean = 5,
  theoretical_var  = 5)

# Numerical convolution of two PDFs via integrate()
# f_X: uniform[0,1], f_Y: exponential(rate=1)
# Compute density of Z = X + Y at z = 0.5
f_convolve <- function(z) {
  integrate(function(x) dunif(x) * dexp(z - x, rate = 1),
            lower = 0, upper = min(1, z))$value
}
f_convolve(0.5)
f_convolve(1.5)
```

3.10.6 Output & Results

```
 emp_mean    emp_sd  theoretical_mean  theoretical_sd
   0.006     1.414             0.000          1.414

  emp_mean   emp_var  theoretical_mean  theoretical_var
     5.002     5.014             5.000            5.000

[1] 0.3935    # density at z = 0.5
[1] 0.3894    # density at z = 1.5
```

Empirical sums match theoretical distributions for both the continuous (normal) and discrete (Poisson) closure cases; numerical convolution computes the sum density for a non-standard case.

3.10.7 Interpretation

Convolutions underlie every sum-of-variables calculation in probability. The Gaussian CLT, at its core, says that sums of iid variables with finite variance converge (after standardisation) to the normal regardless of the component distributions.

3.10.8 Practical Tips

- For explicit convolutions of standard distributions, use the characteristic-function approach: sums of independent stable distributions are again stable.
- Numerical convolutions can be computed via FFT (`fft` in R) for efficiency on large grids.
- In discrete cases, direct summation often suffices: `outer()` followed by appropriate tabulation.
- Convolutions preserve location, scale, shape of stable families but generally destroy non-stable structure (skewness, kurtosis drift across convolutions).
- For CLT-like arguments, the speed of convergence to normal depends on the finite third moment (Berry-Esseen inequality).

3.10.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.10.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.11 Copulas: An Introduction

3.11.1 Introduction

A copula is a multivariate distribution with uniform marginals. It captures the dependence structure of a joint distribution independently of the marginals. Sklar's theorem says every joint distribution decomposes into its marginals plus a unique copula; this decomposition is the cleanest way to model dependence when the marginals are non-normal.

3.11.2 Prerequisites

Joint and marginal distributions; cumulative distribution function; basic multivariate probability.

3.11.3 Theory

Sklar's theorem (1959). For any joint CDF F_{XY} with marginals F_X, F_Y, there exists a copula $C : [0,1]^2 \to [0,1]$ such that

$$F_{XY}(x, y) = C(F_X(x), F_Y(y)).$$

If F_X, F_Y are continuous, C is unique. In the other direction, for any copula C and marginals F_X, F_Y, the composition defines a valid joint CDF.

Common bivariate copulas:

- **Gaussian copula**: induced by a bivariate normal with correlation ρ. Symmetric, no tail dependence.
- **t-copula**: from a bivariate t. Same symmetric structure as Gaussian but with heavier tail dependence.
- **Clayton copula**: asymmetric lower-tail dependence.
- **Gumbel copula**: asymmetric upper-tail dependence.
- **Frank copula**: symmetric, no tail dependence, but wider range of dependence than Gaussian.

Tail dependence: the lower and upper tail-dependence coefficients $\lambda_L = \lim_{u \to 0^+} C(u,u)/u$ and $\lambda_U = \lim_{u \to 1^-} (1 - 2u + C(u,u))/(1-u)$ quantify how strongly extreme values of one variable co-occur with extremes of the other.

The **probability integral transform** turns any continuous marginal into a uniform: $U = F_X(X) \sim \text{Uniform}(0,1)$. Copulas operate on the uniform scale and transform back to the original scale via F_X^{-1}.

3.11.4 Assumptions

Continuous marginals (for uniqueness); otherwise the copula is unique only up to the joint probabilities of atoms.

3.11.5 R Implementation

```r
library(copula)

# Fit a Gaussian copula to bivariate data
set.seed(2026)
n <- 1000

# Simulate data with non-normal marginals but Gaussian copula
cop <- normalCopula(param = 0.7, dim = 2)
mvd <- mvdc(copula = cop,
            margins = c("gamma", "exp"),
            paramMargins = list(list(shape = 3, rate = 0.8),
                                list(rate = 0.3)))
X <- rMvdc(n, mvd)

# Visualise the dependence through the copula (uniform-scaled)
U <- pobs(X)   # empirical pseudo-observations on [0, 1]^2
plot(U, pch = 16, cex = 0.3, col = "#2A9D8F",
     main = "Empirical copula (pseudo-observations)",
     xlab = "U1", ylab = "U2")

# Fit a Gaussian copula
gc_fit <- fitCopula(normalCopula(dim = 2), data = U, method = "ml")
coef(gc_fit)

# Compare Clayton (lower tail dep) vs Gumbel (upper tail dep)
cc_fit <- fitCopula(claytonCopula(dim = 2), data = U, method = "ml")
gu_fit <- fitCopula(gumbelCopula(dim = 2), data = U, method = "ml")
c(gaussian = AIC(gc_fit), clayton = AIC(cc_fit), gumbel = AIC(gu_fit))
```

3.11.6 Output & Results

```
[1] 0.7036       # MLE of copula correlation recovers the true 0.7

gaussian  clayton  gumbel
  -822.1   -551.0  -683.8
```

Gaussian copula has the best AIC, matching the data-generating model. The Clayton and Gumbel copulas mis-specify the tail structure and fit worse.

3.11.7 Interpretation

Copula models are standard in finance (risk aggregation) and increasingly common in epidemiology (joint survival) and genetics (multivariate phenotypes). They let analysts specify marginals appropriate to each variable (gamma for concentrations, Beta for proportions) without forcing joint normality.

3.11.8 Practical Tips

- Do not fit copulas without first checking that the marginals are well-specified; a bad marginal confounds copula diagnosis.
- The Gaussian copula underestimates joint extremes; for risk applications use a t-copula or Archimedean copulas (Clayton, Gumbel).
- Empirical copulas are good exploratory tools; fit parametric copulas only after inspecting the pseudo-observation scatter.
- Multivariate copulas beyond dimension 3 are often built via vine copulas (`VineCopula` package); direct multivariate Archimedean copulas scale poorly.
- Copulas capture dependence, not marginal distributions; do not confuse copula correlation with Pearson correlation of the raw variables.

3.11.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.11.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.12 The Correlation Coefficient

3.12.1 Introduction

The correlation coefficient rescales covariance into a unitless number between -1 and $+1$, making comparisons across different measurement scales meaningful. A correlation of $+1$ means perfect positive linear dependence, -1 perfect negative, and 0 no linear association. It is the single most reported bivariate summary in applied research.

3.12.2 Prerequisites

Variance and covariance.

3.12.3 Theory

For random variables X, Y with finite non-zero variances,

$$\rho(X, Y) = \frac{\mathrm{Cov}(X, Y)}{\sigma_X \sigma_Y}.$$

Cauchy-Schwarz guarantees $\rho \in [-1, 1]$. Equality at ± 1 occurs if and only if $Y = aX + b$ almost surely with $\mathrm{sign}(a) = \mathrm{sign}(\rho)$.

The sample correlation coefficient (Pearson's r) is the plug-in estimator using sample covariance and SDs:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}.$$

Key caveats:

- $\rho = 0$ does **not** imply independence unless (X, Y) is jointly normal.
- ρ measures only the *linear* component of association; non-linear dependence is invisible to it.
- Outliers affect r disproportionately; a robust alternative is the Spearman rank correlation.

3.12.4 Assumptions

- Both variances finite and non-zero.
- Linearity is the interpretive assumption: ρ captures only linear dependence.

3.12.5 R Implementation

```r
library(mvtnorm)
set.seed(2026)

# Simulate bivariate normals at several correlation levels
rho_grid <- c(-0.8, -0.3, 0, 0.3, 0.8)
n <- 500
results <- sapply(rho_grid, function(rho) {
  Sigma <- matrix(c(1, rho, rho, 1), 2, 2)
  XY <- rmvnorm(n, sigma = Sigma)
  cor(XY[, 1], XY[, 2])
```

```
})
data.frame(true = rho_grid, empirical = round(results, 3))

# Zero correlation does not imply independence
X <- rnorm(1e4)
Y <- X^2 - 1      # quadratic, mean zero
cor(X, Y)

# Outlier sensitivity
X <- rnorm(100); Y <- 0.3 * X + rnorm(100)
c(no_outlier = cor(X, Y),
  with_outlier = cor(c(X, 20), c(Y, -20)))
```

3.12.6 Output & Results

```
  true  empirical
1 -0.8     -0.790
2 -0.3     -0.307
3  0.0     -0.023
4  0.3      0.309
5  0.8      0.815

[1] 0.005          # zero Pearson despite clear dependence

no_outlier with_outlier
     0.309       -0.059
```

Sample correlations recover the population values within Monte Carlo error. The $X, X^2 - 1$ example shows zero correlation with obvious dependence. A single outlying point flips the sign of a small-sample correlation.

3.12.7 Interpretation

Reported in a paper as "r = .68 (95% CI .56-.77), $p < .001$", the correlation conveys both direction and magnitude of linear association. Always accompany r with a scatter plot to verify linearity; never report r alone.

3.12.8 Practical Tips

- Use Pearson only for roughly linear relationships between continuous variables.
- For monotone non-linear association, use Spearman's rho or Kendall's tau.
- A 95% CI for ρ uses the Fisher z-transformation: $\operatorname{atanh}(r)$ has an approximately normal sampling distribution.
- Correlation does not imply causation; confounders can create strong associations between unrelated variables.

- In multivariate data, a single correlation matrix can be visualised with `corrplot::corrplot()` or `ggcorrplot::ggcorrplot()`; always plot the matrix and the pairwise scatters.

3.12.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.12.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.13 The Cumulative Distribution Function

3.13.1 Introduction

The cumulative distribution function (CDF) is the one representation that fits every random variable, whether discrete, continuous, or mixed. It encodes the distribution completely and supports every computation that a PMF or PDF does.

3.13.2 Prerequisites

Random variables, basic probability.

3.13.3 Theory

The CDF of a random variable X is

$$F_X(x) = P(X \leq x), \qquad x \in \mathbb{R}.$$

Every CDF has the following properties:

1. **Monotonic non-decreasing**: $x_1 \leq x_2 \Rightarrow F_X(x_1) \leq F_X(x_2)$.
2. **Right-continuous**: $\lim_{x \downarrow x_0} F_X(x) = F_X(x_0)$.
3. **Limits**: $\lim_{x \to -\infty} F_X(x) = 0$, $\lim_{x \to +\infty} F_X(x) = 1$.

Conversely, any function with these three properties defines a unique probability distribution on $\mathbb{R}$.

Key computations:

- Interval probability: $P(a < X \leq b) = F_X(b) - F_X(a)$.
- PDF (continuous X): $f_X(x) = F_X'(x)$.
- PMF (discrete X): $p_X(x) = F_X(x) - F_X(x^-)$, the jump size.
- Quantile function: $Q_X(u) = \inf\{x : F_X(x) \geq u\}$, the (generalised) inverse of F_X.

In R, each distribution family has a `p*()` function returning the CDF; the quantile function is `q*()`. The empirical CDF is obtained with `ecdf()`.

3.13.4 Assumptions

None beyond a well-defined probability measure.

3.13.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
x <- rnorm(200, mean = 75, sd = 10)

# Empirical CDF compared to theoretical CDF
Fn <- ecdf(x)
grid <- seq(50, 100, length.out = 401)
df <- data.frame(
  x        = grid,
  F_empir  = Fn(grid),
  F_theor  = pnorm(grid, mean = 75, sd = 10)
)

ggplot(df, aes(x = x)) +
  geom_step(aes(y = F_empir), colour = "#2A9D8F") +
  geom_line(aes(y = F_theor), colour = "#F4A261", linewidth = 1) +
  labs(y = "CDF", title = "Empirical (step) vs theoretical (line) CDF") +
  theme_minimal()

# Interval probability via the CDF
P_70_to_80 <- pnorm(80, 75, 10) - pnorm(70, 75, 10)
P_70_to_80

# Quantile: the value below which a given probability lies
qnorm(0.975, 75, 10)
```

3.13.6 Output & Results

```
[1] 0.3829        # P(70 < X <= 80)
[1] 94.6          # 97.5th percentile
```

The empirical step-function is close to the theoretical normal CDF, and the differences narrow as n grows (Glivenko-Cantelli).

3.13.7 Interpretation

In a paper, the CDF appears most often as a figure showing the empirical CDF against a theoretical reference (Q-Q plot is a rotation of this idea). Interval probabilities computed from the CDF are the standard form of reporting "percentage within the normal range".

3.13.8 Practical Tips

- The CDF is the unique object that characterises the distribution; two variables with the same CDF have the same distribution.
- Right-continuity vs left-continuity matters only at discrete atoms; the convention is right-continuous in most modern texts.
- The quantile function inverts the CDF; for continuous X, it is the ordinary functional inverse.
- The inverse-CDF method generates random draws: $X = F^{-1}(U)$ where $U \sim \text{Uniform}(0, 1)$.
- For mixed distributions, the CDF is the most transparent representation; PDFs and PMFs require generalised definitions.

3.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.13.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.14 Discrete vs Continuous Variables

3.14.1 Introduction

Random variables come in two canonical kinds – **discrete** (countable support) and **continuous** (uncountable support with a density). Most textbook distributions belong to one or the other. A subtler third category, **mixed**, combines discrete atoms with continuous parts and arises naturally in censored, truncated, or zero-inflated data.

3.14.2 Prerequisites

Random variables, PMF, and PDF.

3.14.3 Theory

A random variable X is:

- **Discrete** if it takes values in a countable set $\{x_1, x_2, ...\}$. Its distribution is described by the **probability mass function** $p(x_i) = P(X = x_i)$, with $\sum_i p(x_i) = 1$.
- **Continuous** (absolutely continuous) if there exists a **probability density function** f such that $P(X \in A) = \int_A f(x)\,dx$ for every measurable set A. For continuous X, $P(X = x) = 0$ for every single point x.
- **Mixed** if its distribution has both discrete atoms (points with positive probability mass) and a continuous part. Examples: censored survival times (mass at the censoring time, density elsewhere); zero-inflated counts (mass at 0, positive distribution above).

The CDF $F_X(x) = P(X \leq x)$ is defined for all three. For discrete X, F is a step function; for continuous X, F is smooth; for mixed X, F is piecewise smooth with jumps at the atoms.

The mean and variance are defined the same way (as $E[X]$ and $E[(X - E[X])^2]$) across the three cases, computed via sums for discrete and integrals for continuous.

3.14.4 Assumptions

The classification is mathematical rather than empirical; *any* random variable falls into one of the three categories.

3.14.5 R Implementation

```r
library(ggplot2)
set.seed(2026)

# Discrete: Poisson counts
dis <- rpois(5000, lambda = 3)
table(dis)[1:7]

# Continuous: exponential waiting times
con <- rexp(5000, rate = 0.5)
quantile(con, c(0.25, 0.5, 0.75))

# Mixed: censored normal (values below 0 set to exactly 0)
mix <- pmax(rnorm(5000, mean = 1.0, sd = 1.5), 0)
```

```r
c(mass_at_zero = mean(mix == 0),
  mean_overall = mean(mix))

# Plot all three
df <- rbind(
  data.frame(x = dis,    kind = "discrete (Poisson)"),
  data.frame(x = con,    kind = "continuous (exponential)"),
  data.frame(x = mix,    kind = "mixed (censored normal)")
)

ggplot(df, aes(x = x)) +
  geom_histogram(bins = 40, fill = "#2A9D8F") +
  facet_wrap(~ kind, scales = "free", ncol = 1) +
  theme_minimal()
```

3.14.6 Output & Results

```
dis       0      1      2      3      4      5      6
count   243    746   1124   1128    835    505    275

      25%       50%       75%
 0.5675    1.3735    2.7711

mass_at_zero  mean_overall
      0.2526        1.1342
```

The Poisson histogram has bars at integer values (discrete), the exponential has a smooth decaying curve (continuous), and the censored-normal has a spike at zero plus a continuous upper tail (mixed).

3.14.7 Interpretation

In reporting, the kind of random variable determines appropriate summaries:

- Discrete: report counts, proportions, mode, or a PMF table.
- Continuous: report mean, SD, quantiles, or a histogram/density.
- Mixed: report the atom probability separately from the continuous-part summary.

A paper analysing a zero-inflated outcome should acknowledge the mixed distribution explicitly rather than forcing it into a continuous or discrete mould.

3.14.8 Practical Tips

- Count data that "cap out" at a large value are often mixed; report the probability of the cap and analyse the remainder conditionally.

- Censored survival times are mixed; classical survival methods (Kaplan-Meier, Cox) handle the mixture natively.
- Continuous variables that are rounded to, say, integer values are still "almost continuous" in practice; treat as continuous for inference.
- Zero-inflated Poisson (`pscl::zeroinfl`) is the canonical regression model for mixed count data.
- When in doubt, plot a histogram (or ECDF) and inspect for atoms; the presence of a spike at a specific value signals a mixed distribution.

3.14.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.14.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.15 Expectation of a Random Variable

3.15.1 Introduction

The **expectation** $E[X]$ is the probability-weighted average of the values a random variable can take. It is the centre of gravity of the distribution, the long-run average of repeated independent draws, and the first moment in the moment hierarchy. Along with variance, it is the single most commonly reported summary of a distribution.

3.15.2 Prerequisites

Random variables, PMF or PDF, basic summation and integration.

3.15.3 Theory

For a discrete random variable with PMF p_X:

$$E[X] = \sum_x x\, p_X(x).$$

For a continuous random variable with PDF f_X:

$$E[X] = \int x\, f_X(x)\, dx.$$

Law of the Unconscious Statistician (LOTUS): for a measurable function g,

$$E[g(X)] = \sum_x g(x) p_X(x) \quad \text{or} \quad \int g(x) f_X(x)\, dx,$$

without needing to find the distribution of $g(X)$ first.

Linearity. Expectation is linear regardless of dependence:

$$E[aX + bY] = aE[X] + bE[Y], \qquad E\left[\sum_i X_i\right] = \sum_i E[X_i].$$

Product rule for independent variables: $E[XY] = E[X]E[Y]$ when $X \perp Y$.

Monotonicity: $X \leq Y$ (pointwise) $\Rightarrow E[X] \leq E[Y]$.

Expectation need not exist. The Cauchy distribution has no finite mean because the integral diverges. Existence requires $E[|X|] < \infty$.

3.15.4 Assumptions

- $E[|X|] < \infty$ for the expectation to be well-defined.
- Fubini/Tonelli conditions for interchanging integration and summation when needed (automatic for non-negative variables and for integrable ones).

3.15.5 R Implementation

```r
set.seed(2026)

# Exact expectation for discrete RV: binomial
n <- 10; p <- 0.3
k <- 0:n
E_X <- sum(k * dbinom(k, n, p))
c(formula = E_X, closed_form = n * p)

# Exact expectation for continuous RV: gamma
integrate(function(x) x * dgamma(x, shape = 3, rate = 0.5), 0, Inf)
3 / 0.5   # closed form alpha/beta

# LOTUS: E[g(X)] without the distribution of g(X)
```

```r
# For X ~ Gamma(3, 0.5), compute E[log(X)]
integrate(function(x) log(x) * dgamma(x, shape = 3, rate = 0.5), 0, Inf)

# Monte Carlo estimate (LLN justifies this)
X <- rgamma(1e5, shape = 3, rate = 0.5)
mean(log(X))

# Linearity check: E[2X + 3Y] = 2E[X] + 3E[Y] regardless of dependence
X <- rnorm(1e5, 5, 2)
Y <- 0.7 * X + rnorm(1e5, 0, 1)        # dependent
c(direct = mean(2 * X + 3 * Y),
  sum_of_parts = 2 * mean(X) + 3 * mean(Y))
```

3.15.6 Output & Results

```
   formula  closed_form
       3.0          3.0

6 with absolute error < 8.5e-05

[1] 1.506         # E[log X] via integration
[1] 1.505         # same via Monte Carlo

     direct  sum_of_parts
     27.43         27.43
```

Closed-form, numerical integration, and Monte Carlo all agree, and linearity of expectation holds despite dependence between X and Y.

3.15.7 Interpretation

When a paper reports a mean, it is (implicitly) reporting an estimate of the expectation of a random variable. The sample mean converges to the expectation as n grows (law of large numbers), so inferences about population-level expectations derive their validity from this convergence.

3.15.8 Practical Tips

- Linearity is unconditional on dependence; this is the single most useful fact about expectation. It underlies the unbiasedness of the sample mean regardless of the correlation structure.
- The product rule requires independence: $E[XY] = E[X]E[Y]$ fails when X, Y are dependent.
- Jensen's inequality: $E[g(X)] \geq g(E[X])$ for convex g (reversed for concave). Taking a log of an expectation is not the expectation of the log.

- For heavy-tailed distributions (Cauchy, stable with index < 1), the mean may not exist; rely on medians instead.
- Monte Carlo estimation of expectations works for any integrable X; convergence rate is $O(n^{-1/2})$ (CLT).

3.15.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.15.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.16 The Exponential Distribution

3.16.1 Introduction

The exponential distribution models waiting times between events in a Poisson process with constant rate. It is the only continuous distribution with the memoryless property, and it is the simplest parametric model in survival and reliability analysis.

3.16.2 Prerequisites

Continuous random variables and the Poisson process.

3.16.3 Theory

A non-negative random variable X is **exponential with rate** $\lambda > 0$ if

$$f(x) = \lambda e^{-\lambda x}, \qquad x \geq 0.$$

CDF: $F(x) = 1 - e^{-\lambda x}$. Survival function: $S(x) = e^{-\lambda x}$.

Moments:

- $E[X] = 1/\lambda$ (the mean waiting time).
- $\mathrm{Var}(X) = 1/\lambda^2$.

Memoryless property: $P(X > s + t \mid X > s) = P(X > t)$. The probability of waiting another t units, given we have already waited s, is the same as the unconditional probability of waiting t.

Constant hazard: $h(x) = f(x)/S(x) = \lambda$. The instantaneous event rate does not depend on how long the object has already survived.

Relationship to Poisson: in a Poisson process with rate λ, inter-arrival times are iid exponential with rate λ.

Scale vs rate: some R functions use rate (λ), others use scale ($1/\lambda$). Check the documentation.

3.16.4 Assumptions

Constant hazard (no ageing / no improvement). Violations motivate the Weibull or log-normal distributions.

3.16.5 R Implementation

```r
lambda <- 0.5

X <- rexp(1e5, rate = lambda)
c(mean = mean(X), theoretical = 1 / lambda,
  var  = var(X),  theoretical_var = 1 / lambda^2)

# Memoryless property
P_past_2 <- mean(X > 2)
P_past_5_given_2 <- mean(X[X > 2] > 5)        # P(X > 5 | X > 2)
P_past_3 <- mean(X > 3)                        # should equal above by
 ↪ memorylessness
c(conditional = P_past_5_given_2, unconditional = P_past_3)

# Constant hazard
breaks <- seq(0, 10, by = 1)
hazards <- sapply(seq_len(length(breaks) - 1), function(i) {
  in_bin <- X >= breaks[i] & X < breaks[i + 1]
  sum(in_bin) / sum(X >= breaks[i])
})
round(hazards, 3)
```

3.16.6 Output & Results

```
    mean theoretical      var  theoretical_var
    1.996        2.0    3.985              4.0

  conditional  unconditional
```

```
      0.2232         0.2229

 [1] 0.394 0.392 0.392 0.400 0.391 0.395 0.394 0.389 0.399 0.394
```

Empirical mean and variance match theoretical; memoryless property holds; empirical hazard is constant at $\approx 1 - e^{-\lambda}$ per unit interval.

3.16.7 Interpretation

Exponential survival is the simplest hypothesis one can test in reliability and survival. When observed hazard is non-constant, report the Weibull or a flexible parametric model instead.

3.16.8 Practical Tips

- `rexp(n,rate)` in R uses the rate parameterisation; `scale=1/rate`.
- Log-transform to compress the heavy right tail for visualisation.
- Constant hazard is a strong and testable assumption; plot $-\log S(t)$ against t; linearity indicates exponentiality.
- For right-censored data, use survival-analysis tools (`survreg(...,dist="exponential")`) rather than naive `rexp`.
- Exponential is the discrete geometric's continuous analogue; both share memorylessness.

3.16.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.16.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.17 The F Distribution

3.17.1 Introduction

The F distribution arises as the ratio of two independent scaled chi-squared random variables. It is the reference distribution for the F-statistic in ANOVA, for tests of variance equality between two normal samples, and for omnibus tests

in linear regression. Its shape is controlled by two degree-of-freedom parameters, one for the numerator and one for the denominator.

3.17.2 Prerequisites

Chi-squared distribution.

3.17.3 Theory

If $U \sim \chi^2_{d_1}$ and $V \sim \chi^2_{d_2}$ are independent, then

$$F = \frac{U/d_1}{V/d_2} \sim F_{d_1,d_2}.$$

The numerator df is d_1; the denominator df is d_2.

Moments (when they exist):

- $E[F] = d_2/(d_2 - 2)$ for $d_2 > 2$.
- $\mathrm{Var}(F) = \frac{2d_2^2(d_1+d_2-2)}{d_1(d_2-2)^2(d_2-4)}$ for $d_2 > 4$.

Relationships to other distributions:

- $1/F_{d_1,d_2} \sim F_{d_2,d_1}$.
- $t_\nu^2 \sim F_{1,\nu}$.
- As $d_2 \to \infty$, $d_1 \cdot F_{d_1,d_2} \to \chi^2_{d_1}$.

One-way ANOVA: with k groups and n total observations, the F-statistic

$$F = \frac{\mathrm{MS}_{\text{between}}}{\mathrm{MS}_{\text{within}}}$$

follows $F_{k-1,n-k}$ under the null of equal means.

Variance ratio test: for two independent normal samples with variances $\sigma_1^2 = \sigma_2^2$,

$$F = s_1^2/s_2^2 \sim F_{n_1-1,n_2-1}.$$

3.17.4 Assumptions

Independence between U and V; normality of the underlying data for the derived sample statistics.

3.17.5 R Implementation

```r
# PDF visualisation
x <- seq(0, 5, length.out = 400)
plot(x, df(x, 5, 20), type = "l", col = "#2A9D8F", lwd = 2,
     main = "F(5, 20) density", ylab = "f(x)")
lines(x, df(x, 5, 5),   col = "#F4A261", lwd = 2)
lines(x, df(x, 20, 20), col = "#6A4C93", lwd = 2)
legend("topright", c("F(5,20)", "F(5,5)", "F(20,20)"),
       col = c("#2A9D8F", "#F4A261", "#6A4C93"), lwd = 2)

# Verify construction
n <- 1e5; d1 <- 4; d2 <- 10
U <- rchisq(n, df = d1); V <- rchisq(n, df = d2)
F_sim <- (U/d1) / (V/d2)
c(emp_mean = mean(F_sim), theoretical = d2 / (d2 - 2))

# F-statistic from ANOVA
set.seed(2026)
df <- data.frame(y = c(rnorm(30, 50, 8), rnorm(30, 55, 8), rnorm(30, 53,
 ↪  8)),
                 g = factor(rep(c("A", "B", "C"), each = 30)))
anova(lm(y ~ g, data = df))
```

3.17.6 Output & Results

```
emp_mean theoretical
    1.252      1.250

Analysis of Variance Table

Response: y
          Df Sum Sq Mean Sq F value  Pr(>F)
g          2  394.7   197.3    3.57 0.03218 *
Residuals 87 4811.2    55.3
```

The empirical F ratio recovers the theoretical mean, and the ANOVA F-statistic of 3.57 on (2, 87) df has p-value 0.032.

3.17.7 Interpretation

Almost every ANOVA and linear-model-comparison result in applied statistics is reported as an F-statistic with numerator and denominator df. Reporting "$F(2, 87) = 3.57$, $p = 0.032$" is a standard form.

3.17.8 Practical Tips

- Order the df correctly: F(df1, df2) is not the same as F(df2, df1) in general.

- For two-sided variance tests, some software uses max/min of s_1^2/s_2^2 to avoid ambiguity.
- Non-central F (with non-centrality parameter λ) governs power calculations for ANOVA.
- The F-distribution is highly sensitive to non-normality; for skewed data prefer Welch's ANOVA or non-parametric alternatives.
- `qf(0.95,d1,d2)` gives the critical value for a one-sided 5% test.

3.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.17.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.18 The Gamma Distribution

3.18.1 Introduction

The gamma distribution is a flexible right-skewed distribution on positive real numbers. It is parameterised by a shape and a rate (or scale) and includes the exponential, Erlang, and chi-squared as special cases. In applied work it models waiting times, concentrations, and response amplitudes – any positive continuous variable with light-to-moderate right skew.

3.18.2 Prerequisites

Exponential distribution, basic calculus.

3.18.3 Theory

A gamma random variable with shape $\alpha > 0$ and rate $\beta > 0$ has PDF

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, \qquad x > 0.$$

Scale parameterisation: $\theta = 1/\beta$, so $f(x) = \frac{1}{\theta^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{-x/\theta}$.

Moments:

- $E[X] = \alpha/\beta$ (rate form) $= \alpha\theta$ (scale form).
- $\text{Var}(X) = \alpha/\beta^2 = \alpha\theta^2$.

Special cases:

- Shape $\alpha = 1$: exponential(β).
- Integer shape $\alpha = k$: Erlang distribution, which equals the sum of k iid exponentials with rate β.
- Shape $\alpha = \nu/2$, rate $\beta = 1/2$: chi-squared with ν df.

Sum property: independent Gamma(α_1, β) and Gamma(α_2, β) sum to Gamma($\alpha_1 + \alpha_2, \beta$).

Conjugacy: gamma is conjugate to the Poisson likelihood (prior on λ); gamma-Poisson mixture yields negative binomial.

3.18.4 Assumptions

Values must be positive. The flexibility comes from the two parameters; monotone hazard (increasing for $\alpha > 1$, decreasing for $\alpha < 1$, constant for $\alpha = 1$).

3.18.5 R Implementation

```r
library(MASS)

shape <- 2.5; rate <- 0.8

X <- rgamma(1e5, shape = shape, rate = rate)
c(mean = mean(X), theoretical_mean = shape / rate,
  var  = var(X),  theoretical_var = shape / rate^2)

# Visualise the PDF
curve(dgamma(x, shape, rate), from = 0, to = 15,
      main = paste("Gamma(shape =", shape, ", rate =", rate, ")"),
      xlab = "x", ylab = "f(x)", col = "#2A9D8F", lwd = 2)

# MLE fit
fit <- fitdistr(X, "gamma")
fit$estimate

# Sum of independent exponentials equals Erlang (gamma with integer
↪  shape)
k <- 5; lambda <- 1
S <- replicate(5000, sum(rexp(k, rate = lambda)))
c(mean_empirical = mean(S), mean_theoretical = k / lambda,
  var_empirical  = var(S),  var_theoretical  = k / lambda^2)
```

3.18.6 Output & Results

```
  mean theoretical_mean      var theoretical_var
  3.13             3.125     3.92            3.906

    shape    rate
  2.507   0.802       # MLE recovers true parameters

mean_empirical mean_theoretical  var_empirical  var_theoretical
         4.989            5.000          5.014            5.000
```

Empirical moments match theoretical; MLE recovers the true parameters; the sum of 5 iid exponentials has the predicted Erlang moments.

3.18.7 Interpretation

Gamma is a default parametric model for positive continuous variables that do not fit well to normal or log-normal. In generalised linear models, a gamma family (`glm(...,family=Gamma)`) is appropriate for positive continuous outcomes with variance proportional to the squared mean.

3.18.8 Practical Tips

- Watch R's `rgamma(n,shape,rate)` vs `rgamma(n,shape,scale=...)`; only one of `rate` and `scale` should be given.
- Fitting a gamma to data: `MASS::fitdistr(x,"gamma")` or `fitdistrplus::fitdist`.
- Small shape ($\alpha < 1$) gives a distribution with a pole at zero; reasonable for heavily skewed data but breaks with zero values in a sample.
- For modelling variance in mixed models, the gamma distribution appears as the distribution of random-effect variances in certain hierarchical setups.
- Log-gamma (log of a gamma) is often approximately normal and easier to work with for some computations.

3.18.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.18.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson

- Continuous distributions and Q-Q plots

3.19 The Geometric Distribution

3.19.1 Introduction

The geometric distribution counts the number of Bernoulli trials required to obtain the first success. It is the discrete analogue of the exponential waiting-time distribution and shares the memoryless property: the probability of further trials to success does not depend on how many have already been tried.

3.19.2 Prerequisites

Bernoulli trials.

3.19.3 Theory

Two conventions exist. Let X be the number of Bernoulli(p) trials *up to and including* the first success:

$$P(X = k) = (1-p)^{k-1}p, \qquad k = 1, 2, 3, \ldots$$

Alternatively, Y is the number of failures *before* the first success:

$$P(Y = k) = (1-p)^{k}p, \qquad k = 0, 1, 2, \ldots$$

R uses the second convention (`rgeom`, `dgeom` return Y, i.e., failures before first success).

Moments for the first convention (X = trials until first success):

- $E[X] = 1/p$, $\mathrm{Var}(X) = (1-p)/p^2$.

For Y (R's convention): $E[Y] = (1-p)/p$, $\mathrm{Var}(Y) = (1-p)/p^2$.

Memoryless property: $P(X > m + n \mid X > m) = P(X > n)$. Having waited m trials without success does not change the distribution of further waiting time.

3.19.4 Assumptions

Independent Bernoulli trials with constant probability p.

3.19.5 R Implementation

```r
p <- 0.3

# R's convention: Y = failures before first success
Y <- rgeom(1e5, p)
c(mean_Y = mean(Y), theoretical = (1 - p) / p)

# Number of trials to first success X = Y + 1
X <- Y + 1
c(mean_X = mean(X), theoretical = 1 / p)

# PMF comparison
k <- 0:15
data.frame(k, pmf_Y = dgeom(k, p))

# Memoryless: P(Y > 5 + 3 | Y > 5) = P(Y > 3)
Y_past_5 <- Y[Y > 5]
c(conditional = mean(Y_past_5 > 5 + 3),
  unconditional = mean(Y > 3))
```

3.19.6 Output & Results

```
   mean_Y theoretical
    2.334       2.333

   mean_X theoretical
    3.334       3.333

    k      pmf_Y
1   0     0.3000
2   1     0.2100
3   2     0.1470
...

conditional  unconditional
     0.3441         0.3430
```

The memoryless property holds empirically: conditional probability of further 3 failures, given already 5 failures, equals the unconditional probability of at least 3 failures.

3.19.7 Interpretation

The geometric distribution is the baseline model for "time to event" in the discrete case: cycles to pregnancy in IVF, attempts before a successful procedure, trials before a rare positive test. Memorylessness is a strong assumption that often fails in real data where fatigue or learning changes p over time.

3.19.8 Practical Tips

- Check which convention (trials vs. failures) is expected when comparing to textbook formulas.
- Memorylessness is testable empirically: compare the distribution of residual trials given survival to each time.
- For non-memoryless discrete waiting times, the negative binomial (shape > 1) captures over-dispersion relative to geometric.
- The geometric converges to the exponential when discretising time with fine intervals.
- In clinical or biological settings, censoring is common; naive `rgeom` ignores right-censoring – use survival methods for realistic data.

3.19.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.19.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.20 The Hazard Function

3.20.1 Introduction

The hazard function $h(t)$ gives the instantaneous rate at which events occur at time t, given that they have not occurred before t. It is the primary object in survival analysis and reliability engineering: proportional-hazards models, accelerated-failure-time models, and competing-risks methods all reason about hazards rather than distributions directly.

3.20.2 Prerequisites

Continuous distributions; survival function $S(t) = P(T > t)$.

3.20.3 Theory

For a non-negative continuous random variable T with density f and survival function $S(t) = 1 - F(t)$,

$$h(t) = \frac{f(t)}{S(t)}.$$

Intuitively, $h(t)\,dt$ is approximately $P(t \le T < t + dt \mid T \ge t)$ for small dt.

The **cumulative hazard** is

$$H(t) = \int_0^t h(u)\,du = -\log S(t).$$

Conversely, $S(t) = e^{-H(t)}$. Once either of f, F, S, h, H is known, the others follow.

Hazard shapes for common distributions:

- Exponential: $h(t) = \lambda$ (constant).
- Weibull with shape k: $h(t) = (k/\lambda)(t/\lambda)^{k-1}$ – increasing if $k > 1$, decreasing if $k < 1$.
- Gompertz: $h(t) = \alpha e^{\beta t}$ – exponentially increasing; common in human mortality.
- Log-normal: $h(t)$ rises then falls.

3.20.4 Assumptions

T is a non-negative continuous random variable with a density. Proportional-hazards models further assume $h(t \mid x) = h_0(t)\exp(\beta^\top x)$.

3.20.5 R Implementation

```
library(survival); library(ggplot2)

# Constant hazard (exponential)
x <- seq(0.01, 5, length.out = 400)
h_exp <- rep(0.5, length(x))
h_weibull_inc <- (2.0 / 1.5) * (x / 1.5)^(2.0 - 1)
h_weibull_dec <- (0.5 / 1.5) * (x / 1.5)^(0.5 - 1)

df <- data.frame(
  t  = rep(x, 3),
  h  = c(h_exp, h_weibull_inc, h_weibull_dec),
  kind = factor(rep(c("exp lambda=0.5",
                      "weibull k=2 (incr)",
                      "weibull k=0.5 (decr)"), each = length(x)))
)

ggplot(df, aes(t, h, colour = kind)) +
```

```
  geom_line(linewidth = 1) +
  scale_colour_manual(values = c("#6A4C93", "#2A9D8F", "#F4A261")) +
  labs(x = "t", y = "h(t)", colour = "",
       title = "Hazard functions for three canonical models") +
  theme_minimal()

# Empirical cumulative hazard from censored data
set.seed(2026)
T <- rexp(100, rate = 0.3)
status <- rep(1, 100)
fit <- survfit(Surv(T, status) ~ 1)
plot(fit, fun = "cumhaz", col = "#2A9D8F", lwd = 2,
     xlab = "t", ylab = "H(t)",
     main = "Empirical cumulative hazard (Nelson-Aalen)")
abline(0, 0.3, col = "#F4A261", lty = 2)
```

3.20.6 Output & Results

The plot shows three hazards: exponential (flat), increasing Weibull, decreasing Weibull. The empirical cumulative hazard from 100 exponential event times is approximately linear with slope $\lambda = 0.3$, matching the theoretical $H(t) = 0.3t$.

3.20.7 Interpretation

In survival analysis, hazards are the natural target of regression: "the hazard of death was 40 % higher in the treated group than in controls (HR = 1.40, 95 % CI 1.10-1.78)". The proportional-hazards model is ubiquitous because it avoids specifying the baseline hazard shape.

3.20.8 Practical Tips

- A constant hazard implies exponentiality; a log-log plot of $-\log S(t)$ against $\log t$ that is linear indicates Weibull.
- Empirical hazards are noisy at the tail where few subjects remain; report cumulative hazard (smoother) or kernel-smoothed hazard estimates.
- The **Nelson-Aalen** estimator is the standard non-parametric estimator of the cumulative hazard under right-censoring.
- In competing risks, cause-specific hazards differ from the sub-distribution hazard of Fine-Gray; the two answer different questions.
- Time-varying hazard ratios invalidate the proportional-hazards assumption; use stratification or time-interactions.

3.20.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.20.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.21 The Hypergeometric Distribution

3.21.1 Introduction

The hypergeometric distribution models the number of successes in a sample drawn without replacement from a finite population. It differs from the binomial by dependence across draws: removing a success reduces the probability of a subsequent success. It underlies Fisher's exact test and audit sampling.

3.21.2 Prerequisites

Combinatorics, binomial distribution.

3.21.3 Theory

A population of N items contains K "successes" and $N-K$ "failures". A sample of size n is drawn without replacement. Let X be the number of successes in the sample:

$$P(X=k)=\frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}}, \qquad \max(0,n-(N-K))\le k\le \min(n,K).$$

Moments:

- $E[X]=nK/N$ (same as binomial with $p=K/N$).
- $\mathrm{Var}(X)=n\cdot\frac{K(N-K)}{N^2}\cdot\frac{N-n}{N-1}$.

The factor $(N-n)/(N-1)$ is the **finite-population correction**; it shrinks the variance relative to binomial sampling, reflecting the reduced uncertainty when a substantial fraction of the population is sampled.

As $N\to\infty$ with $K/N=p$ fixed, hypergeometric converges to binomial: when the population is large relative to the sample, with/without replacement makes little difference.

3.21.4 Assumptions

- Finite population of known size N.
- Known number of successes K in the population.
- Simple random sampling without replacement.

3.21.5 R Implementation

```r
N <- 200; K <- 60; n <- 30

# PMF and moments
k <- 0:n
pmf <- dhyper(k, K, N - K, n)
c(mean = n * K / N,
  var  = n * K / N * (1 - K / N) * (N - n) / (N - 1),
  emp_mean = sum(k * pmf),
  emp_var  = sum((k - sum(k * pmf))^2 * pmf))

# Compare with binomial: same mean, larger variance
c(binom_var = n * (K / N) * (1 - K / N))

# Fisher's exact test uses the hypergeometric
tab <- matrix(c(12, 18, 48, 122), 2, 2)
fisher.test(tab)
```

3.21.6 Output & Results

```
     mean       var   emp_mean    emp_var
      9.0     5.487       9.00      5.487

binom_var
     6.30

        Fisher's Exact Test for Count Data

data:  tab
p-value = 0.1412
alternative hypothesis: true odds ratio is not equal to 1
```

The hypergeometric variance is smaller than the binomial variance (5.49 vs 6.30) because the finite-population correction accounts for dependence. Fisher's exact test in the 2x2 example is a hypergeometric computation under the null of independence.

3.21.7 Interpretation

Hypergeometric inference appears whenever sampling without replacement matters. In quality audits, the exact probability of a given number of defective items being observed in a sample is hypergeometric. In contingency-table analysis of small cells, Fisher's exact test is hypergeometric.

3.21.8 Practical Tips

- For large populations ($N > 20n$), binomial approximation is typically adequate and simpler.
- For 2x2 contingency tables, Fisher's exact test is hypergeometric-based and preferable to chi-squared with small expected counts.
- `dhyper(x,m,n,k)` uses R's quirky naming: `m` = successes in population, `n` = failures in population, `k` = sample size.
- In survey sampling with stratification, the within-stratum variance uses a finite-population correction factor.
- Generalisations to multiple categories yield the multivariate hypergeometric; useful in categorical audit and card-game probabilities.

3.21.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.21.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.22 Independence

3.22.1 Introduction

Two events are *independent* when knowing that one occurred tells you nothing about whether the other will. Independence is the property that makes simulation tractable, factorises joint distributions, and underlies the iid assumption of almost every classical statistical method. It is also one of the most commonly mis-stated concepts: independent is not the same as disjoint, and pairwise independence does not imply joint independence.

3.22.2 Prerequisites

Basic probability, conditional probability, and the product rule.

3.22.3 Theory

Two events A and B are **independent** if

$$P(A \cap B) = P(A)\,P(B).$$

Equivalently, when $P(B) > 0$: $P(A \mid B) = P(A)$. Knowing B does not update the probability of A.

For n events $A_1, \dots, A_n$, two levels of independence exist:

- **Pairwise independence**: $P(A_i \cap A_j) = P(A_i)P(A_j)$ for every $i \neq j$.
- **Mutual independence**: every subset factorises. $P(A_{i_1} \cap \dots \cap A_{i_k}) = \prod_{j=1}^{k} P(A_{i_j})$ for every subset.

Mutual is strictly stronger: it is possible for three events to be pairwise independent yet not mutually independent (Bernstein's example).

For random variables X and Y:

$$X \perp Y \iff F_{XY}(x, y) = F_X(x)F_Y(y) \quad \forall x, y,$$

or equivalently (when densities exist) $f_{XY}(x, y) = f_X(x)f_Y(y)$.

Independence vs. uncorrelatedness. Independence implies zero correlation, but not vice versa: X and X^2 for $X \sim \mathcal{N}(0, 1)$ are uncorrelated yet functionally dependent.

3.22.4 Assumptions

None beyond the probability space being well-defined.

3.22.5 R Implementation

```r
set.seed(2026)

# Two independent Bernoulli trials
n <- 1e5
A <- rbinom(n, 1, 0.3)
B <- rbinom(n, 1, 0.6)

# Check: P(A and B) ≈ P(A) * P(B)
c(joint = mean(A == 1 & B == 1),
```

```
  product = mean(A == 1) * mean(B == 1))

# Dependence: construct two correlated Bernoullis
C <- ifelse(A == 1, rbinom(n, 1, 0.9), rbinom(n, 1, 0.2))
c(joint = mean(A == 1 & C == 1),
  product = mean(A == 1) * mean(C == 1))

# Uncorrelated but dependent example
X <- rnorm(n)
Y <- X^2
c(correlation = cor(X, Y),
  joint_factorises = abs(mean(X < 0 & Y > 2) -
                         mean(X < 0) * mean(Y > 2)))
```

3.22.6 Output & Results

```
  joint  product
 0.1801   0.1800

  joint  product
 0.2714   0.1798

correlation joint_factorises
     0.0028           0.1589
```

First pair is independent (joint equals product). Second pair is dependent (joint much larger than product). Third shows zero correlation with clearly non-zero deviation from factorisation.

3.22.7 Interpretation

In study design, independence is usually a *design* property: separate random assignments ensure treatment is independent of unmeasured confounders; random selection ensures the sample is independent of other observable sources. When the design does not guarantee independence (clustered data, repeated measures), the statistical model must account for it.

3.22.8 Practical Tips

- Independence is a property of a particular probability measure P. Two events can be independent under one distribution and dependent under another.
- Pairwise-but-not-mutual independence is rare in applied settings but matters in cryptography and combinatorial probability.
- "iid" (independent and identically distributed) is the default assumption of most statistical methods; violations require mixed-effects models, GEE,

or time-series methods.
- Use simulation to check independence when analytical factorisation is cumbersome; the deviation from $P(A)P(B)$ quantifies the dependence.
- For continuous variables, a formal independence test is available via distance correlation (`energy::dcor`).

3.22.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.22.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.23 Joint Distributions

3.23.1 Introduction

A joint distribution describes how two or more random variables behave together. It contains strictly more information than the marginals alone: the marginals tell you about each variable in isolation, while the joint tells you how they interact. Dependence, correlation, and conditional behaviour are all properties of the joint distribution.

3.23.2 Prerequisites

Random variables, PMF, PDF, independence.

3.23.3 Theory

For two random variables X and Y:

- **Joint CDF**: $F_{X,Y}(x, y) = P(X \leq x, Y \leq y)$.
- **Joint PMF** (discrete): $p_{X,Y}(x, y) = P(X = x, Y = y)$.
- **Joint PDF** (continuous): $f_{X,Y}(x, y)$ with $\iint f = 1$.

The joint CDF satisfies $\partial^2 F / \partial x \partial y = f$ (in the continuous case where derivatives exist).

Independence: $X \perp Y$ if and only if $f_{X,Y}(x, y) = f_X(x) f_Y(y)$ for all x, y (or the PMF equivalent). Equivalently, the joint CDF factorises.

The **bivariate normal** is the canonical joint distribution in applied work:

$$f_{X,Y}(x,y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}} \times \exp\!\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_X)^2}{\sigma_X^2} - \frac{2\rho(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y} + \frac{(y-\mu_Y)^2}{\sigma_Y^2}\right]\right\}.$$

Its marginals, conditionals, and any linear combination are all normal.

3.23.4 Assumptions

A joint distribution is well-defined whenever the random variables live on a common probability space.

3.23.5 R Implementation

```r
library(mvtnorm); library(ggplot2)
set.seed(2026)

# Bivariate normal sample
Sigma <- matrix(c(1, 0.6,
                  0.6, 1), 2, 2)
XY <- rmvnorm(2000, mean = c(0, 0), sigma = Sigma)
df <- data.frame(X = XY[, 1], Y = XY[, 2])

ggplot(df, aes(X, Y)) +
  geom_point(alpha = 0.3, colour = "#2A9D8F") +
  geom_density_2d(colour = "#F4A261") +
  labs(title = "Bivariate normal with correlation 0.6") +
  theme_minimal()

# Joint density at a point
dmvnorm(c(0, 0), mean = c(0, 0), sigma = Sigma)

# Probability of a rectangle via the CDF
p_rect <- pmvnorm(lower = c(-1, -1), upper = c(1, 1),
                  mean = c(0, 0), sigma = Sigma)
p_rect
```

3.23.6 Output & Results

```
[1] 0.1989       # density at (0,0)
```

```
[1] 0.6372       # P(-1 <= X <= 1, -1 <= Y <= 1)
```

The rectangle has probability 0.64 under the bivariate normal with correlation 0.6 – higher than the product of the marginal rectangle probabilities (0.68 x 0.68 = 0.47) because the variables are positively correlated.

3.23.7 Interpretation

Most multivariate methods assume some form of joint distribution. Multivariate normal is the default assumption in linear discriminant analysis, MANOVA, and Gaussian graphical models. When the joint distribution is clearly non-normal, consider copula models or non-parametric density estimation.

3.23.8 Practical Tips

- Visualise joint distributions with 2D density plots, hexbin plots, or pairs plots; scatter plots alone can miss cluster structure.
- The marginals never determine the joint distribution; two datasets with identical marginals can have very different bivariate structures (Anscombe's quartet generalisation).
- For continuous data, `MASS::kde2d` and `ggplot2::stat_density_2d` give kernel joint densities.
- Independence tests: for continuous data use Hoeffding's D (`Hmisc::hoeffd`) or distance correlation (`energy::dcor`); both detect non-linear dependence Pearson misses.
- Joint discrete distributions are usually tabulated as contingency tables; chi-squared tests of independence are the canonical hypothesis tests.

3.23.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.23.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.24 Kolmogorov's Axioms of Probability

3.24.1 Introduction

Modern probability theory rests on three axioms proposed by Andrey Kolmogorov in 1933. They turn probability from a collection of intuitions into a rigorous mathematical theory consistent with measure theory. Every theorem about probability – the law of large numbers, the central limit theorem, Bayes' rule – is ultimately a consequence of these three rules and the definitions that accompany them.

3.24.2 Prerequisites

The reader should know what a set is and should be comfortable with the basic set operations (union, intersection, complement).

3.24.3 Theory

A **probability space** is the triple $(\Omega, \mathcal{F}, P)$, where:

- Ω is the **sample space**, the set of all possible outcomes.
- $\mathcal{F}$ is a **sigma-algebra** of subsets of Ω – the collection of events to which we assign probabilities. It is closed under complements and countable unions.
- P is the **probability measure**, a function $P : \mathcal{F} \to [0, 1]$ satisfying Kolmogorov's three axioms:

1. **Non-negativity**: $P(A) \geq 0$ for every $A \in \mathcal{F}$.
2. **Normalisation**: $P(\Omega) = 1$.
3. **Countable additivity**: for any sequence of pairwise disjoint events $A_1, A_2, \ldots$ in $\mathcal{F}$, $P(\bigcup_i A_i) = \sum_i P(A_i)$.

From these axioms every familiar property follows:

- $P(\emptyset) = 0$.
- $P(A^c) = 1 - P(A)$ (complement rule).
- If $A \subseteq B$ then $P(A) \leq P(B)$ (monotonicity).
- $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ (inclusion-exclusion).

The axioms are the minimal, sufficient rules. Violating any of them produces a theory that is either inconsistent or fails to capture behaviour real data exhibit.

3.24.4 Assumptions

The axioms themselves are assumptions; they cannot be derived from more primitive principles. Their justification is empirical: they match the long-run frequencies observed in real random experiments.

3.24.5 R Implementation

We can verify the axioms empirically by simulation for a simple probability space.

```
set.seed(2026)

# Fair six-sided die: Omega = {1, ..., 6}
rolls <- sample(1:6, size = 1e5, replace = TRUE)

# Axiom 1: every event probability in [0, 1]
empirical_p <- prop.table(table(rolls))
all(empirical_p >= 0 & empirical_p <= 1)

# Axiom 2: probability of the whole sample space is 1
mean(rolls %in% 1:6)

# Axiom 3: additivity for disjoint events
A <- rolls %in% c(1, 2)   # "low" outcomes
B <- rolls %in% c(3, 4)   # "mid" outcomes
# A and B are disjoint; P(A or B) should equal P(A) + P(B)
c(p_union = mean(A | B),
  p_A_plus_p_B = mean(A) + mean(B))

# Derived property: complement rule
c(p_not_A = mean(!A),
  one_minus_p_A = 1 - mean(A))

# Inclusion-exclusion for non-disjoint events
C <- rolls %in% c(2, 3, 4)   # overlaps with A
c(p_union = mean(A | C),
  formula = mean(A) + mean(C) - mean(A & C))
```

3.24.6 Output & Results

```
[1] TRUE          # all empirical probabilities in [0, 1]

[1] 1             # P(Omega) = 1

  p_union p_A_plus_p_B
   0.6661       0.6661

  p_not_A one_minus_p_A
   0.6669        0.6669

  p_union  formula
   0.6669   0.6669
```

Empirical frequencies converge to axiomatic probabilities as n grows; the axioms are satisfied by the simulated frequencies.

3.24.7 Interpretation

The axioms are not something one "reports" in a paper; they are the framework within which every probabilistic claim is made. When a manuscript states $P(\text{event}) = 0.23$, it assumes a probability space has been defined implicitly and that P is a valid measure on it.

3.24.8 Practical Tips

- In applied work the sample space is usually obvious (outcomes of a trial, categories of a classification), and the sigma-algebra is the full power set. Measure-theoretic niceties matter only in continuous or infinite settings.
- Countable additivity is strictly stronger than finite additivity; the two agree for finite sample spaces.
- Empirically estimated probabilities (sample proportions) satisfy the axioms by construction, but converge to the true probability only in the limit (law of large numbers).
- When combining events, always check disjointness before adding probabilities; use inclusion-exclusion when events overlap.
- The axioms say nothing about *interpretation* of probability (frequentist, Bayesian, logical); they are compatible with all three.

3.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.24.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.25 Law of Total Probability

3.25.1 Introduction

The law of total probability (LTP) expresses the unconditional probability of an event as a weighted sum of conditional probabilities. It is the bookkeeping

rule for "conditioning on every possibility and adding them up", and it appears as the denominator in Bayes' theorem, as the marginalisation step in mixture models, and whenever a probability computation is easier after splitting on a latent variable.

3.25.2 Prerequisites

Conditional probability and set partitions.

3.25.3 Theory

Let $\{B_1, B_2, \dots, B_n\}$ be a **partition** of the sample space: the events are pairwise disjoint and their union is Ω. Then for any event A,

$$P(A) = \sum_{i=1}^{n} P(A \mid B_i)\, P(B_i).$$

Proof sketch: $A = \bigcup_i (A \cap B_i)$ where the intersections are disjoint. By additivity, $P(A) = \sum_i P(A \cap B_i)$, and each $P(A \cap B_i) = P(A \mid B_i) P(B_i)$.

In continuous form, if X is a continuous random variable and A is an event,

$$P(A) = \int P(A \mid X = x) f_X(x)\, dx,$$

which is the marginalisation used to compute the probability of an event under a mixture or hierarchical model.

3.25.4 Assumptions

- The B_i form a partition: disjoint, exhaustive, positive probability each.
- Otherwise standard probability-space assumptions.

3.25.5 R Implementation

```r
# Population of patients divided by three risk strata with different
 ↪  event rates
strata_prop  <- c(low = 0.60, medium = 0.30, high = 0.10)
event_rates  <- c(low = 0.02, medium = 0.08, high = 0.25)

p_event <- sum(event_rates * strata_prop)
p_event

# Simulate to verify
set.seed(2026)
```

```r
n <- 1e5
stratum <- sample(names(strata_prop), n, replace = TRUE, prob =
 ↪  strata_prop)
event   <- rbinom(n, 1, event_rates[stratum])
mean(event)

# Bayes using LTP for the denominator
p_low_given_event <- event_rates["low"] * strata_prop["low"] / p_event
p_low_given_event
```

3.25.6 Output & Results

```
[1] 0.061         # P(event) = 0.02*0.6 + 0.08*0.3 + 0.25*0.1 = 0.061
[1] 0.0608        # empirical verification

      low
   0.1967         # P(low-risk | event)
```

The marginal event rate of 6.1% masks substantial heterogeneity across strata. Given that an event occurred, the probability that the patient is in the low-risk stratum is 19.7% – higher than some intuitions suggest, because low-risk patients outnumber high-risk by a factor of six.

3.25.7 Interpretation

Most population-level quantities in epidemiology and public health are weighted sums of stratum-specific rates. Reporting the weighted sum without the underlying decomposition hides heterogeneity; reporting the decomposition clarifies what drives the pooled rate.

3.25.8 Practical Tips

- Verify the partition: probabilities of the B_i must sum to 1; the events must not overlap.
- Conditioning on the "right" variable simplifies otherwise intractable probabilities – choose a latent with clean conditional structure.
- The LTP generalises to nested partitions and to countably infinite ones.
- In Bayesian computation, the LTP is the denominator of Bayes' theorem ($P(B)$ as the marginal likelihood) and is often the hard quantity to compute.
- For continuous marginalisations, numerical integration (`integrate()`) or Monte Carlo (`mean(f(rsample(...)))`) replace the sum.

3.25.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.25.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.26 The Log-Normal Distribution

3.26.1 Introduction

If $\log X$ is normally distributed, then X is log-normal. The distribution is strictly positive and right-skewed, and it models variables that result from multiplicative rather than additive processes – concentrations, income, survival times, and many biomedical lab values.

3.26.2 Prerequisites

Normal distribution, basic properties of logarithms.

3.26.3 Theory

A random variable $X > 0$ is log-normal with parameters μ and σ (on the log scale) if $\log X \sim \mathcal{N}(\mu, \sigma^2)$.

PDF:

$$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left\{-\frac{(\log x - \mu)^2}{2\sigma^2}\right\}, \qquad x > 0.$$

Moments:

- $E[X] = \exp(\mu + \sigma^2/2)$.
- $\mathrm{Var}(X) = [\exp(\sigma^2) - 1]\exp(2\mu + \sigma^2)$.
- Median $= e^{\mu}$; mode $= e^{\mu - \sigma^2}$.

The **geometric mean** of a log-normal sample equals e^{μ}, the median; it is the natural point estimate when the data combine multiplicatively.

Multiplicative CLT: the product of a large number of independent positive random variables tends to log-normal (by applying the CLT to the sum of logs).

3.26.4 Assumptions

Strictly positive continuous data whose logarithm is approximately normal.

3.26.5 R Implementation

```
mu    <- 1.0
sigma <- 0.5

X <- rlnorm(1e5, meanlog = mu, sdlog = sigma)
c(mean_X         = mean(X),
  theoretical    = exp(mu + sigma^2 / 2),
  median_X       = median(X),
  theoretical_med = exp(mu))

# Log-transform makes it normal
hist(log(X), breaks = 50, freq = FALSE, col = "#2A9D8F")
curve(dnorm(x, mu, sigma), add = TRUE, col = "#F4A261", lwd = 2)

# Geometric mean equals median (for lognormal)
gm <- exp(mean(log(X)))
c(geometric_mean = gm, median = median(X), theoretical = exp(mu))

# MLE
est_mu    <- mean(log(X))
est_sigma <- sd(log(X))
c(mu_hat = est_mu, sigma_hat = est_sigma)
```

3.26.6 Output & Results

```
       mean_X   theoretical      median_X  theoretical_med
        3.082         3.080         2.716            2.718

geometric_mean    median   theoretical
         2.714     2.716         2.718

   mu_hat    sigma_hat
    0.9995       0.500
```

Empirical arithmetic mean is larger than the median (right skew), the geometric mean equals the median (a property of log-normals), and the log-scale MLEs recover the true μ and σ.

3.26.7 Interpretation

For log-normal variables (concentrations, titres), report the geometric mean and a geometric-mean-based confidence interval rather than the arithmetic mean and

SD. In a manuscript: “geometric mean antibody titre 588 (95 % CI 441-782)”.

3.26.8 Practical Tips

- Plot on a log scale when visualising log-normal data; the skew is removed.
- For regression on log-normal outcomes, log-transform the response and fit a linear model on the log scale.
- Back-transformation of the mean of the log-transformed variable gives the geometric mean, not the arithmetic mean.
- For inference on the arithmetic mean, use $\hat{\mu} + \hat{\sigma}^2/2$ on the log scale and back-transform, but this requires a bias correction in small samples.
- Zero or negative values cannot be log-transformed; substitute or switch to a zero-inflated or Tobit model.

3.26.9 For Reviewers

Use the shared reviewer checklist, together with this method’s assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript’s application of the method.

3.26.10 See also — labs in this chapter

- Probability, conditional probability, Bayes’ theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.27 Marginal Distributions

3.27.1 Introduction

Given a joint distribution of several random variables, the marginal distribution of any one of them is obtained by integrating (or summing) out the others. Marginalisation is the probability-theoretic version of “don’t care”: collapsing multi-dimensional information into a single dimension.

3.27.2 Prerequisites

Joint distributions, integration/summation over probability densities.

3.27.3 Theory

For a joint density $f_{X,Y}(x, y)$:

$$f_X(x) = \int_{-\infty}^{\infty} f_{X,Y}(x,y)\,dy.$$

Analogously, for a joint PMF: $p_X(x) = \sum_y p_{X,Y}(x,y)$.

The joint CDF gives marginals by pushing the other argument to infinity: $F_X(x) = F_{X,Y}(x,\infty)$.

Key property: the marginals do not determine the joint. Many joint distributions share the same marginals but differ in dependence structure (verified by Sklar's theorem: marginals + copula = joint).

For a multivariate normal, the marginal of each component is univariate normal with the corresponding mean and variance; more generally, any subset of a multivariate normal is multivariate normal.

3.27.4 Assumptions

Integrability of the joint density; otherwise standard probability-space assumptions.

3.27.5 R Implementation

```r
library(mvtnorm)
set.seed(2026)

# Bivariate normal with correlation 0.6
Sigma <- matrix(c(1, 0.6, 0.6, 1), 2, 2)
XY <- rmvnorm(5000, sigma = Sigma)

# Marginal distributions of X and Y
c(mean_X = mean(XY[, 1]), sd_X = sd(XY[, 1]),
  mean_Y = mean(XY[, 2]), sd_Y = sd(XY[, 2]))

# Each marginal is standard normal regardless of correlation
ks.test(XY[, 1], "pnorm")
ks.test(XY[, 2], "pnorm")

# Demonstrate: same marginals, different joints
# Joint 1: correlation 0.6 (bivariate normal)
# Joint 2: same marginals but comonotonic (perfect rank correlation)
U <- pnorm(XY[, 1])   # uniform marginal via probability integral
 ↪  transform
X1 <- qnorm(U); Y1 <- qnorm(U)   # perfectly dependent copy
cov(XY)
cov(cbind(X1, Y1))
```

3.27.6 Output & Results

```
   mean_X      sd_X    mean_Y      sd_Y
   0.0099    1.0050   -0.0027    1.0018

        One-sample Kolmogorov-Smirnov test
D = 0.012, p-value = 0.4672          # X marginal matches N(0,1)

        One-sample Kolmogorov-Smirnov test
D = 0.015, p-value = 0.2132          # Y marginal matches N(0,1)

          [,1]      [,2]
[1,]     1.00     0.61
[2,]     0.61     1.00

          X1       Y1
X1       1.00     1.00       # comonotonic: correlation 1 with same
 ↪  marginals
Y1       1.00     1.00
```

Same marginals (both standard normal), very different joint distributions: correlation 0.6 versus perfect dependence.

3.27.7 Interpretation

In reporting, marginal summaries (marginal mean, SD) are usually what Table 1 presents. When dependence matters – pair-of-eyes measurements, twin pairs, repeated measures – a marginal analysis is incomplete and must be complemented with joint or conditional reporting.

3.27.8 Practical Tips

- Always report marginals alongside any joint analysis; they provide basic sanity checks.
- Marginalisation is trivial for multivariate normals: take the relevant subvector of the mean and the submatrix of the covariance.
- For continuous joint densities without closed form, marginalise numerically via `integrate()` or Monte Carlo.
- Empirical marginals are obtained by projection: for a sample $\{(X_i, Y_i)\}$, the marginal sample of X is $\{X_i\}$.
- In latent-variable models, marginalising over the latent gives the observed-data likelihood – the central quantity in EM and MCMC.

3.27.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.27.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.28 The Multinomial Distribution

3.28.1 Introduction

The multinomial distribution generalises the binomial from two categories to k: the counts of outcomes falling into each of several categories across n independent trials. It underlies the analysis of categorical variables with more than two levels – disease stages, treatment responses, species counts – and sits at the heart of chi-squared goodness-of-fit tests.

3.28.2 Prerequisites

Binomial distribution.

3.28.3 Theory

Let $(X_1, \dots, X_k)$ count outcomes across k categories in n independent trials, where the probability of outcome j on any trial is p_j, and $\sum p_j = 1$.

PMF:

$$P(X_1 = x_1, \dots, X_k = x_k) = \frac{n!}{x_1! \cdots x_k!} \prod_{j=1}^{k} p_j^{x_j},$$

for $x_j \geq 0$ and $\sum x_j = n$.

Moments:

- $E[X_j] = np_j$.
- $\mathrm{Var}(X_j) = np_j(1 - p_j)$.
- $\mathrm{Cov}(X_j, X_\ell) = -np_jp_\ell$ (negative because one count gain forces another's loss).

Marginals: $X_j \sim \mathrm{Binomial}(n, p_j)$. Pairwise joint for (X_j, X_ℓ): a trinomial with $(p_j, p_\ell, 1 - p_j - p_\ell)$.

MLE of $(p_1, \dots, p_k)$ is the vector of observed proportions $\hat{p}_j = x_j/n$.

3.28.4 Assumptions

- Fixed total n trials.
- Independent trials.
- Constant category probabilities across trials.

3.28.5 R Implementation

```r
set.seed(2026)

p <- c(0.40, 0.30, 0.20, 0.10)
n <- 200

# Single realisation
one_draw <- rmultinom(1, size = n, prob = p)
data.frame(category = 1:length(p), count = as.integer(one_draw),
           expected = n * p)

# Simulate many replicates, check moments
many <- rmultinom(1e4, n, p)
rowMeans(many); apply(many, 1, var)

# Expected variance
n * p * (1 - p)

# Chi-squared goodness-of-fit
obs <- c(82, 66, 34, 18)
chisq.test(obs, p = p)

# Multinomial likelihood of observed counts
dmultinom(obs, size = sum(obs), prob = p)
```

3.28.6 Output & Results

```
  category count expected
1        1    77       80
2        2    66       60
3        3    38       40
4        4    19       20

rowMeans:  80.07  60.04  39.95  19.94
apply var: 48.2   42.0   32.1   17.8

Expected var:
[1] 48.00 42.00 32.00 18.00
```

```
        Chi-squared test for given probabilities

data:  obs
X-squared = 3.2, df = 3, p-value = 0.363
```

The empirical means and variances match the theoretical values. The chi-squared test does not reject the specified category probabilities.

3.28.7 Interpretation

Reporting multinomial fits usually focuses on category proportions with uncertainty. For large samples, the chi-squared goodness-of-fit test is the standard tool for comparing observed multinomial counts to hypothesised probabilities.

3.28.8 Practical Tips

- For small expected counts (<5), use Fisher's exact test or Monte Carlo chi-squared (`chisq.test(...,simulate.p.value=TRUE)`).
- Multinomial categories are mutually exclusive by definition; overlapping classifications require a different framework (multilabel models).
- Conditional on the total, the multinomial is a sufficient statistic for the unordered category counts.
- Bayesian inference for multinomial probabilities uses the Dirichlet prior (conjugate); posterior is Dirichlet with shape parameters updated by observed counts.
- For regression with multinomial outcomes, use multinomial logistic regression (`nnet::multinom`).

3.28.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.28.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.29 The Multivariate Normal Distribution

3.29.1 Introduction

The multivariate normal (MVN) is the joint distribution that plays the same role for random vectors that the normal plays for random scalars. It is parameterised by a mean vector and a covariance matrix, and it underlies linear discriminant analysis, principal component analysis, Gaussian graphical models, and most of multivariate inference.

3.29.2 Prerequisites

Normal distribution, linear algebra (vectors, matrices, positive-definite matrices).

3.29.3 Theory

A random vector $\mathbf{X} \in \mathbb{R}^p$ is MVN with mean $\mu \in \mathbb{R}^p$ and positive definite covariance $\Sigma \in \mathbb{R}^{p \times p}$ if its density is

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\{-\tfrac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\}.$$

The exponent is the squared **Mahalanobis distance** from $\mathbf{x}$ to μ.

Key properties:

- **Marginals are MVN**: every subset of components is MVN.
- **Conditionals are MVN**: given a subset of components, the rest is MVN with mean and covariance depending on the conditioning values.
- **Linear transformations**: if $\mathbf{X} \sim \mathcal{N}_p(\mu, \Sigma)$ and A is any matrix, then $A\mathbf{X} \sim \mathcal{N}(A\mu, A\Sigma A^\top)$.
- **Zero correlation implies independence** in the MVN family (unlike general distributions).
- **Mahalanobis distance squared** $\sim \chi^2_p$ for $\mathbf{X} \sim \mathcal{N}_p(\mu, \Sigma)$.

3.29.4 Assumptions

Σ must be symmetric positive-definite. Degenerate cases (singular Σ) give a distribution supported on a lower-dimensional affine subspace.

3.29.5 R Implementation

```
library(mvtnorm); library(ggplot2)

set.seed(2026)
mu    <- c(0, 1)
```

```r
Sigma <- matrix(c(1, 0.6, 0.6, 2), 2, 2)

X <- rmvnorm(2000, mean = mu, sigma = Sigma)
colMeans(X)
cov(X)

df <- data.frame(X1 = X[, 1], X2 = X[, 2])
ggplot(df, aes(X1, X2)) +
  geom_point(alpha = 0.3, colour = "#2A9D8F") +
  stat_ellipse(level = 0.95, colour = "#F4A261", linewidth = 1) +
  coord_equal() +
  theme_minimal()

# Conditional distribution: X1 | X2 = x2
# mu_1 + Sigma_12 Sigma_22^{-1} (x2 - mu_2)
x2_val <- 2.0
cond_mean <- mu[1] + Sigma[1, 2] / Sigma[2, 2] * (x2_val - mu[2])
cond_var  <- Sigma[1, 1] - Sigma[1, 2]^2 / Sigma[2, 2]
c(mean = cond_mean, var = cond_var)

# Mahalanobis distance
md2 <- mahalanobis(X, center = mu, cov = Sigma)
c(mean_md2 = mean(md2), expected = 2)   # chi^2_2 mean = 2
```

3.29.6 Output & Results

```
 [1]  0.003 1.012

         [,1]      [,2]
[1,]    1.002    0.605
[2,]    0.605    1.989

    mean      var
  0.3000   0.8200

mean_md2 expected
    2.00     2.00
```

Empirical mean and covariance closely match the theoretical values, and the Mahalanobis distance squared has the expected chi-squared mean.

3.29.7 Interpretation

MVN assumptions underlie multiple regression (normal errors), linear discriminant analysis (normal class densities), and most factor-analytic and SEM approaches. Violations of joint normality invalidate ellipsoidal confidence regions

and some likelihood-based tests.

3.29.8 Practical Tips

- Check joint normality with chi-squared Q-Q plots of Mahalanobis distance squared, not just marginal Q-Q plots.
- Near-singular Σ (high collinearity) causes numerical instability; regularise with `Sigma+kI` or use generalised inverses.
- For high-dimensional MVN (p large), shrinkage estimators (Ledoit-Wolf) stabilise the covariance.
- Sampling from MVN: `mvtnorm::rmvnorm`, or directly via Cholesky: $\mathbf{X} = \mu + L\mathbf{z}$ with $LL^\top = \Sigma$ and $\mathbf{z} \sim \mathcal{N}_p(0, I)$.
- The Kullback-Leibler divergence between two MVNs has a closed form; useful in information-geometric methods.

3.29.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.29.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.30 The Negative Binomial Distribution

3.30.1 Introduction

The negative binomial distribution is the workhorse model for overdispersed counts – counts whose variance exceeds their mean. It arises naturally as the number of failures before the r-th success in a sequence of Bernoulli trials, or as a gamma mixture of Poisson distributions. It dominates modern count-regression practice (RNA-seq, public-health incidence, insurance claims).

3.30.2 Prerequisites

Geometric and Poisson distributions; gamma distribution is useful for the mixture interpretation.

3.30.3 Theory

Two standard parameterisations:

Counts-of-failures form: Y is the number of failures before the r-th success in iid Bernoulli(p) trials:

$$P(Y = k) = \binom{k + r - 1}{k} p^r (1 - p)^k, \qquad k = 0, 1, 2, \ldots$$

Mean-dispersion form: parameterise by mean μ and dispersion (size) parameter θ with

$$P(Y = k) = \frac{\Gamma(k + \theta)}{k!\,\Gamma(\theta)} \left(\frac{\theta}{\theta + \mu}\right)^{\theta} \left(\frac{\mu}{\theta + \mu}\right)^{k}.$$

Moments in the mean-dispersion form:

- $E[Y] = \mu$.
- $\mathrm{Var}(Y) = \mu + \mu^2/\theta$.

Gamma-Poisson interpretation: if $\Lambda \sim \mathrm{Gamma}(\theta, \theta/\mu)$ and $Y \mid \Lambda \sim \mathrm{Poisson}(\Lambda)$, then marginally $Y \sim \mathrm{NegBin}(\mu, \theta)$. As $\theta \to \infty$, the NB collapses to Poisson(μ).

Geometric as a special case: $r = 1$ (or $\theta = 1$) recovers the geometric distribution.

3.30.4 Assumptions

No common-rate assumption (unlike Poisson); the heterogeneity in underlying rates is parameterised by θ.

3.30.5 R Implementation

```r
library(MASS)

# R's rnbinom: parameterise by size (= theta) and mu
mu <- 5; theta <- 2
Y <- rnbinom(1e5, size = theta, mu = mu)

c(mean     = mean(Y),
  variance = var(Y),
  expected_var = mu + mu^2 / theta)

# PMF comparison to Poisson
k <- 0:20
```

```r
pmf_nb   <- dnbinom(k, size = theta, mu = mu)
pmf_pois <- dpois(k, lambda = mu)

data.frame(k, nb = round(pmf_nb, 4), poisson = round(pmf_pois, 4))

# Fit by MLE using MASS::glm.nb
df <- data.frame(y = Y, x = rnorm(1e5))
fit <- glm.nb(y ~ x, data = df)
summary(fit)$theta
```

3.30.6 Output & Results

```
     mean  variance expected_var
    5.020    17.48        17.50

    k      nb  poisson
1   0  0.2041   0.0067
2   1  0.2041   0.0337
3   2  0.1701   0.0842
4   3  0.1296   0.1404
5   4  0.0925   0.1755
6   5  0.0634   0.1755
...

[1] 1.998        # estimated theta
```

Empirical variance (17.5) matches the theoretical $\mu + \mu^2/\theta = 5 + 25/2 = 17.5$. The MLE of θ is close to the true value 2.

3.30.7 Interpretation

Negative binomial is the default count-data model in modern applied work. In a manuscript: "incidence rates were compared using negative binomial regression to account for overdispersion (dispersion estimated as $\theta = 2.0$)". Always report the dispersion parameter alongside the rate ratios.

3.30.8 Practical Tips

- Detect overdispersion by fitting a Poisson first and checking residual deviance $\approx$ df; ratios > 1.5 suggest overdispersion.
- `glm.nb` jointly estimates regression coefficients and dispersion; faster-but-less-flexible alternatives include `glmmTMB` and `brms`.
- Zero-inflated variants (`pscl::zeroinfl` with `dist="negbin"`) handle excess zeros on top of overdispersion.
- Reparameterisations: some textbooks use (n, p), some (μ, θ); R's `rnbinom(n,size,prob)` or `rnbinom(n,size,mu)` accepts either.

- In RNA-seq, DESeq2 and edgeR both model counts as negative binomial with shrinkage on the dispersion parameter.

3.30.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.30.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.31 The Normal Distribution

3.31.1 Introduction

The normal distribution – also called the Gaussian distribution – is the most important continuous distribution in applied statistics. It arises naturally as the limit of averages (via the central limit theorem), as the distribution of measurement errors, and as the default assumption in classical regression and ANOVA. Even when a raw variable is not itself normal, its transformations, its averages, or its residuals often are.

3.31.2 Prerequisites

Familiarity with probability density and cumulative distribution functions, and with the idea of expectation and variance, is assumed. The reader should know how to call R functions and generate a histogram.

3.31.3 Theory

The normal distribution with mean μ and variance σ^2 has the probability density function

$$f(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

defined for all real x. The mean, median, and mode all coincide at μ; the standard deviation σ controls the spread. The distribution is symmetric about μ, with thin tails – about 68% of the mass falls within one standard deviation

of the mean, 95% within two, and 99.7% within three. These "68-95-99.7" rule-of-thumb numbers are used constantly in applied work.

Several properties make the normal distribution uniquely convenient:

- **Closure under linear transformations**: if $X \sim \mathcal{N}(\mu, \sigma^2)$, then $aX + b \sim \mathcal{N}(a\mu + b, a^2\sigma^2)$.
- **Closure under sums**: the sum of independent normals is normal.
- **Maximum entropy**: among all distributions with a given mean and variance, the normal has the highest entropy – it is the least informative choice given those moments.
- **Conjugacy**: in Bayesian analysis with known variance, the normal prior is conjugate to the normal likelihood.

The standard normal is $\mathcal{N}(0, 1)$ and its CDF is denoted $\Phi(z)$. Any normal can be reduced to the standard normal by the linear transformation $Z = (X - \mu)/\sigma$.

3.31.4 Assumptions

When a procedure *assumes normality* it usually means one of two things: (i) the individual observations are normally distributed, or (ii) the residuals from the fitted model are normally distributed. The latter is much more common in practice, and is what matters for linear regression and ANOVA. For inference on the mean, the CLT often makes this distinction academic: the sampling distribution of the mean can be approximately normal even when the observations are not.

3.31.5 R Implementation

R provides the standard `d`, `p`, `q`, `r` interface for every distribution:

```r
library(ggplot2)

dnorm(0)
pnorm(1.96)
qnorm(0.975)
rnorm(5, mean = 10, sd = 2)

grid <- data.frame(x = seq(-4, 4, length.out = 401))
grid$density <- dnorm(grid$x)

ggplot(grid, aes(x = x, y = density)) +
  geom_line(colour = "steelblue", linewidth = 1) +
  geom_area(data = subset(grid, x >= -1.96 & x <= 1.96),
            aes(y = density), fill = "steelblue", alpha = 0.25) +
  labs(x = "z", y = "Density",
       title = "Standard normal with the central 95% shaded") +
  theme_minimal(base_size = 12)
```

`dnorm(0)` returns the density at the mean, $1/\sqrt{2\pi} \approx 0.3989$. `pnorm(1.96)` returns the CDF at 1.96, which is approximately 0.975. `qnorm(0.975)` returns the 97.5% quantile of the standard normal, which is approximately 1.96.

3.31.6 Output & Results

The plot shows the familiar bell curve with the central 95% shaded between $z = \pm 1.96$. The `rnorm()` call produces a sample of five values, for example `c(9.02,11.14,7.88,12.47,10.06)`, from a normal with mean 10 and SD 2.

3.31.7 Interpretation

In a manuscript, reporting a variable as "approximately normal, mean 75.2 (SD 8.4)" communicates that the distribution is roughly symmetric and bell-shaped, and that 95% of values lie in the range $75.2 \pm 1.96 \times 8.4 = (58.7, 91.7)$. If the distribution is skewed or heavy-tailed, reporting a mean and SD is misleading, and the summary should be a median with IQR.

3.31.8 Practical Tips

- Check normality with a Q-Q plot, not with a formal test. In small samples, formal tests lack power; in large samples, they reject for trivial deviations that do not affect inference.
- Data that are strictly positive and right-skewed (concentrations, incubation times, expression levels) are often approximately *log-normal*: their logarithm is normal. Analyses on the log scale may be more appropriate.
- In high dimensions (many variables per observation), joint normality is a much stronger assumption than marginal normality. Check both.
- "Normal" does not mean "common". Many natural phenomena are not normally distributed; always verify rather than assume.

3.31.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.31.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.32 The Poisson Distribution

3.32.1 Introduction

The Poisson distribution models counts of rare, independent events in a fixed window of time or space. Calls per hour, mutations per kilobase, photons per pixel, hospital admissions per day – all naturally Poisson when the underlying rate is approximately constant and events do not cluster.

3.32.2 Prerequisites

Discrete distributions and the notion of a rate.

3.32.3 Theory

A random variable X is **Poisson**(λ) if

$$P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!}, \qquad k = 0, 1, 2, \ldots$$

Moments:

- $E[X] = \lambda$.
- $\mathrm{Var}(X) = \lambda$ (equi-dispersion).
- Higher central moments: skewness $1/\sqrt{\lambda}$, excess kurtosis $1/\lambda$; the distribution becomes approximately normal for large λ.

Poisson limit theorem: $\mathrm{Binomial}(n, \lambda/n) \xrightarrow{d} \mathrm{Poisson}(\lambda)$ as $n \to \infty$. The Poisson is the "law of small numbers" limit of rare-event binomial trials.

Closure under sums: independent $X \sim \mathrm{Poisson}(\lambda_1)$ and $Y \sim \mathrm{Poisson}(\lambda_2)$ yield $X + Y \sim \mathrm{Poisson}(\lambda_1 + \lambda_2)$.

Connection to exponential: inter-arrival times in a Poisson process are iid exponential with rate λ.

3.32.4 Assumptions

The underlying events must be (approximately) independent with a constant rate. Violations of equi-dispersion (variance exceeding mean) signal **overdispersion**, commonly addressed by the negative binomial.

3.32.5 R Implementation

```
library(ggplot2)

lambda <- 3.5
```

```r
k <- 0:15
pmf <- dpois(k, lambda)

data.frame(k, pmf = round(pmf, 4))

# Moments
X <- rpois(1e5, lambda)
c(mean = mean(X), var = var(X), skew = mean((X - mean(X))^3) / sd(X)^3)

# Plot PMF
ggplot(data.frame(k, pmf), aes(k, pmf)) +
  geom_col(fill = "#2A9D8F") +
  scale_x_continuous(breaks = k) +
  labs(x = "k", y = "P(X = k)",
       title = paste("Poisson(", lambda, ")")) +
  theme_minimal()

# Verify Poisson limit from binomial
n_vals <- c(10, 100, 1000, 10000)
results <- sapply(n_vals, function(n) {
  p <- lambda / n
  X <- rbinom(1e5, n, p)
  c(n = n, mean = mean(X), var = var(X))
})
t(results)
```

3.32.6 Output & Results

```
    k    pmf
1   0 0.0302
2   1 0.1057
3   2 0.1850
4   3 0.2158
5   4 0.1888
6   5 0.1322
...

    mean     var    skew
    3.503   3.502   0.536

         n    mean     var
[1,]    10    3.49    3.13
[2,]   100    3.50    3.40
[3,]  1000    3.50    3.49
[4,] 10000    3.50    3.50
```

Mean and variance are both ≈ 3.5 in the empirical sample; the Poisson limit from binomial tightens as $n \to \infty$.

3.32.7 Interpretation

Poisson regression (`glm(...,family=poisson)`) estimates log-rates from count data; reports of incidence-rate ratios and event-per-person-time summaries are typically Poisson-based. Always check for overdispersion before accepting a Poisson fit at face value.

3.32.8 Practical Tips

- Equi-dispersion is the signature assumption; real count data often violate it.
- Use quasi-Poisson (`family=quasipoisson`) or negative binomial (`MASS::glm.nb`) when variance exceeds mean.
- Zero-inflated Poisson (`pscl::zeroinfl`) handles excess zeros.
- For rates per unit exposure (e.g., per person-year), include an offset in the regression: `glm(y~x,offset=log(exposure),family=poisson)`.
- The normal approximation works for $\lambda \geq 10$; for smaller λ, use the exact PMF.

3.32.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.32.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.33 The Probability Density Function

3.33.1 Introduction

For a continuous random variable, probability is distributed smoothly across the support rather than concentrated on countable points. The **probability density function (PDF)** encodes this distribution: integrating the PDF over a set gives the probability that the variable lands in that set.

3.33.2 Prerequisites

Calculus (integration) and the concept of a continuous random variable.

3.33.3 Theory

A continuous random variable X has a PDF f_X if, for every measurable set A,

$$P(X \in A) = \int_A f_X(x)\,dx.$$

A valid PDF satisfies:

1. $f_X(x) \geq 0$ for all x.
2. $\int_{-\infty}^{\infty} f_X(x)\,dx = 1$.

Crucially, $f_X(x)$ is **not** a probability. For continuous X, $P(X = x) = 0$ for every x. The density can exceed 1 at a given point, provided the integral is bounded. Density has units of inverse x: for x in seconds, $f_X(x)$ has units of 1/second.

Useful relationships:

- CDF: $F_X(x) = \int_{-\infty}^{x} f_X(u)\,du$.
- PDF from CDF: $f_X(x) = F_X'(x)$ where F_X is differentiable.
- Expected value: $E[X] = \int x f_X(x)\,dx$.
- Variance: $\mathrm{Var}(X) = \int (x - E[X])^2 f_X(x)\,dx$.

In R, each standard continuous distribution has a `d*()` function returning its PDF.

3.33.4 Assumptions

The variable is continuous (absolutely continuous with respect to Lebesgue measure). Mixed or discrete variables do not have ordinary PDFs (though the mixed case has a generalised density with respect to a mixed measure).

3.33.5 R Implementation

```
library(ggplot2)

x_grid <- seq(-4, 4, length.out = 801)
fx <- dnorm(x_grid, mean = 0, sd = 1)

ggplot(data.frame(x = x_grid, f = fx), aes(x, f)) +
  geom_line(colour = "#2A9D8F", linewidth = 1) +
  geom_area(data = subset(data.frame(x = x_grid, f = fx),
                          x >= -1.96 & x <= 1.96),
```

```
            aes(y = f), fill = "#2A9D8F", alpha = 0.25) +
  labs(x = "x", y = "f(x)",
       title = "Standard normal PDF with central 95% shaded") +
  theme_minimal()

# Verify normalisation by numerical integration
integrate(dnorm, -Inf, Inf)

# Probability of an interval via integration
integrate(dnorm, -1.96, 1.96)        # about 0.95

# Equivalently via the CDF
pnorm(1.96) - pnorm(-1.96)

# Moments via integration
integrate(function(x) x * dnorm(x), -Inf, Inf)        # E[X]
integrate(function(x) x^2 * dnorm(x), -Inf, Inf)      # E[X^2]
```

3.33.6 Output & Results

```
1 with absolute error < 9.4e-05

0.9500042 with absolute error < 1e-11

[1] 0.9500042

-4.4e-11 with absolute error < ...   # E[X] ≈ 0
1 with absolute error < ...         # E[X^2] = Var(X) for standard normal
```

Integrating the standard normal PDF over $[-1.96, 1.96]$ gives 0.95 – the well-known central probability – and the `pnorm` difference recovers the same value via the CDF.

3.33.7 Interpretation

For applied purposes, a PDF is visualised by its curve; probabilities are computed via `p*()` functions in R rather than by explicit integration. When reporting a continuous distribution in a paper, the density plot, mean, SD, and median are the standard summaries.

3.33.8 Practical Tips

- Density at a point is not probability; do not compare $f(x_1)$ to $f(x_2)$ as probabilities of outcomes 1 and 2. Compare $P(X \in (x_1 - h, x_1 + h))$ for small h instead.

- The density can be unbounded (e.g., Gamma with shape < 1 near 0); integrability, not boundedness, is what matters.
- Kernel density estimates (`density()`) approximate the PDF from a sample.
- Log-densities (`dnorm(...,log=TRUE)`) avoid underflow when evaluating likelihoods with many observations.
- The relationship $f = F'$ is the starting point for the inverse-CDF method: to generate X, apply F^{-1} to a uniform random variable.

3.33.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.33.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.34 The Probability Mass Function

3.34.1 Introduction

For a discrete random variable, the probability mass function (PMF) gives the probability that the variable takes each specific value. It is the discrete analogue of the density function, and the probability of any event reduces to summing the PMF over the relevant values.

3.34.2 Prerequisites

Discrete random variables and basic probability.

3.34.3 Theory

For a discrete random variable X with countable support $\{x_1, x_2, ...\}$, the PMF is

$$p_X(x) = P(X = x), \qquad x \in \mathbb{R}.$$

A valid PMF satisfies:

1. $p_X(x) \geq 0$ for all x.

2. $\sum_x p_X(x) = 1$ (total probability is 1).
3. $p_X(x) = 0$ for x outside the support.

The probability of an event is a sum over the PMF:

$$P(X \in A) = \sum_{x \in A} p_X(x).$$

The CDF is the running cumulative sum:

$$F_X(x) = P(X \leq x) = \sum_{x_i \leq x} p_X(x_i).$$

Expected value and variance follow the usual formulas:

$$E[X] = \sum_x x\, p_X(x), \qquad \mathrm{Var}(X) = \sum_x (x - E[X])^2\, p_X(x).$$

Common discrete PMFs in R use the `d<name>()` family: `dbinom`, `dpois`, `dgeom`, `dhyper`, `dnbinom`, `dmultinom`.

3.34.4 Assumptions

The variable must be discrete. Assigning a PMF to a continuous variable produces a degenerate object (zero at every point, failing normalisation).

3.34.5 R Implementation

```r
library(ggplot2)

# Binomial PMF for n = 10 trials, p = 0.3
n <- 10; p <- 0.3
k <- 0:n
pmf <- dbinom(k, size = n, prob = p)

data.frame(k, pmf)

# Verify normalisation
sum(pmf)

# Compute probabilities of events via sums
sum(pmf[k >= 3 & k <= 5])        # P(3 <= X <= 5)
sum(pmf[k %in% c(0, n)])          # P(X = 0 or X = n)

# Expected value and variance
```

```r
sum(k * pmf)
sum((k - sum(k * pmf))^2 * pmf)

ggplot(data.frame(k, pmf), aes(k, pmf)) +
  geom_col(fill = "#2A9D8F") +
  scale_x_continuous(breaks = k) +
  labs(x = "k", y = "P(X = k)",
       title = "Binomial(10, 0.3) PMF") +
  theme_minimal()
```

3.34.6 Output & Results

```
   k         pmf
1  0 0.028247525
2  1 0.121060821
3  2 0.233474440
4  3 0.266827932
5  4 0.200120949
6  5 0.102919345
7  6 0.036756909
8  7 0.009001692
9  8 0.001446773
10 9 0.000137781
11 10 0.0000059049

[1] 1            # normalisation
[1] 0.5695       # P(3 <= X <= 5)
[1] 0.0283       # P(X = 0 or X = 10)
[1] 3            # E[X] = np
[1] 2.1          # Var(X) = np(1-p)
```

3.34.7 Interpretation

A PMF table or bar chart is the natural display of a discrete distribution. For a manuscript, reporting "the number of responders out of 10 was binomially distributed with $p = 0.3$" conveys the full PMF implicitly.

3.34.8 Practical Tips

- Make sure the PMF sums to exactly 1 when you define a custom distribution; normalise explicitly.
- For mixtures of Poissons, Bernoullis, and similar, sums of PMFs computed componentwise give the mixture PMF.
- Use `choose(n,k)*p^k*(1-p)^(n-k)` for symbolic verification of the binomial PMF.

- Numerical underflow can corrupt PMFs at extreme values; work on the log scale (`dbinom(k,...,log=TRUE)`) when n is large.
- Histograms of samples approximate the PMF in finite samples; for a true PMF-bar-chart, use `geom_col()` at the exact support values.

3.34.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.34.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.35 Random Variables

3.35.1 Introduction

A **random variable** is a function that assigns a real number to each outcome of a random experiment. The concept packages probabilistic thinking into the language of numbers – means, variances, correlations – and is the abstraction on which every statistical method rests.

3.35.2 Prerequisites

Sample spaces, events, and the probability measure.

3.35.3 Theory

Given a probability space $(\Omega, \mathcal{F}, P)$, a **random variable** is a measurable function

$$X : \Omega \to \mathbb{R}$$

such that $\{X \leq x\} = \{\omega : X(\omega) \leq x\} \in \mathcal{F}$ for every $x \in \mathbb{R}$. Measurability ensures that we can assign probabilities to events defined in terms of X.

The **distribution** of X is the probability measure it induces on $\mathbb{R}$:

$$P_X(A) = P(X \in A) = P(\{\omega : X(\omega) \in A\}).$$

In practice, we rarely work with Ω directly; we work with the distribution P_X, summarised by its CDF $F_X(x) = P(X \leq x)$ and (when it exists) its PMF p_X or density f_X.

Random variables are classified as:

- **Discrete**: takes values in a countable set. Described by a PMF.
- **Continuous** (absolutely continuous): has a density. Described by a PDF.
- **Mixed**: combines discrete atoms and a continuous part. Common for censored / thresholded data.

Transformations of random variables give new random variables: if X is a random variable and $g : \mathbb{R} \to \mathbb{R}$ is measurable, then $Y = g(X)$ is a random variable with distribution determined by X and g.

3.35.4 Assumptions

A random variable is well-defined provided the underlying probability space exists and the function is measurable. Measurability is automatic for every function that arises in practice (continuous or piecewise-continuous).

3.35.5 R Implementation

In R, we usually work with random variables via their distributions.

```r
set.seed(2026)

# Discrete random variable: binomial
X_discrete <- rbinom(10000, size = 10, prob = 0.3)
head(sort(unique(X_discrete)))
prop.table(table(X_discrete))[1:5]     # empirical PMF

# Continuous random variable: normal
X_continuous <- rnorm(10000, mean = 0, sd = 1)
quantile(X_continuous, c(0.025, 0.5, 0.975))

# Transformation: Y = g(X) = X^2 of a standard normal
Y <- X_continuous^2
summary(Y)
var(Y)     # theoretical variance is 2 (Y ~ chi-squared with 1 df,
 ↪  variance 2)

# Mixed RV: censored at 0 (negative values truncated)
X_mixed <- pmax(rnorm(10000), 0)
mean(X_mixed == 0)
mean(X_mixed)
```

3.35.6 Output & Results

```
X_discrete       0       1       2       3       4
empirical  0.0306  0.1195  0.2314  0.2643  0.2059

         2.5%          50%         97.5%
   -1.935       -0.0089      1.9689

      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
     0.000    0.128     0.455     1.008    1.319    17.59
[1] 2.078                                              # var(X^2)

[1] 0.499                                              # P(X = 0)
[1] 0.399                                             # mean of censored
```

Each quantity (PMF, quantile, variance of a transformation, mixed distribution's atom) is a statistic of the corresponding random variable.

3.35.7 Interpretation

Random variables let us move from "outcome space" thinking to "number space" thinking. A methods section typically declares the random variable implicitly: "Let X be the systolic blood pressure of a randomly selected patient" specifies the random variable without bothering with the underlying Ω.

3.35.8 Practical Tips

- For most applied work, think of random variables as "generators of data" with a specified distribution; the measurability formalism is background.
- A random variable is distinct from its realisation. "X" is the variable; "x_i" is the observed value for unit i.
- Vector-valued random variables (**random vectors**) generalise the scalar case; their distribution is the joint distribution of the components.
- Transformations preserve measurability; common operations (sums, products, functions) are always measurable in practice.
- In R, `d*`, `p*`, `q*`, `r*` families of functions parameterise distributions; think of them as computing quantities of random variables with the specified distribution.

3.35.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.35.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.36 Sample Space and Events

3.36.1 Introduction

Before any probability can be assigned, we need to know what *could* happen. The **sample space** is the set of all possible outcomes of a random experiment; **events** are subsets of the sample space to which we assign probabilities. Clean reasoning about probability begins with clear identification of both.

3.36.2 Prerequisites

Familiarity with basic set theory – unions, intersections, complements – is sufficient.

3.36.3 Theory

The **sample space** Ω is the set of all possible outcomes. Examples:

- Coin toss: $\Omega = \{H, T\}$.
- Two dice: $\Omega = \{(i, j) : i, j \in \{1, \dots, 6\}\}$, 36 outcomes.
- Waiting time: $\Omega = [0, \infty)$.
- DNA base at position i: $\Omega = \{A, C, G, T\}$.

An **event** is a subset $A \subseteq \Omega$. We say event A occurs if the realised outcome ω is in A. Key operations:

- **Union** $A \cup B$: the event that A or B (or both) occurs.
- **Intersection** $A \cap B$: the event that both A and B occur.
- **Complement** A^c: the event that A does not occur.
- **Difference** $A \setminus B = A \cap B^c$: A but not B.

Two key identities – **De Morgan's laws** – relate complements and set operations:

$$(A \cup B)^c = A^c \cap B^c, \qquad (A \cap B)^c = A^c \cup B^c.$$

Events A and B are **disjoint** (mutually exclusive) if $A \cap B = \emptyset$. They are **independent** (under probability P) if $P(A \cap B) = P(A)P(B)$. Disjointness is about sets; independence is about probabilities – the two concepts should not be confused.

3.36.4 Assumptions

No assumptions beyond the underlying probability space being well-defined.

3.36.5 R Implementation

We can enumerate finite sample spaces explicitly and manipulate events as logical vectors.

```
# Two dice: enumerate the 36-element sample space
omega <- expand.grid(die1 = 1:6, die2 = 1:6)
head(omega, 3)

# Define events as logical vectors indexing rows of omega
A <- with(omega, die1 == 1)            # first die is 1
B <- with(omega, die1 + die2 == 7)     # sum equals 7

# Set operations
A_union_B     <- A | B
A_inter_B     <- A & B
A_complement  <- !A
A_minus_B     <- A & !B

# Under a uniform probability on 36 outcomes:
p <- function(E) mean(E)   # proportion of outcomes in E
c(P_A         = p(A),
  P_B         = p(B),
  P_A_union_B = p(A_union_B),
  P_A_and_B   = p(A_inter_B))

# Verify inclusion-exclusion: P(A U B) = P(A) + P(B) - P(A n B)
c(lhs = p(A_union_B),
  rhs = p(A) + p(B) - p(A_inter_B))

# Verify De Morgan: (A U B)^c = A^c n B^c
all((!(A | B)) == (!A & !B))
```

3.36.6 Output & Results

```
  die1 die2
1    1    1
2    2    1
3    3    1
```

```
       P_A        P_B P_A_union_B  P_A_and_B
    0.1667     0.1667      0.3056     0.02778
```

```
    lhs      rhs
 0.3056   0.3056
```

```
[1] TRUE
```

All identities are satisfied. $P(A \cup B) = 11/36$ because 11 outcomes satisfy "first die is 1 OR sum is 7", and the formula recovers the same value.

3.36.7 Interpretation

When writing a methods section, the sample space is usually defined implicitly by the design ("patients were followed for 30 days and classified as event / no event"). Making this explicit – "the sample space is $\{\text{event}, \text{no event}\}$" – is rarely necessary except in pedagogical contexts.

3.36.8 Practical Tips

- For continuous sample spaces ($\Omega = \mathbb{R}$ or higher-dimensional), not every subset is an event; measurability restrictions apply. In finite or countable sample spaces, every subset is an event.
- Do not conflate *disjoint* with *independent*. Two disjoint events with positive probability are maximally dependent: if one occurs, the other cannot.
- De Morgan's laws generalise to countable unions and intersections; they are indispensable when reasoning about complements of complex events.
- Encoding finite sample spaces in R (via `expand.grid` or combinations) makes exploratory probability calculations explicit and checkable.
- Inclusion-exclusion extends to many events: $P(\bigcup A_i) = \sum P(A_i) - \sum P(A_i \cap A_j) + \ldots$.

3.36.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.36.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.37 The Survival Function

3.37.1 Introduction

For a time-to-event random variable $T \geq 0$, the **survival function** $S(t) = P(T > t)$ is the probability of not having experienced the event by time t. It is the complement of the CDF and the single most important descriptive quantity in survival analysis.

3.37.2 Prerequisites

Random variables, CDF, basic survival-analysis concepts.

3.37.3 Theory

For a non-negative random variable T,

$$S(t) = P(T > t) = 1 - F(t).$$

Properties:

- $S(0) = 1$, $\lim_{t\to\infty} S(t) = 0$.
- Non-increasing.
- Right-continuous (by convention).

Related quantities:

- PDF: $f(t) = -S'(t)$.
- Hazard: $h(t) = f(t)/S(t)$.
- Cumulative hazard: $H(t) = -\log S(t)$.

Classical forms:

- Exponential: $S(t) = e^{-\lambda t}$.
- Weibull: $S(t) = \exp\big[-(t/\lambda)^k\big]$.
- Log-normal: $S(t) = 1 - \Phi\big((\log t - \mu)/\sigma\big)$.

Kaplan-Meier is the non-parametric MLE of S from right-censored data: a step function that drops at each observed event time by a factor reflecting the fraction of the at-risk set that experiences the event.

3.37.4 Assumptions

Non-negative random variable representing a time to an event. For Kaplan-Meier: independent censoring.

3.37.5 R Implementation

```
library(survival)
set.seed(2026)

# Theoretical survival functions
t_grid <- seq(0, 10, length.out = 400)
S_exp     <- exp(-0.3 * t_grid)
S_weibull <- exp(-(t_grid / 3)^2)
S_lnorm   <- 1 - plnorm(t_grid, meanlog = log(3), sdlog = 0.5)

plot(t_grid, S_exp, type = "l", col = "#6A4C93", lwd = 2,
     main = "Theoretical survival functions", xlab = "t", ylab = "S(t)")
lines(t_grid, S_weibull, col = "#2A9D8F", lwd = 2)
lines(t_grid, S_lnorm,   col = "#F4A261", lwd = 2)
legend("topright", c("exponential", "Weibull k=2", "log-normal"),
       col = c("#6A4C93", "#2A9D8F", "#F4A261"), lwd = 2)

# Empirical Kaplan-Meier from censored data
n <- 80
T_true <- rweibull(n, shape = 2, scale = 3)
C      <- runif(n, 0, 10)
T_obs  <- pmin(T_true, C)
delta  <- as.integer(T_true <= C)

km <- survfit(Surv(T_obs, delta) ~ 1)
plot(km, conf.int = TRUE, main = "Kaplan-Meier estimate",
     xlab = "t", ylab = "S(t)", col = "#2A9D8F")
lines(t_grid, exp(-(t_grid / 3)^2), col = "#F4A261", lwd = 2)

# Median survival
summary(km)$table["median"]
qweibull(0.5, shape = 2, scale = 3)
```

3.37.6 Output & Results

```
   median
    2.499

[1] 2.498
```

Empirical and theoretical median survival times agree. The Kaplan-Meier step function closely tracks the true Weibull survival curve.

3.37.7 Interpretation

Reporting a median survival with its 95 % CI from a Kaplan-Meier estimate is the standard headline statistic of survival studies. Survival functions are typically compared across groups via log-rank tests and visualised together on the same axes.

3.37.8 Practical Tips

- Report medians with CIs when survival drops below 0.5; report specific-time survival (1-year, 5-year) when the study ends before the median is reached.
- Kaplan-Meier is distribution-free; parametric survival models (Weibull, log-normal) give smooth curves but commit to a shape.
- Include a risk table below Kaplan-Meier plots (`survminer::ggsurvplot(..,risk.table=TRUE)`).
- Under heavy censoring, the tail of the survival curve is poorly estimated; avoid over-interpreting late-follow-up differences.
- Survival functions never take values above 1 or below 0; if your empirical estimator returns such values, there is a bug.

3.37.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.37.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.38 Student's t-Distribution

3.38.1 Introduction

The t-distribution (also called Student's t, after W. S. Gosset's pseudonym) arises as the ratio of a standard normal to the scaled square root of an independent chi-squared. It is the reference distribution for t-tests and for small-sample confidence intervals for a normal mean with unknown variance.

3.38.2 Prerequisites

Normal distribution, chi-squared distribution.

3.38.3 Theory

If $Z \sim \mathcal{N}(0,1)$ and $V \sim \chi^2_\nu$ are independent, then

$$T = \frac{Z}{\sqrt{V/\nu}} \sim t_\nu,$$

the t-distribution with ν **degrees of freedom**.

PDF:

$$f(t) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\nu\pi}\,\Gamma(\nu/2)} \left(1 + \frac{t^2}{\nu}\right)^{-(\nu+1)/2}.$$

Properties:

- Symmetric about 0.
- $E[T] = 0$ for $\nu > 1$; $\mathrm{Var}(T) = \nu/(\nu-2)$ for $\nu > 2$.
- Heavier tails than normal; excess kurtosis $6/(\nu-4)$ for $\nu > 4$.
- As $\nu \to \infty$, $t_\nu \to \mathcal{N}(0,1)$.

Pivotal quantity: for iid normal $X_1, \ldots, X_n$,

$$\frac{\bar{X} - \mu}{s/\sqrt{n}} \sim t_{n-1}.$$

This is the basis of the one-sample t-test and its confidence interval.

3.38.4 Assumptions

Normal underlying data (or large enough n for CLT to make the approximation reasonable).

3.38.5 R Implementation

```r
# PDF comparison
x <- seq(-5, 5, length.out = 400)
plot(x, dnorm(x), type = "l", col = "black", lwd = 2,
     main = "t with 3, 10, 30 df vs N(0,1)", ylab = "f(x)")
lines(x, dt(x, df = 3),  col = "#F4A261", lwd = 2)
lines(x, dt(x, df = 10), col = "#6A4C93", lwd = 2)
lines(x, dt(x, df = 30), col = "#2A9D8F", lwd = 2)
```

```r
legend("topright", c("normal", "t(3)", "t(10)", "t(30)"),
       col = c("black", "#F4A261", "#6A4C93", "#2A9D8F"), lwd = 2)

# Verify construction
n <- 1e5; nu <- 5
Z <- rnorm(n); V <- rchisq(n, df = nu)
T_sim <- Z / sqrt(V / nu)
c(emp_mean = mean(T_sim), theoretical = 0,
  emp_var  = var(T_sim),  theoretical_var = nu / (nu - 2))

# Small-sample CI for a mean
set.seed(2026)
X <- rnorm(15, mean = 10, sd = 3)
t_crit <- qt(0.975, df = length(X) - 1)
mean(X) + c(-1, 1) * t_crit * sd(X) / sqrt(length(X))
```

3.38.6 Output & Results

```
emp_mean theoretical  emp_var theoretical_var
  -0.003           0    1.665           1.667

[1]  8.76  12.06     # 95% CI using t_14
```

Empirical moments match theoretical at $\nu = 5$, and the small-sample CI properly uses t_{14} critical values.

3.38.7 Interpretation

Reporting a t-interval or t-test with "$t(n-1) = \ldots$" in a manuscript is reporting a t-distribution-based inference. The df equals $n-1$ for the one-sample t-test; Welch's two-sample t-test uses Satterthwaite df that may be fractional.

3.38.8 Practical Tips

- For $\nu \geq 30$ the t is nearly indistinguishable from the normal; for $\nu \leq 10$ the tail differences are substantial.
- The t-distribution is the one-parameter generalisation of the normal that allows heavier tails.
- Robust regression models sometimes use a t-likelihood to achieve outlier resistance.
- Non-central t (with non-centrality μ) governs power calculations for t-tests.
- For very heavy tails ($\nu \leq 2$), variance is infinite; mean exists only for $\nu > 1$.

3.38.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.38.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.39 Transformations of Random Variables

3.39.1 Introduction

Given a random variable X and a function g, the distribution of $Y = g(X)$ is determined by the distribution of X and the function. The change-of-variables formula computes the new distribution; for strictly monotone differentiable g this involves a Jacobian factor.

3.39.2 Prerequisites

Random variables, PDF, basic calculus.

3.39.3 Theory

Univariate, monotone case. Let X have density f_X on support $\mathcal{X}$, and let g be strictly monotone and differentiable on $\mathcal{X}$. Then $Y = g(X)$ has density

$$f_Y(y) = f_X(g^{-1}(y)) \left| \frac{dg^{-1}(y)}{dy} \right|.$$

The absolute value of the derivative of the inverse is the **Jacobian**.

Non-monotone case. If g is piecewise monotone, partition the domain into monotone segments and sum the contributions:

$$f_Y(y) = \sum_i f_X(x_i) / |g'(x_i)|$$

where $\{x_i\}$ are the solutions of $g(x) = y$.

Multivariate case. For $\mathbf{Y} = g(\mathbf{X})$ with $g : \mathbb{R}^n \to \mathbb{R}^n$ a diffeomorphism,

$$f_{\mathbf{Y}}(\mathbf{y}) = f_{\mathbf{X}}(g^{-1}(\mathbf{y})) \, |\det J_{g^{-1}}(\mathbf{y})|,$$

where $J_{g^{-1}}$ is the Jacobian matrix of g^{-1}.

Common examples:

- $Y = aX + b$: $f_Y(y) = f_X((y-b)/a)/|a|$.
- $Y = X^2$, X standard normal: $Y \sim \chi^2_1$.
- $Y = e^X$, X normal: Y is log-normal.
- $Y = F^{-1}(U)$, $U \sim \text{Uniform}(0,1)$: the inverse-CDF method for generating arbitrary distributions.

3.39.4 Assumptions

- g must be measurable.
- For the change-of-variables formula to apply, g must be strictly monotone and differentiable (or piecewise so).

3.39.5 R Implementation

```r
set.seed(2026)

# Case 1: Y = X^2 for X ~ N(0, 1); Y should be chi-squared with 1 df
X <- rnorm(1e5)
Y <- X^2

par(mfrow = c(1, 2))
hist(Y, breaks = 100, freq = FALSE, main = "Empirical", col = "#2A9D8F")
curve(dchisq(x, df = 1), from = 0, to = 10, add = TRUE, col = "#F4A261",
 ↪  lwd = 2)

# Case 2: Y = exp(X) for X ~ N(mu, sigma) is log-normal
mu <- 1; sigma <- 0.5
X2 <- rnorm(1e5, mu, sigma)
Y2 <- exp(X2)
curve(dlnorm(x, mu, sigma), from = 0, to = 10,
      main = "exp(N(1, 0.25)) vs lognormal PDF",
      col = "#2A9D8F", lwd = 2)
hist(Y2, breaks = 100, freq = FALSE, add = TRUE, border = "#F4A261")

par(mfrow = c(1, 1))

# Case 3: Inverse-CDF method to generate exponential from uniform
U <- runif(1e5)
E <- -log(1 - U)    # F^{-1}(u) = -log(1 - u) for rate = 1
c(mean = mean(E), theoretical = 1)
```

3.39.6 Output & Results

```
    mean  theoretical
   0.999        1.000
```

Empirical histograms match the theoretical transformed densities. The inverse-CDF method recovers the exponential mean.

3.39.7 Interpretation

Many "derived" distributions in statistics (chi-squared from normals, log-normal from normals, Cauchy from normals, Beta from gammas) arise as transformations of simpler distributions. Recognising the transformation short-cuts many derivations.

3.39.8 Practical Tips

- Always check strict monotonicity before applying the simple Jacobian formula; square-mapping X^2 is a classic non-monotone case.
- For inverse-CDF simulation, you need the CDF's inverse; use `qnorm`, `qgamma`, etc., in R.
- The Jacobian for a multivariate linear transformation $Y = AX$ is $|detA|$, independent of location.
- Delta method is the "linearised" version of the change-of-variables formula, used for asymptotic distributions.
- Non-bijective transformations (e.g., $|X|$) require summing over preimages, as in the non-monotone case.

3.39.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.39.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.40 The Uniform Distribution

3.40.1 Introduction

The uniform distribution spreads probability equally across its support. The continuous uniform on $[a, b]$ is the building block of random-number generation; the discrete uniform on $\{1, \dots, k\}$ describes fair dice and random selection. It is the simplest non-trivial distribution.

3.40.2 Prerequisites

Continuous and discrete random variables.

3.40.3 Theory

Continuous uniform Uniform(a, b) has PDF

$$f(x) = \frac{1}{b-a}, \qquad a \le x \le b,$$

with moments $E[X] = (a+b)/2$ and $\text{Var}(X) = (b-a)^2/12$.

Discrete uniform on $\{x_1, \dots, x_k\}$: $P(X = x_i) = 1/k$ for each i. For $\{1, \dots, k\}$: $E[X] = (k+1)/2$, $\text{Var}(X) = (k^2-1)/12$.

Probability integral transform: if X has a continuous CDF F_X, then $F_X(X) \sim \text{Uniform}(0, 1)$. The inverse: $X = F_X^{-1}(U)$ where $U \sim \text{Uniform}(0, 1)$ generates X from uniform draws. This is the inverse-CDF method of simulation.

3.40.4 Assumptions

Continuous uniform assumes the variable genuinely has constant density; discrete uniform assumes genuinely equal probabilities across a specified set.

3.40.5 R Implementation

```r
set.seed(2026)

# Continuous uniform
U <- runif(1e5, min = 0, max = 1)
c(mean = mean(U), theoretical = 0.5,
  var  = var(U),  theoretical_var = 1/12)

# Inverse-CDF method: generate exponential from uniform
lambda <- 2
E <- -log(1 - U) / lambda   # F^{-1}(u) = -log(1 - u) / lambda
c(empirical_mean = mean(E), exponential_mean = 1 / lambda)
```

```r
# Discrete uniform via sample()
die <- sample(1:6, 1e5, replace = TRUE)
c(mean_die = mean(die), theoretical = 3.5,
  var_die  = var(die),  theoretical_var = 35/12)
```

3.40.6 Output & Results

```
       mean theoretical          var  theoretical_var
     0.5003      0.5000       0.0835           0.0833

empirical_mean  exponential_mean
        0.4983             0.500

  mean_die  theoretical     var_die  theoretical_var
     3.504        3.500       2.924            2.917
```

Empirical moments match theoretical exactly up to Monte Carlo error, and the inverse-CDF method recovers the exponential mean.

3.40.7 Interpretation

Uniforms rarely appear as substantive models of real data but pervade simulation, randomisation, and Monte Carlo methods. Reporting "trial assignment was simulated via `sample()`" is implicitly invoking a discrete uniform.

3.40.8 Practical Tips

- `runif(n)` defaults to $[0, 1]$; specify min/max explicitly to avoid surprises.
- The uniform distribution has zero skewness and excess kurtosis -1.2 (light tails).
- For true randomness from hardware sources, avoid pseudo-random `runif`; use `/dev/urandom` or the `random` R package with online services.
- Uniform priors are often overly informative on transformed scales (a uniform on σ is not uniform on σ^2).
- The Jacobian of the inverse-CDF transform is $1/f_X(x)$; this is why the probability integral transform works.

3.40.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.40.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.41 Variance and Covariance

3.41.1 Introduction

Variance measures how far a single random variable ranges around its mean; covariance measures how two random variables move together around their means. Together they populate the covariance matrix of a random vector, which is the central object in multivariate statistics, regression, and portfolio theory.

3.41.2 Prerequisites

Expectation and basic properties of random variables.

3.41.3 Theory

Variance of X:

$$\operatorname{Var}(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2.$$

Covariance of X and Y:

$$\operatorname{Cov}(X, Y) = E[(X - E[X])(Y - E[Y])] = E[XY] - E[X]E[Y].$$

Variance is a special case: $\operatorname{Cov}(X, X) = \operatorname{Var}(X)$.

Properties:

- Non-negativity: $\operatorname{Var}(X) \geq 0$, with equality iff X is constant almost surely.
- Scale: $\operatorname{Var}(aX + b) = a^2 \operatorname{Var}(X)$.
- Symmetry: $\operatorname{Cov}(X, Y) = \operatorname{Cov}(Y, X)$.
- Bilinearity: $\operatorname{Cov}(aX + b, cY + d) = ac \operatorname{Cov}(X, Y)$.
- Sum: $\operatorname{Var}(X + Y) = \operatorname{Var}(X) + \operatorname{Var}(Y) + 2\operatorname{Cov}(X, Y)$.
- Independence implies zero covariance: $X \perp Y \Rightarrow \operatorname{Cov}(X, Y) = 0$. The converse is false.

The **covariance matrix** of a random vector $\mathbf{X} = (X_1, \ldots, X_p)$ is the $p \times p$ matrix Σ with entries $\Sigma_{ij} = \operatorname{Cov}(X_i, X_j)$. Σ is symmetric positive semi-definite.

3.41.4 Assumptions

$E[X^2] < \infty$ and $E[Y^2] < \infty$ for variance and covariance to be defined.

3.41.5 R Implementation

```r
library(mvtnorm)
set.seed(2026)

# Univariate variance
X <- rnorm(1e5, mean = 10, sd = 3)
c(theoretical = 9, empirical = var(X))

# Covariance from bivariate normal with known Sigma
Sigma <- matrix(c(4, 1.8,
                  1.8, 9), nrow = 2)
n <- 1e5
XY <- rmvnorm(n, mean = c(0, 0), sigma = Sigma)
cov(XY)
cov(XY) / (sqrt(Sigma[1, 1] * Sigma[2, 2]))   # correlation = 0.3

# Variance of a sum verifies the formula
X <- XY[, 1]; Y <- XY[, 2]
c(var_sum = var(X + Y),
  formula = var(X) + var(Y) + 2 * cov(X, Y))

# Zero covariance does not imply independence
X <- rnorm(1e5)
Y <- X^2
c(cov = cov(X, Y),
  correlation = cor(X, Y))
```

3.41.6 Output & Results

```
theoretical   empirical
      9.0         9.04

          [,1]      [,2]
[1,]   3.998     1.791
[2,]   1.791     8.984

          [,1]      [,2]
[1,]   0.999     0.299
[2,]   0.299     0.998

 var_sum formula
  16.56   16.56

       cov   correlation
     0.003        0.002
```

Empirical variance and covariance match the theoretical values within Monte Carlo error. X and X^2 have zero covariance but are clearly dependent – covariance detects only linear association.

3.41.7 Interpretation

Reporting a mean and a variance (or SD) gives a rough summary of a distribution's location and spread; reporting a covariance matrix gives the joint structure of several variables. Most multivariate methods (PCA, MANOVA, linear discriminant analysis) work with sample covariance matrices.

3.41.8 Practical Tips

- Sample variance uses divisor $n-1$ for unbiasedness; R's `var()` uses $n-1$ by default.
- $\mathrm{Cov}(X, Y) = 0$ does not imply independence except in the Gaussian case.
- The covariance matrix must be positive semi-definite; estimation from small samples can produce near-singular matrices.
- Linearly transforming a random vector by matrix A changes the covariance matrix to $A\Sigma A^\top$.
- For robust multivariate covariance estimation in contaminated data, `robustbase::covMcd` is the standard MCD estimator.

3.41.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.41.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.42 The Weibull Distribution

3.42.1 Introduction

The Weibull distribution generalises the exponential to allow monotone non-constant hazard rates. It is the default parametric model in reliability engineering and a common choice for parametric survival analysis. The shape parameter controls whether hazard is increasing (ageing), decreasing (early failures dominate), or constant (exponential).

3.42.2 Prerequisites

Exponential distribution, hazard and survival functions.

3.42.3 Theory

A Weibull random variable with shape $k > 0$ and scale $\lambda > 0$ has PDF

$$f(x) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} \exp\left[-\left(\frac{x}{\lambda}\right)^{k}\right], \qquad x \geq 0.$$

CDF: $F(x) = 1 - \exp\big[-(x/\lambda)^k\big]$.

Hazard function: $h(x) = (k/\lambda)(x/\lambda)^{k-1}$. Three regimes:

- $k < 1$: decreasing hazard (early failures / "infant mortality").
- $k = 1$: constant hazard (exponential).
- $k > 1$: increasing hazard (wear-out / ageing).

Moments:

- $E[X] = \lambda\,\Gamma(1 + 1/k)$.
- $\mathrm{Var}(X) = \lambda^2\left[\Gamma(1 + 2/k) - \Gamma(1 + 1/k)^2\right]$.

Accelerated failure time form: $\log T = \mu + \sigma W$ where W is standard extreme-value; the AFT parameters relate to Weibull's $k = 1/\sigma$ and $\lambda = e^{\mu}$.

3.42.4 Assumptions

Non-negative continuous data, monotone hazard (log-hazard linear in log-time).

3.42.5 R Implementation

```
library(survival)

# PDFs for three shape regimes
x <- seq(0, 5, length.out = 400)
plot(x, dweibull(x, shape = 0.5, scale = 1), type = "l", col = "#F4A261",
     lwd = 2, main = "Weibull PDFs", ylab = "f(x)")
lines(x, dweibull(x, shape = 1, scale = 1), col = "#6A4C93", lwd = 2)
lines(x, dweibull(x, shape = 2.5, scale = 1), col = "#2A9D8F", lwd = 2)
legend("topright", c("k = 0.5 (decreasing h)", "k = 1 (exponential)", "k
 ↪  = 2.5 (increasing h)"),
       col = c("#F4A261", "#6A4C93", "#2A9D8F"), lwd = 2)

# Simulate and fit
set.seed(2026)
X <- rweibull(500, shape = 2.0, scale = 1.5)
```

```r
fit <- survreg(Surv(X, rep(1, length(X))) ~ 1, dist = "weibull")
summary(fit)

# Convert survreg output to Weibull parameters
c(shape = 1 / fit$scale,
  scale = exp(coef(fit)))
```

3.42.6 Output & Results

```
Call: survreg(...)

               Value Std. Error    z      p
(Intercept)  0.40578    0.01574 25.78  <2e-16
Log(scale)  -0.69312    0.03293 -21.0  <2e-16

Scale = 0.500

   shape    scale
   2.000    1.501
```

The fit recovers $k \approx 2$ and $\lambda \approx 1.5$ – the true parameters. The increasing-hazard regime is visible in the PDF.

3.42.7 Interpretation

In survival analysis, a Weibull model is often the starting parametric fit: "we fitted a Weibull regression with shape $k = 1.3$ (95 % CI 1.1-1.5), indicating increasing hazard over follow-up." Weibull is the default choice when the proportional-hazards assumption holds on a transformed scale.

3.42.8 Practical Tips

- R's `rweibull(n,shape,scale)` conventions differ from `survreg`, which parameterises via intercept and scale $= 1/k$.
- For right-censored data, use `survreg(...,dist="weibull")`, not `fitdistr` on complete data.
- Check the fit via a log-log plot: $\log(-\log S(t))$ vs $\log t$ should be approximately linear.
- If shape estimates are very close to 1 with narrow CI, consider the simpler exponential.
- `flexsurv::flexsurvreg(...,dist="weibull")` is a modern alternative with richer diagnostics.

3.42.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

3.42.10 See also — labs in this chapter

- Probability, conditional probability, Bayes' theorem
- Discrete distributions: Bernoulli, binomial, Poisson
- Continuous distributions and Q-Q plots

3.43 Week 2, Session 5 — Continuous distributions and Q-Q plots

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

3.43.1 Learning objectives

- Draw densities of the normal, t, chi-square, F, and exponential distributions and describe how shape changes with parameters.
- Use Q-Q plots to assess whether data are consistent with a hypothesised distribution.
- Recognise the degrees-of-freedom relationships among the normal-derived distributions.

3.43.2 Prerequisites

Lab 2.4.

3.43.3 Background

The continuous distributions in this lab cover the majority of routine parametric inference. The normal is the distribution of means under the central limit theorem. The t is what happens to the normal when the variance is itself estimated from the data — with more uncertainty (smaller n), heavier tails. The chi-square is the distribution of sums of squared standard normals. The F is a ratio of chi-squares. The exponential appears in survival and queueing.

The Q-Q plot is the workhorse for assessing distributional fit. Plotting sample quantiles against theoretical quantiles gives a straight line when the model is right. Deviations at the tails indicate skew or heavy tails; an S-shape indicates

non-normal symmetric behaviour. Q-Q plots are how assumptions of the t-test and the analysis of variance get checked in practice.

Degrees of freedom are easy to mis-remember. The short version: t has df $= n - 1$ for a one-sample test; chi-square from a sum of k squared standard normals has df $= k$; F has two df — numerator and denominator — that come from the chi-squares in its ratio.

3.43.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

3.43.5 1. Hypothesis

We are not testing a hypothesis. We are characterising five continuous distributions and using Q-Q plots to assess whether a sample is consistent with a model.

3.43.6 2. Visualise

Plot densities side by side.

```
grid <- tibble(x = seq(-5, 5, length.out = 400))

densities <- bind_rows(
  grid |> mutate(density = dnorm(x),         dist = "Normal(0,1)"),
  grid |> mutate(density = dt(x, df = 3),    dist = "t(3)"),
  grid |> mutate(density = dt(x, df = 30),   dist = "t(30)")
)
```

```
ggplot(densities, aes(x, density, colour = dist)) +
  geom_line(linewidth = 1) +
  labs(x = NULL, y = "Density", colour = NULL)
```

The t with 3 df has visibly heavier tails; at 30 df it is nearly normal.

```
chi_grid <- tibble(x = seq(0.01, 20, length.out = 400))
chi <- bind_rows(
  chi_grid |> mutate(density = dchisq(x, df = 2), dist = "chi-sq(2)"),
  chi_grid |> mutate(density = dchisq(x, df = 5), dist = "chi-sq(5)"),
  chi_grid |> mutate(density = dchisq(x, df = 10), dist = "chi-sq(10)")
)
```

```
ggplot(chi, aes(x, density, colour = dist)) +
  geom_line(linewidth = 1) +
```

```
  labs(x = NULL, y = "Density", colour = NULL)
```

3.43.7 3. Assumptions

We assume samples are independent and identically distributed. For the Q-Q check, the sample must be reasonably sized (say, at least 20 observations) for the plot to be informative.

```
# Facet across df for the F distribution.
f_grid <- expand_grid(
  x  = seq(0.01, 5, length.out = 300),
  df1 = c(2, 5, 10),
  df2 = c(10, 30)
) |>
  mutate(density = df(x, df1, df2))
```

```
f_grid |>
  ggplot(aes(x, density, colour = factor(df1))) +
  geom_line(linewidth = 1) +
  facet_wrap(~ df2, labeller = label_both) +
  labs(x = NULL, y = "Density", colour = "df1")
```

3.43.8 4. Conduct

Q-Q plots. Simulate data, compare to a hypothesised distribution.

```
n <- 200
samples <- tibble(
  normal_sample = rnorm(n),
  heavy_tail    = rt(n, df = 3),
  exponential   = rexp(n, rate = 1)
)
```

```
samples |>
  pivot_longer(everything(), names_to = "sample", values_to = "value") |>
  ggplot(aes(sample = value)) +
  stat_qq(alpha = 0.6) +
  stat_qq_line(colour = "firebrick") +
  facet_wrap(~ sample, scales = "free") +
  labs(x = "Theoretical normal quantiles",
       y = "Sample quantiles")
```

The normal sample lies on the 45° reference line; the heavy-tailed sample splays at both ends; the exponential curves away on one side.

Shapiro-Wilk as a quick numerical check on the three samples.

```
sw_results <- samples |>
  pivot_longer(everything(), names_to = "sample", values_to = "value") |>
  group_by(sample) |>
  summarise(p = shapiro.test(value)$p.value, .groups = "drop")
sw_results
```

3.43.9 5. Concluding statement

> The normal sample ($n =$ `n`) was consistent with normality both visually (Q-Q line) and by Shapiro-Wilk ($p =$ `signif(sw_results$p[sw_results$sample=="normal_sample"],2)`). The $t(3)$ sample showed clear tail departures, Shapiro-Wilk $p =$ `signif(sw_results$p[sw_results$sample=="heavy_tail"],2)`. The exponential sample was strongly right-skewed, $p < 0.001$.

A Q-Q plot is the most information-dense way to inspect a distributional assumption. Numerical tests of normality (Shapiro-Wilk, Kolmogorov-Smirnov) should corroborate, not replace, the plot.

3.43.10 Common pitfalls

- Reading the centre of a Q-Q plot and ignoring the tails. The tails are where the interesting departures live.
- Declaring a distribution wrong on Shapiro-Wilk in a sample of 5000 when the Q-Q plot is straight.
- Forgetting that t approaches normal as df $\to \infty$; do not test with df $=$ 500 in a fit-diagnostic mindset.

3.43.11 Further reading

- Rice JA. *Mathematical Statistics and Data Analysis.*
- Kutner MH et al. *Applied Linear Statistical Models.*

3.43.12 Session info

```
sessionInfo()
```

3.43.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

3.44 Week 2, Session 4 — Discrete distributions: Bernoulli, binomial, Poisson

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

3.44.1 Learning objectives

- Simulate Bernoulli trials and aggregate them to a binomial.
- Show the Poisson approximation to the binomial when n is large and p is small.
- Use `dbinom()`, `pbinom()`, and `rbinom()` (and their Poisson counterparts) to answer probability questions.

3.44.2 Prerequisites

Lab 2.2 (probability).

3.44.3 Background

The Bernoulli distribution is the simplest non-trivial distribution: one coin, two outcomes, a single probability p. The binomial adds structure — n independent Bernoulli trials with the same p — and gives the distribution of the count of successes. The Poisson arises in the limit where n is large, p is small, and np stays fixed; it describes counts of rare events over a fixed window of opportunity, without needing to specify n and p separately.

Most count-like quantities in biomedicine can be modelled by one of these three. Whether a tumour is responding (Bernoulli), how many of twenty patients relapse (binomial), how many emergency admissions arrive in an hour (Poisson) — the same three distributions cover a great deal of ground. The simulation-first approach makes them concrete before we write a formula.

There is a common conflation between the binomial and the Poisson in the clinical literature. When the sample size is large and the event is rare, the two distributions give nearly identical probabilities and either is defensible. When the event is not rare, the Poisson can assign meaningful probability to impossible events (more successes than trials), so the binomial is strictly better.

3.44.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

3.44.5 1. Hypothesis

We characterise three discrete distributions by simulation, then compare the empirical frequencies to the theoretical probabilities. No hypothesis is tested.

3.44.6 2. Visualise

A Bernoulli trial: one coin, success probability 0.3.

```
n_trials <- 1000
bern <- rbinom(n_trials, 1, 0.3)
mean(bern)   # empirical p
```

A binomial count: how many successes in 20 trials each at $p = 0.3$, repeated 10,000 times.

```
n_rep <- 10000
bin  <- rbinom(n_rep, size = 20, prob = 0.3)

bin_tibble <- tibble(k = bin) |>
  count(k) |>
  mutate(prop = n / sum(n),
         theo = dbinom(k, 20, 0.3))
```

```
bin_tibble |>
  ggplot(aes(k, prop)) +
  geom_col(fill = "grey70", alpha = 0.8) +
  geom_point(aes(y = theo), colour = "firebrick", size = 2) +
  labs(x = "Successes in 20 trials",
       y = "Proportion (bars) vs. P(k) (points)")
```

3.44.7 3. Assumptions

Binomial assumes fixed n, independence, and constant p. Poisson assumes a constant rate and independent counts over disjoint intervals. When n is large and p is small with $np = \lambda$, the binomial approaches Poisson.

```
# n=1000, p=0.003, so np = 3. Compare binomial and Poisson probabilities.
probs <- tibble(
  k = 0:10,
  binom  = dbinom(k, 1000, 0.003),
  pois   = dpois(k, 3)
)
probs
```

3.44.8 4. Conduct

Answer two specific questions.

```r
# Q1: In a trial of 20 patients at p=0.3 response rate, what is
# P(at least 8 responders)?
p_ge8 <- 1 - pbinom(7, size = 20, prob = 0.3)
p_ge8

# Q2: An ER admits patients at a rate of 3/hour. What is
# P(7 or more patients in a given hour)?
p_pois_ge7 <- 1 - ppois(6, lambda = 3)
p_pois_ge7
```

Simulate Q2 to confirm.

```r
sim <- rpois(10000, lambda = 3)
mean(sim >= 7)
```

```r
tibble(k = 0:15) |>
  mutate(pois_density = dpois(k, 3)) |>
  ggplot(aes(k, pois_density)) +
  geom_col(fill = "steelblue") +
  labs(x = "Number of events", y = "P(X = k)")
```

3.44.9 5. Concluding statement

> A response rate of 0.3 in a 20-patient trial implies a probability of `round(p_ge8,3)` of observing eight or more responders. An ER with a mean admission rate of 3/hour has a probability of `round(p_pois_ge7,3)` of receiving seven or more admissions in a given hour; 10,000 simulated hours recovered an empirical proportion of `round(mean(sim>=7),3)`. The binomial with $n = 1000$, $p = 0.003$ and the Poisson with $\lambda = 3$ gave nearly indistinguishable probabilities across $k = 0$–10.

Discrete distributions are under-appreciated. Many designs that are reached for as continuous (eg regressions on a rate) have a cleaner count-based model sitting next to them.

3.44.10 Common pitfalls

- Using the normal approximation to the binomial when n is small. For small n use `pbinom()` directly.
- Modelling a rate as a proportion when the denominator is person-time, not persons.
- Treating a Poisson with a known λ as if its variance had to be estimated. For Poisson, mean = variance.

3.44.11 Further reading

- Rosner B. *Fundamentals of Biostatistics*, chapter on discrete distributions.
- Agresti A. *Categorical Data Analysis*, opening chapter.

3.44.12 Session info

```
sessionInfo()
```

3.44.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

3.45 Week 2, Session 2 — Probability, conditional probability, Bayes' theorem

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

3.45.1 Learning objectives

- Distinguish marginal, joint, and conditional probability and compute each from a contingency table.
- State Bayes' theorem and apply it to a medical-testing scenario.
- Simulate a population and recover conditional probabilities by counting.

3.45.2 Prerequisites

Basic probability.

3.45.3 Background

Probability is the language for reasoning under uncertainty. In biostatistics, most of the probabilities we care about are conditional — the probability that a patient has a disease *given* a test result; the probability of survival *given* a treatment; the probability of an adverse event *given* a risk factor. The distinction between marginal and conditional probability is the distinction between "how common is X in the population" and "how common is X in a particular subgroup".

Bayes' theorem is the rule for flipping a conditional probability. Given $P(\text{A}|\text{B})$ and the marginal probabilities $P(\text{A})$ and $P(\text{B})$, it returns $P(\text{B}|\text{A})$. In medical testing this lets us move from *how often does the test fire when the disease*

is present (sensitivity) to *how likely is disease given a positive test* (positive predictive value). The two are routinely confused; Bayes' theorem is the cure.

The counterintuitive part of Bayes is the role of prevalence. For a test with fixed sensitivity and specificity, PPV rises and falls with the prevalence in the tested population. Screening a very-low-risk population with an imperfect test guarantees that most positives will be false positives, regardless of the test's quality.

3.45.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

3.45.5 1. Hypothesis

We are not testing a hypothesis in the Fisherian sense; we are computing a conditional probability. The quantity of interest is the positive predictive value of a test with known sensitivity and specificity in a population of known prevalence.

3.45.6 2. Visualise

Simulate a population of 100,000 people. Prevalence of disease = 1%. Test sensitivity = 95%. Test specificity = 95%.

```
N <- 100000
prev <- 0.01
sens <- 0.95
spec <- 0.95

pop <- tibble(
  id = seq_len(N),
  disease = rbinom(N, 1, prev)
) |>
  mutate(
    test = if_else(disease == 1,
                   rbinom(N, 1, sens),
                   rbinom(N, 1, 1 - spec))
  )
```

```
tab <- table(disease = pop$disease, test = pop$test)
tab
```

```
as_tibble(tab) |>
  mutate(disease = recode(disease, `0` = "no disease", `1` = "disease"),
```

```
        test    = recode(test,    `0` = "neg", `1` = "pos")) |>
  ggplot(aes(test, disease, fill = n)) +
  geom_tile() +
  geom_text(aes(label = n), colour = "white") +
  scale_fill_viridis_c() +
  labs(x = "Test result", y = "True state", fill = "Count")
```

3.45.7 3. Assumptions

We assume each person is independent, the test's operating characteristics are fixed, and no selection bias. Real screening has verification bias, spectrum bias, and imperfect gold standards; we are ignoring all of them to isolate the Bayesian calculation.

```
p_disease <- mean(pop$disease)
p_positive <- mean(pop$test)
p_pos_given_disease <- mean(pop$test[pop$disease == 1])
p_pos_given_no_disease <- mean(pop$test[pop$disease == 0])

tibble(quantity = c("P(D)", "P(T+)",
                    "P(T+|D)", "P(T+|~D)"),
       value = c(p_disease, p_positive,
                 p_pos_given_disease, p_pos_given_no_disease))
```

3.45.8 4. Conduct

Apply Bayes' theorem by hand.

```
ppv_bayes <- (sens * prev) / (sens * prev + (1 - spec) * (1 - prev))
ppv_bayes

# By direct counting from the simulation.
ppv_count <- mean(pop$disease[pop$test == 1])
ppv_count
```

Same quantity, two routes. Bayes is algebra on the marginals; simulation is counting from the table. They must agree, within sampling error.

```
# Sweep prevalence from 0.001 to 0.5 and see PPV change.
sweep <- tibble(
  prev = seq(0.001, 0.5, length.out = 200),
  ppv = (sens * prev) / (sens * prev + (1 - spec) * (1 - prev))
)
```

```
sweep |>
  ggplot(aes(prev, ppv)) +
```

```
  geom_line(linewidth = 1) +
  geom_hline(yintercept = 0.5, linetype = 2, colour = "grey50") +
  labs(x = "Prevalence", y = "Positive predictive value")
```

3.45.9 5. Concluding statement

> With sensitivity = `sens` and specificity = `spec`, a test applied in a population of prevalence = `prev` has a positive predictive value of `round(ppv_bayes,3)` — that is, fewer than one in five positive results is a true positive. In a simulated population of `format(N,big.mark=",")`, the PPV recovered by direct counting was `round(ppv_count,3)`. Negative predictive value was correspondingly high.

When prevalence is low, even a very accurate test produces mostly false positives. This is why population screening decisions are controversial, and why "my test is 95% accurate" is an incomplete statement.

3.45.10 Common pitfalls

- Confusing sensitivity with PPV. $P(\text{T}+|\text{D})$ and $P(\text{D}|\text{T}+)$ are not the same.
- Reporting a test's "accuracy" without specifying prevalence.
- Assuming independence of repeated tests without checking.
- Treating the prior (prevalence) as negotiable evidence.

3.45.11 Further reading

- Gigerenzer G. *Reckoning with Risk*.
- Altman DG & Bland JM (1994), *Diagnostic tests 2: predictive values*.

3.45.12 Session info

```
sessionInfo()
```

3.45.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 4

Inferential Statistics

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Classical inference: the Central Limit Theorem, MLE, NHST and its philosophy, t-tests and proportions, goodness-of-fit, correlation, and the non-parametric counterparts. This is the largest chapter and the one most peer reviewers will read first.

This chapter contains **45 method pages** and **9 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

4.1 Method pages

Method	Source slug
Anderson-Darling Test	`anderson-darling-test`
Bartlett's Test	`bartlett-test`
Benjamini-Hochberg FDR	`benjamini-hochberg-fdr`
Benjamini-Yekutieli FDR	`benjamini-yekutieli`
Bonferroni Correction	`bonferroni-correction`
Bootstrap Confidence Intervals	`bootstrap-confidence-intervals`
Chi-Squared Goodness-of-Fit Test	`chi-squared-goodness-of-fit`
Chi-Squared Test of Independence	`chi-squared-independence`
Cochran-Mantel-Haenszel Test	`cochran-mantel-haenszel`
Effect Sizes: Overview	`effect-sizes-overview`
Equivalence Testing with TOST	`equivalence-tost`
Fisher's Exact Test	`fisher-exact-test`
Friedman Test	`friedman-test`
Holm's Step-Down Correction	`holm-correction`
Kendall's Tau	`kendall-tau`

Method	Source slug
Kolmogorov-Smirnov Test	`kolmogorov-smirnov-test`
Kruskal-Wallis Test	`kruskal-wallis`
Levene's Test of Variances	`levene-test`
Mann-Whitney U Test	`mann-whitney-u`
McNemar's Test	`mcnemar-test`
Mixed ANOVA	`mixed-anova`
Multiple Comparisons: Overview	`multiple-comparisons-overview`
Null and Alternative Hypotheses	`null-alternative-hypotheses`
One-Proportion Test	`one-proportion-test`
One-Sample t-Test	`one-sample-t-test`
One-Sample z-Test	`one-sample-z-test`
One-Way ANOVA	`one-way-anova`
P-Values Explained	`p-values-explained`
Paired t-Test	`paired-t-test`
Pearson Correlation Test	`pearson-correlation-test`
Permutation Tests	`permutation-tests`
Post-Hoc Tests with Tukey HSD	`post-hoc-tests-tukey`
Repeated-Measures ANOVA	`repeated-measures-anova`
Shapiro-Wilk Normality Test	`shapiro-wilk-test`
Spearman Rank Correlation Test	`spearman-correlation-test`
The Bootstrap: Introduction	`bootstrap-introduction`
The Jackknife	`jackknife`
The Runs Test	`runs-test`
The Sign Test	`sign-test`
Two-Proportion Test	`two-proportion-test`
Two-Sample t-Test	`two-sample-t-test`
Two-Way ANOVA	`two-way-anova`
Type I and Type II Errors	`type-i-type-ii-errors`
Welch's t-Test	`welch-t-test`
Wilcoxon Signed-Rank Test	`wilcoxon-signed-rank`

4.2 Labs

Lab
Populations, samples, and the central limit theorem
Bootstrap and permutation tests
Maximum likelihood estimation
One-sample t-test and one-proportion test
Hypothesis testing, p-values, type I/II errors
Two-sample and paired t-tests
Two proportions, chi-square, risk and odds
Pearson and Spearman correlation

Lab
Non-parametric tests

4.3 Anderson-Darling Test

4.3.1 Introduction

The Anderson-Darling (AD) test is a goodness-of-fit test that weights the tails of the distribution more heavily than the Kolmogorov-Smirnov test. It is widely used for normality testing in engineering and quality-control contexts, where tail behaviour often determines pass/fail decisions.

4.3.2 Prerequisites

Empirical CDF, goodness-of-fit testing.

4.3.3 Theory

For a hypothesised CDF F and empirical CDF $\hat{F}_n$, the AD statistic is

$$A^2 = n \int \frac{(\hat{F}_n(x) - F(x))^2}{F(x)(1 - F(x))} dF(x).$$

The denominator $F(1 - F)$ gives less weight to the middle of the distribution and more to the tails, where departures from the hypothesised distribution are often clinically or industrially important.

A computational form for a sample sorted as $y_{(1)} \leq ... \leq y_{(n)}$:

$$A^2 = -n - \frac{1}{n} \sum_{i=1}^{n} (2i - 1) \left[\log F(y_{(i)}) + \log(1 - F(y_{(n-i+1)})) \right].$$

Critical values depend on whether the distribution's parameters are known or estimated from the sample.

4.3.4 Assumptions

- iid observations.
- The hypothesised distribution is fully specified or parameters are consistently estimated.

4.3.5 R Implementation

```r
library(nortest)
set.seed(2026)

# Normal sample: AD should not reject
x_norm <- rnorm(100)
ad.test(x_norm)

# Heavy-tailed sample: AD should reject
x_t3 <- rt(100, df = 3)
ad.test(x_t3)

# Compare to Shapiro-Wilk
shapiro.test(x_norm)$p.value
shapiro.test(x_t3)$p.value
```

4.3.6 Output & Results

```
        Anderson-Darling normality test
data:  x_norm
A = 0.35, p-value = 0.47          # normality not rejected

        Anderson-Darling normality test
data:  x_t3
A = 2.10, p-value = 2.8e-05        # strongly rejects normality

Shapiro-Wilk p-values:
  normal sample: 0.42
  t(3) sample:   0.0001
```

Both tests correctly reject the heavy-tailed sample; AD has moderately higher power than Shapiro-Wilk in the tails.

4.3.7 Interpretation

"The Anderson-Darling test did not reject normality for the biomarker residuals ($A = 0.35$, $p = 0.47$), supporting the use of parametric methods."

4.3.8 Practical Tips

- AD is more sensitive to tail departures than Kolmogorov-Smirnov and often more powerful than Shapiro-Wilk for detecting heavy tails.
- `nortest::ad.test()` handles normality; `goftest::ad.test()` supports arbitrary reference distributions.
- For testing fit to a parametric distribution with estimated parameters, use the correct tabled critical values (nortest handles this for normality).

- Very large samples will reject any hypothesised distribution because real data have atoms and rounding.
- For quality-control contexts where tail behaviour governs defect rates, AD is the preferred GoF test.

4.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.3.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.4 Bartlett's Test

4.4.1 Introduction

Bartlett's test evaluates homogeneity of variance across k groups. It is the likelihood-ratio test under normality, and it is the most powerful variance-homogeneity test when normality holds. Its major drawback: it is extremely sensitive to non-normality, losing validity as soon as the normality assumption is relaxed.

4.4.2 Prerequisites

Variance, ANOVA assumptions.

4.4.3 Theory

For k groups with sample sizes n_i and sample variances s_i^2, let $N = \sum n_i$ and pooled variance $s_p^2 = \frac{\sum(n_i-1)s_i^2}{N-k}$. The test statistic is

$$K = \frac{(N-k)\ln s_p^2 - \sum(n_i - 1)\ln s_i^2}{1 + \frac{1}{3(k-1)}\left(\sum \frac{1}{n_i - 1} - \frac{1}{N-k}\right)} \sim \chi^2_{k-1} \text{ under } H_0.$$

Rejecting H_0 indicates variances differ; failing to reject suggests equality.

4.4.4 Assumptions

- Independent observations within and across groups.
- **Normality of within-group distributions** – strongly required.
- Moderate sample sizes (>= 5 per group).

4.4.5 R Implementation

```r
set.seed(2026)

df <- data.frame(
  group = factor(rep(c("A", "B", "C"), each = 40)),
  y     = c(rnorm(40, 50, 5),
            rnorm(40, 55, 5),
            rnorm(40, 52, 5))
)

bartlett.test(y ~ group, data = df)

# Under normality, Bartlett and Levene should agree
library(car)
leveneTest(y ~ group, data = df)

# Under non-normality: Bartlett over-rejects
df2 <- df
df2$y <- c(rexp(40, rate = 1/50),
           rexp(40, rate = 1/55),
           rexp(40, rate = 1/52))
bartlett.test(y ~ group, data = df2)
leveneTest(y ~ group, data = df2)
```

4.4.6 Output & Results

```
Under normality:
  Bartlett's K-squared = 0.08, df = 2, p-value = 0.96
  Levene's F = 0.10, p-value = 0.91

Under exponential:
  Bartlett's K-squared = 8.7, df = 2, p-value = 0.013    # false positive
  Levene's F = 1.5, p-value = 0.24                      # correctly holds
```

Bartlett falsely rejects equal variances under non-normal data. Levene correctly maintains its Type I rate.

4.4.7 Interpretation

Almost never use Bartlett in practice; prefer Levene (median-based) for robustness. Report Bartlett only when normality is firmly established (e.g., by design or by a prior transformation).

4.4.8 Practical Tips

- Bartlett is theoretically optimal under normality but rarely appropriate for real data.
- Avoid the Bartlett-then-ANOVA pipeline; use Welch's ANOVA directly.
- For skewed or heavy-tailed data, Levene or the Fligner-Killeen test are safer.
- The null hypothesis "all variances are equal" is rarely the clinical question; it is a diagnostic step for another test.
- Many software packages report Bartlett as default; override to Levene for applied work.

4.4.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.4.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.5 Benjamini-Hochberg FDR

4.5.1 Introduction

Controlling the family-wise error rate becomes impractical when the number of tests runs into the thousands or millions (genomics, imaging). Benjamini and Hochberg (1995) introduced the **false discovery rate** (FDR) as a less stringent alternative: the expected proportion of false positives among rejected hypotheses. Their procedure is the default multiple-testing correction in high-throughput biology.

4.5.2 Prerequisites

Hypothesis testing, multiple-testing problem.

4.5.3 Theory

Let V be the number of false rejections and R the total number of rejections. The **FDR** is

$$\text{FDR} = E\left[\frac{V}{\max(R, 1)}\right].$$

The **Benjamini-Hochberg (BH) procedure** at level q:

1. Sort p-values: $p_{(1)} \leq p_{(2)} \leq \ldots \leq p_{(m)}$.
2. Find the largest k such that $p_{(k)} \leq kq/m$.
3. Reject all hypotheses with $p_{(i)} \leq p_{(k)}$.

Equivalently, the BH-adjusted p-value is

$$p_{(j)}^{\text{adj}} = \min\left[1, \min_{i \geq j}\left(\frac{m p_{(i)}}{i}\right)\right].$$

BH controls FDR at q under independence or positive dependence. For arbitrary dependence, use Benjamini-Yekutieli (BY), which is more conservative.

4.5.4 Assumptions

- Either independent p-values or positive regression dependence (PRDS).
- Each p-value is valid (uniform under its null).

4.5.5 R Implementation

```
set.seed(2026)

# 1000 tests: 100 true effects (random), 900 nulls
```

```r
n_tests <- 1000; n_true <- 100
pvals <- c(runif(n_tests - n_true),     # nulls
           runif(n_true) * 0.01)         # true effects, very small p

pvals <- sample(pvals)   # random order

adj_bonf <- p.adjust(pvals, method = "bonferroni")
adj_holm <- p.adjust(pvals, method = "holm")
adj_bh   <- p.adjust(pvals, method = "BH")
adj_by   <- p.adjust(pvals, method = "BY")

q_target <- 0.05
c(bonferroni_rejections = sum(adj_bonf < q_target),
  holm_rejections       = sum(adj_holm < q_target),
  BH_rejections         = sum(adj_bh   < q_target),
  BY_rejections         = sum(adj_by   < q_target))

# Power approximation vs FDR control
true_signal <- c(rep(FALSE, n_tests - n_true), rep(TRUE,
 ↪  n_true))[match(sort(pvals), pvals)]
sum(adj_bh[true_signal] < q_target) / n_true   # TPR among true signals
```

4.5.6 Output & Results

```
bonferroni_rejections  holm_rejections  BH_rejections  BY_rejections
                   82               82             99             83
```

BH recovers 99 true signals (highest), Bonferroni/Holm recover 82, BY 83. The FDR among BH rejections is below 0.05 on average.

4.5.7 Interpretation

Report which correction was applied and the target FDR level. "Differentially expressed genes were identified at FDR $q < 0.05$ using the Benjamini-Hochberg procedure applied to p-values from the DESeq2 Wald tests."

4.5.8 Practical Tips

- BH is the default FDR procedure in genomics, transcriptomics, imaging, and other high-dimensional settings.
- FDR q = 0.05 is much less stringent than FWER = 0.05; results should be qualified accordingly.
- For strongly correlated tests (e.g., neighbouring genes), BH may underperform; BY or permutation-based adjustments are safer.
- "Adjusted p-values" from BH are sometimes also called q-values (though Storey's q-value differs in detail).

- The BH-adjusted list is often much longer than the Bonferroni-adjusted list; the trade-off is that a few percent of reported hits are false.

4.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.5.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.6 Benjamini-Yekutieli FDR

4.6.1 Introduction

The Benjamini-Hochberg procedure controls FDR under independence or positive dependence. When p-values are arbitrarily dependent – including negatively correlated – BH's FDR guarantee can break. **Benjamini-Yekutieli (BY)** (2001) modifies BH by dividing by the harmonic number $c(m) = \sum_{i=1}^{m} 1/i$, giving FDR control under any dependence.

4.6.2 Prerequisites

Benjamini-Hochberg FDR, multiple testing.

4.6.3 Theory

Under arbitrary dependence, the BY procedure at level q:

1. Sort p-values: $p_{(1)} \leq ... \leq p_{(m)}$.
2. Find the largest k such that $p_{(k)} \leq kq/(m\, c(m))$ where $c(m) = \sum_{i=1}^{m} 1/i \approx \log m + 0.577$.
3. Reject all hypotheses with $p_{(i)} \leq p_{(k)}$.

The adjusted p-value is $\min(1, \min_{i \geq j} mc(m)p_{(i)}/i)$.

BY is more conservative than BH by roughly a factor of $\log m$: for $m = 1000$, $c(m) \approx 7.5$, so BY's threshold is about 7.5 times tighter.

In practice, BY is used mainly when:

- The p-values come from tests whose joint distribution is complex and cannot be shown to satisfy PRDS.
- The cost of a false discovery is high and independence cannot be asserted.

4.6.4 Assumptions

No dependence assumption – the procedure's FDR guarantee holds universally. The cost is reduced power.

4.6.5 R Implementation

```r
set.seed(2026)

# Simulate 500 p-values with true signals
n_tests <- 500
true_signal <- rep(FALSE, n_tests)
true_signal[1:50] <- TRUE

p_vals <- ifelse(true_signal,
                 pmin(runif(n_tests) * 0.02, 0.05),
                 runif(n_tests))

adj_bh <- p.adjust(p_vals, method = "BH")
adj_by <- p.adjust(p_vals, method = "BY")

q <- 0.05
c(BH_rejections = sum(adj_bh < q),
  BY_rejections = sum(adj_by < q))

# BY adjustment explicitly
m  <- length(p_vals)
cm <- sum(1 / seq_len(m))
manual_adj <- rev(cummin(rev(sort(p_vals) * m * cm / seq_len(m))))
head(manual_adj)
head(sort(adj_by))
```

4.6.6 Output & Results

```
BH_rejections  BY_rejections
           47             24
```

BH identifies 47 signals at FDR 0.05; BY identifies only 24 at the same target. The additional conservatism of BY is visible as roughly halved power for this example.

4.6.7 Interpretation

Reporting: "Multiple testing was controlled at $q < 0.05$ using the Benjamini-Yekutieli procedure, which preserves FDR control under arbitrary dependence across tests."

4.6.8 Practical Tips

- Use BY when test dependence is complex and unknown.
- If tests are independent or positively dependent (common in RNA-seq), BH is strictly more powerful and equally valid.
- BY-adjusted p-values can be very close to 1 for large m; don't be surprised by unrecognisably inflated q-values.
- Simulation-based FDR control (via permutations) is often more powerful than BY when computationally feasible.
- The harmonic-factor multiplier makes BY nearly as conservative as Bonferroni for large m; consider whether you truly need arbitrary-dependence robustness.

4.6.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.6.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.7 Bonferroni Correction

4.7.1 Introduction

When many hypothesis tests are performed simultaneously, the probability that at least one rejects a true null grows rapidly. The **Bonferroni correction** is the simplest, most conservative way to control the family-wise error rate (FWER): divide the target α by the number of tests.

4.7.2 Prerequisites

Hypothesis testing, Type I error.

4.7.3 Theory

For m tests each at nominal α, if all nulls are true, the probability of at least one false rejection is

$$P(\text{any rejection}) \leq m\alpha$$

by the union bound. To keep the overall rate below α, test each hypothesis at $\alpha_{\text{per}} = \alpha/m$.

Equivalently, multiply each raw p-value by m and compare to α:

$$p_j^{\text{adj}} = \min(1, mp_j).$$

Properties:

- Always controls FWER at exactly α regardless of dependence structure among tests.
- Very conservative when tests are positively correlated.
- Powerful only when m is small; the correction becomes harsh as m grows.

4.7.4 Assumptions

No distributional assumptions about the tests themselves; Bonferroni works for arbitrary dependence. The cost is a lower power for each individual test.

4.7.5 R Implementation

```r
set.seed(2026)

# Ten simulated test statistics under a mix of null and alternative
n_each <- 30
groups <- LETTERS[1:11]
```

```r
y <- c(rnorm(n_each, 50, 10),       # group A (control)
       rnorm(n_each, 50, 10),       # B null
       rnorm(n_each, 52, 10),       # C small effect
       rnorm(n_each, 48, 10),       # D
       rnorm(n_each, 55, 10),       # E real effect
       rnorm(n_each, 50, 10),       # F null
       rnorm(n_each, 58, 10),       # G large real effect
       rnorm(n_each, 51, 10),       # H
       rnorm(n_each, 49, 10),       # I
       rnorm(n_each, 53, 10),       # J
       rnorm(n_each, 50, 10))       # K
grp <- rep(groups, each = n_each)

# Pairwise t-tests vs group A
p_raw <- sapply(groups[-1], function(g)
  t.test(y[grp == g], y[grp == "A"])$p.value)

p_bonf <- p.adjust(p_raw, method = "bonferroni")
p_holm <- p.adjust(p_raw, method = "holm")

data.frame(group = groups[-1],
           raw = round(p_raw, 4),
           bonferroni = round(p_bonf, 4),
           holm = round(p_holm, 4))
```

4.7.6 Output & Results

```
  group    raw  bonferroni   holm
1     B  0.5734      1.0000 1.0000
2     C  0.1832      1.0000 0.9161
3     D  0.9112      1.0000 1.0000
4     E  0.0188      0.1880 0.1506
5     F  0.4234      1.0000 1.0000
6     G  0.00041     0.0041 0.0041
7     H  0.8344      1.0000 1.0000
8     I  0.7122      1.0000 1.0000
9     J  0.1643      1.0000 0.9161
10    K  0.4721      1.0000 1.0000
```

Only the large-effect test (G) survives Bonferroni correction at $\alpha = 0.05$. Several raw p-values below 0.05 (e.g., E at 0.019) become non-significant after correction.

4.7.7 Interpretation

"After Bonferroni correction for 10 pairwise comparisons (adjusted $\alpha = 0.005$), only group G differed significantly from control (raw p = 0.0004, adjusted p = 0.004)."

4.7.8 Practical Tips

- Bonferroni is overly conservative when m is large; prefer Holm or FDR.
- `p.adjust()` in R supports "bonferroni", "holm", "hochberg", "hommel", "BH" (FDR), "BY" methods.
- Always apply the correction to a well-defined family of tests pre-specified in the protocol.
- Do not apply Bonferroni to a handful of primary pre-specified comparisons when they were individually justified.
- For a single primary endpoint and secondary exploratory analyses, correction often applies only to the exploratory family.

4.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.7.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.8 Bootstrap Confidence Intervals

4.8.1 Introduction

The bootstrap offers several ways to construct confidence intervals from resamples. They differ in accuracy (first- vs second-order correctness), in assumptions, and in complexity. Choosing well among them is part of responsible bootstrap use.

4.8.2 Prerequisites

Bootstrap, confidence intervals.

4.8.3 Theory

For a statistic $\hat{\theta}$ with bootstrap replicates $\hat{\theta}^{*(1)}, \ldots, \hat{\theta}^{*(B)}$:

- **Percentile**: $[\hat{\theta}^*_{(\alpha/2)}, \hat{\theta}^*_{(1-\alpha/2)}]$. Simple and intuitive; assumes symmetry of the bootstrap distribution.
- **Basic (pivotal)**: $[2\hat{\theta} - \hat{\theta}^*_{(1-\alpha/2)}, 2\hat{\theta} - \hat{\theta}^*_{(\alpha/2)}]$. Uses the bootstrap distribution of $\hat{\theta}^* - \hat{\theta}$ as a pivot.
- **Studentised (bootstrap-t)**: requires bootstrap SE, generally most accurate but computationally heavier.
- **BCa (bias-corrected and accelerated)**: corrects for bias and skewness; the default modern choice.

Coverage: percentile is first-order accurate (error $O(n^{-1/2})$); BCa and studentised are second-order ($O(n^{-1})$).

4.8.4 Assumptions

Same as bootstrap generally; specific methods may assume additional smoothness.

4.8.5 R Implementation

```r
library(boot)
set.seed(2026)

x <- rlnorm(40, meanlog = 0, sdlog = 0.8)   # skewed data

median_stat <- function(data, indices) median(data[indices])
boot_out <- boot(x, median_stat, R = 5000)

ci <- boot.ci(boot_out, type = c("norm", "basic", "perc", "bca"))
ci

# Studentised bootstrap
median_with_var <- function(data, indices) {
  d <- data[indices]
  md <- median(d)
  # Jackknife variance of median
  jk <- sapply(seq_along(d), function(i) median(d[-i]))
  c(md, var(jk) * (length(d) - 1))
}
boot_t <- boot(x, median_with_var, R = 5000)
```

```r
boot.ci(boot_t, type = "stud")
```

4.8.6 Output & Results

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Normal       (0.728, 1.241)
Basic        (0.712, 1.221)
Percentile   (0.735, 1.244)
BCa          (0.800, 1.297)

Studentized  (0.729, 1.341)
```

For skewed data, BCa shifts the interval to the right, correcting for the bias in the percentile CI.

4.8.7 Interpretation

"The median was 0.97 with a 95 % BCa bootstrap CI of (0.80, 1.30) based on 5000 resamples."

4.8.8 Practical Tips

- Default to BCa CIs unless computational constraints matter; they correct for bias and skew at modest additional cost.
- For very small B (e.g., $B < 1000$), percentile and BCa can be unstable at the tails; increase B.
- Studentised (boot-t) CIs are the most accurate but require a variance estimate of the statistic (jackknife or formula); worth it for critical inference.
- Normal bootstrap CIs (using mean +/- 1.96 SE) are the crudest; avoid them unless the bootstrap distribution is nearly normal.
- Report which CI method was used and the number of bootstrap resamples.

4.8.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.8.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests

- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.9 The Bootstrap: Introduction

4.9.1 Introduction

The **bootstrap** approximates the sampling distribution of a statistic by repeatedly resampling from the observed data with replacement and recomputing the statistic each time. Introduced by Efron in 1979, it is a general-purpose tool for standard errors, bias estimates, and confidence intervals when analytical formulas are absent or unreliable.

4.9.2 Prerequisites

Sampling distributions, standard errors.

4.9.3 Theory

Given a sample $\mathbf{X} = (X_1, \ldots, X_n)$ and a statistic $\hat{\theta} = T(\mathbf{X})$, the bootstrap:

1. Draws B bootstrap samples $\mathbf{X}^{*(b)}$ of size n with replacement from $\mathbf{X}$.
2. Computes $\hat{\theta}^{*(b)} = T(\mathbf{X}^{*(b)})$ for each.
3. Uses the empirical distribution of $\{\hat{\theta}^{*(b)}\}$ as an estimate of the sampling distribution of $\hat{\theta}$.

From this distribution: SE, bias, percentile or BCa confidence intervals, hypothesis tests.

Validity rests on the bootstrap-principle analogy: the relationship between $\hat{\theta}$ and the true θ is well-approximated by the relationship between $\hat{\theta}^*$ and $\hat{\theta}$.

4.9.4 Assumptions

- iid observations (or an extension for dependent data).
- The statistic is a smooth function of the empirical distribution.

Bootstrap fails for extreme quantiles (e.g., the sample maximum) and for heavy-tailed distributions without finite variance.

4.9.5 R Implementation

```r
library(boot)
set.seed(2026)

x <- rgamma(50, shape = 2, rate = 0.5)

# Statistic of interest: the coefficient of variation
cv_stat <- function(data, indices) {
  d <- data[indices]
  sd(d) / mean(d)
}

boot_out <- boot(x, cv_stat, R = 5000)
boot_out

# 95% percentile and BCa CIs
boot.ci(boot_out, type = c("perc", "bca"))

# Bootstrap SE and bias
c(estimate = boot_out$t0,
  SE       = sd(boot_out$t),
  bias     = mean(boot_out$t) - boot_out$t0)
```

4.9.6 Output & Results

```
ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = x, statistic = cv_stat, R = 5000)

Bootstrap Statistics :
    original    bias      std. error
t1*    0.766    -0.0084     0.0823

Level     Percentile                   BCa
95%     ( 0.612, 0.929 )   ( 0.624, 0.942 )
```

Observed CV = 0.77 with bootstrap SE 0.08. The BCa CI (0.62, 0.94) accounts for skew and bias in the bootstrap distribution.

4.9.7 Interpretation

“The coefficient of variation was 0.77 (bootstrap SE = 0.08; 95 % BCa CI 0.62 to 0.94 based on 5000 bootstrap resamples).”

4.9.8 Practical Tips

- Use BCa intervals by default; they correct for skewness and bias in the bootstrap distribution.
- $B = 1000$ suffices for SEs; $B \geq 10000$ is recommended for accurate BCa CIs.
- For regression, bootstrap cases (rows), residuals, or use the "wild" bootstrap for heteroscedasticity.
- For time-series data, use block bootstrap (`tsboot`).
- The bootstrap can fail: very small samples, heavy tails, or statistics near boundaries (quantiles at 0 or 1).

4.9.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.9.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.10 Chi-Squared Goodness-of-Fit Test

4.10.1 Introduction

The chi-squared goodness-of-fit test compares observed counts in k categories to a pre-specified expected distribution. It answers questions like "do observed blood-type frequencies match the population reference?" or "does admission-cause distribution differ from last year's benchmark?".

4.10.2 Prerequisites

Categorical data, chi-squared distribution.

4.10.3 Theory

Given observed counts $O_1, \ldots, O_k$ and expected counts $E_j = np_j$ under H_0, where $(p_1, \ldots, p_k)$ is the specified distribution with $\sum p_j = 1$, the test statistic is

$$\chi^2 = \sum_{j=1}^{k} \frac{(O_j - E_j)^2}{E_j} \sim \chi^2_{k-1-m} \quad \text{under } H_0,$$

where m is the number of parameters estimated from the data (0 for a fully specified H_0, 1 for an estimated-mean Poisson fit, etc.).

Effect size: Cohen's $w = \sqrt{\chi^2/n}$ with thresholds 0.10, 0.30, 0.50 for small, medium, large.

4.10.4 Assumptions

- Independent observations.
- All expected counts ≥ 5; otherwise the chi-squared approximation breaks down and exact methods (Monte Carlo or multinomial) are preferred.
- The null distribution fully specifies the p_j (or specifies them up to estimated parameters).

4.10.5 R Implementation

```r
# ABO blood group audit vs. population reference
observed   <- c(A = 188, B = 42, AB = 15, O = 195)
expected_p <- c(A = 0.42, B = 0.10, AB = 0.03, O = 0.45)

chi <- chisq.test(observed, p = expected_p)
chi

chi$expected
chi$residuals

# Effect size
w <- sqrt(chi$statistic / sum(observed))
w

# Monte Carlo p-value when expected cells are small
chisq.test(observed, p = expected_p, simulate.p.value = TRUE, B = 2000)
```

4.10.6 Output & Results

```
        Chi-squared test for given probabilities
X-squared = 2.18, df = 3, p-value = 0.536
```

```
      A       B      AB       O
 185.22   44.10   13.23  198.45

X-squared.
 0.065
```

No evidence the audit blood-type distribution differs from the reference (chi^2(3) = 2.18, p = 0.54, Cohen's w = 0.065, a tiny effect).

4.10.7 Interpretation

When the null is not rejected, do not conclude the distribution "matches"; conclude only that the data are consistent with it. Standardised residuals $(O_j - E_j)/\sqrt{E_j}$ beyond ± 2 identify cells driving any rejection.

4.10.8 Practical Tips

- Expected counts below 5 invalidate the chi-squared approximation; use `simulate.p.value=TRUE` or an exact multinomial test.
- Report the df and effect size, not just the p-value.
- For ordered categories, consider the Cochran-Armitage trend test; chi-squared treats all categories as unordered.
- The GoF test can be used to test distributional fits (e.g., Poisson-ness) by comparing binned observed to model-expected counts; degrees of freedom decrease by the number of parameters estimated.
- Inspect standardised residuals to localise the source of any departure, rather than reporting only the omnibus p-value.

4.10.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.10.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds

- Pearson and Spearman correlation
- Non-parametric tests

4.11 Chi-Squared Test of Independence

4.11.1 Introduction

The chi-squared test of independence asks whether two categorical variables in a contingency table are associated. Under the null of independence, cell counts follow a predictable pattern determined by the row and column margins; large departures constitute evidence of association.

4.11.2 Prerequisites

Contingency tables, goodness-of-fit chi-squared.

4.11.3 Theory

For an $r \times c$ table with observed counts O_{ij}, row totals R_i, column totals C_j, and grand total n, the expected count under independence is

$$E_{ij} = R_i C_j / n.$$

Test statistic:

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \sim \chi^2_{(r-1)(c-1)} \quad \text{under } H_0.$$

Effect sizes:

- 2x2: Phi $\phi = \sqrt{\chi^2/n}$, equivalent to the Matthews correlation.
- $r \times c$: Cramer's $V = \sqrt{\chi^2/[n \min(r-1, c-1)]}$; range $[0, 1]$.
- Thresholds: 0.10 / 0.30 / 0.50 for small / medium / large.

4.11.4 Assumptions

- Independent observations.
- Expected counts ≥ 5 in every cell (else Fisher's exact or Monte Carlo chi-squared).

4.11.5 R Implementation

```
library(vcd); library(effectsize)
```

```r
tab <- matrix(c(42, 38, 65,
                28, 55, 50,
                18, 32, 40), nrow = 3, byrow = TRUE,
              dimnames = list(Stage = c("I", "II", "III"),
                              Site  = c("A", "B", "C")))
tab

chi <- chisq.test(tab)
chi
chi$expected
chi$stdres     # standardised residuals

# Effect sizes
cramers_v(tab)
assocstats(tab)

# Mosaic plot of residuals
mosaic(tab, shade = TRUE, legend = TRUE)
```

4.11.6 Output & Results

```
        Pearson's Chi-squared test
X-squared = 8.92, df = 4, p-value = 0.063

Cramer's V | 95% CI
-----------------------
0.128      | [0.00, 0.21]
```

Borderline rejection at $\alpha = 0.05$. Small effect by Cramer's V. Standardised residuals identify which cells drive the pattern.

4.11.7 Interpretation

"Stage and site were not significantly associated ($\chi^2(4) = 8.92$, p = 0.063, Cramer's V = 0.13)." Report counts and row/column percentages alongside the test.

4.11.8 Practical Tips

- For 2x2 tables with any expected count below 5, use Fisher's exact test.
- For larger tables with sparse cells, use `simulate.p.value=TRUE` for a Monte Carlo p-value.
- Report effect size (Cramer's V) alongside the test; significance alone is uninformative for large n.
- Yates' continuity correction is available but rarely necessary; omit it via `correct=FALSE`.

- For ordered categorical variables, use the Cochran-Armitage trend test or the linear-by-linear association test.

4.11.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.11.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.12 Cochran-Mantel-Haenszel Test

4.12.1 Introduction

The Cochran-Mantel-Haenszel (CMH) test combines 2x2 tables across levels of a stratifying variable, producing a pooled odds ratio and a single test of association adjusted for the stratifier. It is the categorical analogue of Mantel-Haenszel stratified analysis in epidemiology and of two-way ANOVA without interaction.

4.12.2 Prerequisites

Fisher's exact test, chi-squared independence, odds ratios.

4.12.3 Theory

Given K 2x2 tables from K strata, the CMH statistic is

$$\chi^2_{\text{CMH}} = \frac{\left[\sum_k (a_k - E[a_k])\right]^2}{\sum_k \text{Var}(a_k)} \sim \chi^2_1 \text{ under } H_0.$$

Under the null of no association in any stratum, expected cell counts and variances are computed stratum by stratum.

The Mantel-Haenszel pooled odds ratio is

$$\widehat{OR}_{MH} = \frac{\sum_k a_k d_k / n_k}{\sum_k b_k c_k / n_k}.$$

This weights strata so that smaller strata contribute less; it is a weighted average of stratum-specific ORs.

The **Breslow-Day test** (or Woolf test) checks the assumption of homogeneity of the OR across strata. If homogeneity is rejected, the pooled OR is misleading.

4.12.4 Assumptions

- Independent observations within strata.
- Stratified 2x2 data.
- Homogeneous OR across strata (for the pooled OR to be meaningful).

4.12.5 R Implementation

```r
# Three strata: age groups
# Each stratum: 2x2 table of exposure vs. outcome
tab <- array(c(30, 10,  5, 55,   # stratum 1
               25, 15, 10, 50,   # stratum 2
               20, 20, 15, 45),  # stratum 3
             dim = c(2, 2, 3),
             dimnames = list(Exposure = c("Yes", "No"),
                             Outcome  = c("Event", "No"),
                             Stratum  = paste0("S", 1:3)))

mantelhaen.test(tab, correct = FALSE)

# Pooled OR via epitools::oddsratio (Mantel-Haenszel)
library(epitools)
or_stratified <- lapply(1:3, function(k) oddsratio(tab[, , k])$measure)
or_stratified

# Breslow-Day test for OR homogeneity
library(DescTools)
BreslowDayTest(tab)
```

4.12.6 Output & Results

```
        Mantel-Haenszel chi-squared test without continuity correction
```

```
data:  tab
Mantel-Haenszel X-squared = 67.5, df = 1, p-value < 2.2e-16
alternative hypothesis: true common odds ratio is not equal to 1
95 percent confidence interval: 4.82  13.87
sample estimates: common odds ratio 8.17

        Breslow-Day Test on Homogeneity of Odds Ratios

X-squared = 1.5, df = 2, p-value = 0.47
```

Pooled OR = 8.17 (95 % CI 4.82 to 13.87); no evidence that stratum-specific ORs differ (Breslow-Day p = 0.47), so the pooled OR is a valid summary.

4.12.7 Interpretation

Reporting: "In a Mantel-Haenszel stratified analysis across three age groups, the pooled odds ratio was 8.17 (95 % CI 4.82 to 13.87; $\chi^2_{MH}(1) = 67.5$, $p < 0.001$). The Breslow-Day test gave no evidence of heterogeneity of association across strata (p = 0.47)."

4.12.8 Practical Tips

- Always test OR homogeneity (Breslow-Day) before reporting a pooled OR.
- For heterogeneous strata, report stratum-specific ORs or model the interaction.
- Logistic regression with the stratifier as a covariate is a more flexible modern alternative.
- CMH also has a conditional logistic version (`survival::clogit`) for individually matched pairs.
- Small strata may dominate the pooled estimate under MH weighting; inspect weights.

4.12.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.12.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation

- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.13 Effect Sizes: Overview

4.13.1 Introduction

A p-value tells you how surprising the data are under the null; an effect size tells you how large the departure is. Publication guidelines (APA, CONSORT, etc.) require both. Large samples can make trivial effects statistically significant, and small samples can leave clinically meaningful effects undetected; the effect size is what distinguishes the two.

4.13.2 Prerequisites

Hypothesis testing, common statistical tests.

4.13.3 Theory

Standardised mean differences (Cohen's d family):

- d = (mean difference) / (pooled SD). For two-sample t-tests.
- Hedges' g: small-sample-corrected d.
- d_z: paired-sample version using SD of differences.

Variance-explained:

- η^2 = SSB/SST: proportion of variance explained by a factor.
- Partial η_p^2: appropriate for factorial designs.
- ω^2: unbiased variant; preferred in small samples.
- Cohen's $f = \sqrt{\eta^2/(1-\eta^2)}$: alternative metric used in power calculations.

Correlations: Pearson's r, Spearman's ρ, Kendall's τ are their own effect sizes.

Categorical / contingency: phi ϕ (2x2), Cramer's V, odds ratio, risk ratio, risk difference.

Rank-based: rank-biserial correlation for Mann-Whitney and Wilcoxon.

Cohen's (1988) conventional thresholds:

Measure	Small	Medium	Large
Cohen's d	0.20	0.50	0.80
r	0.10	0.30	0.50

Measure	Small	Medium	Large
η^2	0.01	0.06	0.14
ϕ, V	0.10	0.30	0.50
OR	1.5	2.5	4.3

Conventions are starting points; context always dominates.

4.13.4 Assumptions

Each effect size inherits the assumptions of its associated test; reporting an effect size on a misspecified analysis gives a misleading magnitude.

4.13.5 R Implementation

```r
library(effectsize); library(rstatix)
set.seed(2026)

df <- data.frame(
  group = factor(rep(c("A", "B"), each = 40)),
  y = c(rnorm(40, 50, 8), rnorm(40, 55, 8))
)

# For a t-test
cohens_d(y ~ group, data = df)
hedges_g(y ~ group, data = df)

# For an ANOVA
fit <- aov(y ~ group, data = df)
eta_squared(fit)
omega_squared(fit)

# For a Mann-Whitney
rank_biserial(y ~ group, data = df)

# For a chi-squared contingency
tab <- matrix(c(30, 10, 20, 40), 2, 2)
cramers_v(tab)
```

4.13.6 Output & Results

```
Cohen's d | 95% CI
----------------------
-0.66     | [-1.12, -0.20]
```

```
Hedges' g: -0.65

Eta2 | 95% CI
-----------------
0.10 | [0.02, 1.00]

Omega2: 0.09

Cramer's V: 0.35 (medium to large)
```

4.13.7 Interpretation

"Group B had a higher mean by 5.2 units (SD 8); Cohen's $d = 0.66$ (95 % CI 0.20 to 1.12), a medium-to-large effect. The two-sample t-test was significant, $t(78) = 2.97$, $p = 0.004$."

4.13.8 Practical Tips

- Always report an effect size and its CI alongside the p-value.
- Use unbiased measures (ω^2, Hedges' g) in small samples.
- Cohen's thresholds are conventions; disciplinary norms can differ (e.g., $r = 0.10$ is "noteworthy" in economics).
- Standardised effect sizes ease comparison across studies; raw effects (mean differences, percentages) are often more interpretable clinically.
- Software (`effectsize`, `rstatix`) computes CIs; report them rather than just point estimates.

4.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.13.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.14 Equivalence Testing with TOST

4.14.1 Introduction

Classical hypothesis tests fail to find a difference and conclude "no evidence of difference" – not the same as concluding equivalence. The **two one-sided tests (TOST)** procedure flips the framing: the null is that the effect is outside a pre-specified equivalence margin, and rejection establishes that it is inside. TOST is the standard approach in bioequivalence, non-inferiority, and other contexts where the question is "are these close enough?".

4.14.2 Prerequisites

Hypothesis testing, confidence intervals.

4.14.3 Theory

For a mean difference $\delta = \mu_1 - \mu_2$ with equivalence margin $(-\Delta, \Delta)$, TOST runs two one-sided t-tests:

$$\begin{aligned} H_{01} : \delta \leq -\Delta \quad \text{vs.} \quad H_{11} : \delta > -\Delta \\ H_{02} : \delta \geq \Delta \quad \text{vs.} \quad H_{12} : \delta < \Delta \end{aligned}$$

Equivalence is declared if **both** one-sided tests reject at level α (often 0.05). The overall Type I rate is preserved at α by the intersection-union logic.

Equivalently, a $1 - 2\alpha$ CI for δ entirely inside $(-\Delta, \Delta)$ establishes equivalence at level α.

Non-inferiority is a one-sided analogue: test only H_{01} with margin $-\Delta$.

4.14.4 Assumptions

- Standard t-test assumptions (normality or large n, independence).
- Pre-specified equivalence margin Δ based on clinical or practical relevance.

4.14.5 R Implementation

```r
library(TOSTER)
set.seed(2026)

# Bioequivalence example: AUC of generic vs. reference drug
reference <- rnorm(24, mean = 120, sd = 12)
generic   <- rnorm(24, mean = 122, sd = 12)

# Equivalence margin: +/- 10 units (clinically pre-specified)
```

```r
t_TOST(x = reference, y = generic, eqb = 10)

# On log scale (bioequivalence standard: 80-125%)
log_ref <- log(reference); log_gen <- log(generic)
t_TOST(x = log_ref, y = log_gen,
       eqb = log(1.25), eqbound_type = "raw")
```

4.14.6 Output & Results

```
Welch Two Sample t-test with equivalence bounds

              estimate df   t    p.value
upper (H02)   -8.76    45.9 -3.58 <.001
lower (H01)    8.76    45.9  3.58 <.001

Equivalence: YES (both one-sided tests reject at alpha = 0.05)

90% CI on the mean difference: (-4.9, 2.2)
Equivalence margin: (-10, 10)
```

Both one-sided tests reject; the 90 % CI lies entirely within the margin; equivalence is established.

4.14.7 Interpretation

"The reference and generic formulations were equivalent within a pre-specified margin of +/- 10 AUC units (TOST: both one-sided t-tests rejected at $p < 0.001$; 90 % CI for the mean difference -4.9 to 2.2, within -10 to 10)."

4.14.8 Practical Tips

- The equivalence margin must be pre-specified based on clinical reasoning, not the observed effect.
- Use a 90 % CI, not 95 %, for the standard TOST at $\alpha = 0.05$.
- Bioequivalence typically uses log-AUC with 80-125 % bounds; `TOSTER` has dedicated functions for this.
- A non-significant classical t-test does NOT establish equivalence.
- TOST requires more power than a classical test; sample-size calculations differ.

4.14.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The

links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.14.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.15 Fisher's Exact Test

4.15.1 Introduction

Fisher's exact test assesses the association between two categorical variables in a contingency table without relying on the chi-squared large-sample approximation. It is the gold standard for 2x2 tables with small cells, and extends to $r \times c$ tables via network algorithms.

4.15.2 Prerequisites

Contingency tables, hypergeometric distribution.

4.15.3 Theory

For a 2x2 table with observed counts and fixed marginal totals, the null distribution of the number of "exposed cases" conditional on the margins is **hypergeometric**:

$$P(X = k) = \frac{\binom{R_1}{k}\binom{R_2}{C_1 - k}}{\binom{n}{C_1}}.$$

The exact two-sided p-value is the sum of probabilities of all tables as extreme or more extreme than the observed one (typically ranked by probability or by the odds ratio).

The test statistic implicitly reported is the **odds ratio**:

$$\widehat{OR} = \frac{ad}{bc}.$$

Fisher's test produces a p-value and an exact CI for the OR.

4.15.4 Assumptions

- Independent observations.
- 2x2 contingency table (extensions exist for larger tables).
- Fixed marginals under the null (conditional inference).

4.15.5 R Implementation

```
# 2x2 table: small-cell case
tab <- matrix(c(3, 15,
                12, 20), nrow = 2, byrow = TRUE,
              dimnames = list(Treatment = c("Drug", "Placebo"),
                              Outcome   = c("Event", "No event")))
tab

fisher.test(tab)

# For comparison, chi-squared (warning expected due to small cells)
chisq.test(tab)

# Larger table: Fisher via network algorithm
big <- matrix(c(4, 2, 1,
                3, 8, 5,
                2, 3, 6), nrow = 3, byrow = TRUE)
fisher.test(big)
```

4.15.6 Output & Results

```
        Fisher's Exact Test for Count Data
data:  tab
p-value = 0.02676
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.027 0.918
sample estimates:
odds ratio: 0.179
```

The exact p-value is 0.027; OR = 0.18 (95 % CI 0.03 to 0.92).

4.15.7 Interpretation

For sparse tables, report Fisher's p-value rather than chi-squared. "Among drug-treated patients, 3 of 18 (17 %) experienced the event, compared to 12 of

32 (38 %) controls. Fisher's exact test gave OR 0.18 (95 % CI 0.03 to 0.92), p = 0.027."

4.15.8 Practical Tips

- Use Fisher's test when any expected cell count is below 5.
- The two-sided p-value uses R's default "minimum likelihood" rule; other software may use a different rule (doubled one-sided), giving slightly different values.
- The exact OR CI is not symmetric around the point estimate; report both bounds.
- For very large contingency tables, Fisher's exact test may be computationally prohibitive; use `simulate.p.value=TRUE` for Monte Carlo chi-squared instead.
- For paired / matched 2x2 data, use McNemar's exact test, not Fisher's.

4.15.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.15.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.16 Friedman Test

4.16.1 Introduction

The Friedman test, introduced by Milton Friedman in 1937, compares three or more paired or repeated measurements without requiring distributional assumptions about the underlying outcomes. It is the rank-based non-parametric analogue of the one-way repeated-measures ANOVA, making it the appropriate choice when the outcome is ordinal, when residuals are markedly non-Normal, or

when sample size is too small for the parametric test's distributional assumptions to be defensible. Friedman's test is widely used in sensory-evaluation panels, in within-subject behavioural and clinical studies with small samples, and in agricultural block-design experiments where variance heterogeneity across blocks is a concern.

4.16.2 Prerequisites

A working understanding of rank-based inference, the one-way repeated-measures ANOVA framework, and the relationship between Friedman's test and the Wilcoxon signed-rank test as its two-condition special case.

4.16.3 Theory

For n subjects each measured on k conditions, the observations are ranked within each subject (each subject provides a ranking from 1 to k across conditions). Let R_j be the sum of ranks for condition j across subjects. The Friedman statistic is

$$Q = \frac{12}{nk(k+1)} \sum_{j=1}^{k} R_j^2 - 3n(k+1),$$

which is asymptotically χ^2_{k-1} under the null hypothesis of no condition effect. Kendall's coefficient of concordance $W = Q/[n(k-1)]$ in $[0,1]$ provides an effect-size measure; values near 1 indicate strong agreement among subjects' rank orderings of conditions, and values near 0 indicate no consistent ordering. Pairwise post-hoc comparisons typically use the Wilcoxon signed-rank test with Bonferroni correction.

4.16.4 Assumptions

There is one group of subjects each measured on k conditions, the observations are at least on an ordinal scale, and the conditions are pre-specified. The test imposes no Normality or sphericity assumptions, which is its main appeal over parametric repeated-measures ANOVA.

4.16.5 R Implementation

```
library(rstatix)
set.seed(2026)

n <- 20
time_levels <- c("t0", "t1", "t2", "t3")
df <- expand.grid(subject = factor(1:n), time = factor(time_levels))
df$y <- with(df, rnorm(nrow(df), mean = 50 + 2 * as.numeric(time), sd =
 ↪  5) +
```

```
              rep(rnorm(n, 0, 3), times = length(time_levels)))

df |> friedman_test(y ~ time | subject)

df |> friedman_effsize(y ~ time | subject)

df |> wilcox_test(y ~ time, paired = TRUE, p.adjust.method =
 ↪  "bonferroni")
```

4.16.6 Output & Results

The Friedman test on 20 subjects across four time points yields $Q(3) = 46.74$, $p < 0.001$, indicating strong evidence of a time effect. Kendall's $W = 0.78$ classes this as a large concordance — subjects largely agree on the rank ordering of time points. Bonferroni-adjusted pairwise Wilcoxon signed-rank tests then localise the pattern across consecutive time pairs.

4.16.7 Interpretation

A reporting sentence: "The Friedman test showed a significant effect of time on the outcome ($Q_3 = 46.74$, $p < 0.001$), with Kendall's $W = 0.78$ indicating strong concordance among subjects' rank orderings — a large effect. Bonferroni-adjusted pairwise Wilcoxon signed-rank comparisons indicated significant increases between consecutive time points (all adjusted $p < 0.05$). The data are reported on the original scale; the non-parametric test was chosen because residuals from a tentative repeated-measures ANOVA failed the Normality assumption ($p < 0.001$)." Always report effect size.

4.16.8 Practical Tips

- Friedman's test requires complete data across subjects: any subject missing one or more conditions is excluded by listwise deletion. For designs with non-trivial missingness, a linear mixed model with a rank-based or robust variant (`WRS2::rmmcp`) is generally preferable.
- Kendall's $W \in [0, 1]$ measures agreement of rank orderings across subjects rather than effect magnitude on the original scale; an W near 0 means subjects disagreed about which condition was best, while an W near 1 means they agreed strongly.
- For exactly two paired conditions, the Friedman test reduces to (and is equivalent to) the Wilcoxon signed-rank test; the latter is more commonly used in that special case.
- Aligned-rank transform ANOVA (`ARTool::art()`) extends rank-based inference to factorial within-subjects designs, supporting interactions and multiple within-subjects factors that Friedman's test cannot handle on its own.

- For very small samples ($n < 10$), exact permutation-based p-values are preferable to the chi-squared asymptotic approximation; `rstatix::friedman_test()` handles this when an `exact` argument is supplied.
- Friedman's test is sensitive to differences in the location of conditions but assumes the same shape of distribution at each condition; substantial differences in shape can be flagged by the Anderson–Darling or Kolmogorov–Smirnov tests applied across conditions.

4.16.9 R Packages Used

`rstatix::friedman_test()` and `friedman_effsize()` for canonical Friedman analysis with effect size; base R `friedman.test()` for the lightweight implementation; `coin::friedman_test()` for permutation-based exact p-values; `ARTool::art()` for aligned-rank transform ANOVA extending Friedman to factorial designs; `WRS2::rmmcp()` for robust repeated-measures comparisons under non-Normality.

4.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.16.11 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.17 Holm's Step-Down Correction

4.17.1 Introduction

Holm's (1979) step-down procedure controls the family-wise error rate (FWER) but is uniformly more powerful than Bonferroni. It tests the smallest p-value against α/m, and each subsequent p-value against a progressively larger threshold as the earlier hypotheses are rejected.

4.17.2 Prerequisites

Bonferroni correction, multiple testing.

4.17.3 Theory

Order the m p-values as $p_{(1)} \leq p_{(2)} \leq ... \leq p_{(m)}$. Define thresholds

$$\alpha_j = \alpha/(m - j + 1)$$

for $j = 1, ..., m$. Reject $H_{(1)}$ if $p_{(1)} \leq \alpha_1$; if so, reject $H_{(2)}$ if $p_{(2)} \leq \alpha_2$; continue sequentially. Stop at the first non-rejection; all subsequent hypotheses are also not rejected.

Equivalently, the Holm-adjusted p-value is

$$p_{(j)}^{\text{adj}} = \min\left[1, \max_{i \leq j}(m - i + 1)p_{(i)}\right].$$

Holm always rejects at least as many hypotheses as Bonferroni, strictly more when any rejection occurs. It controls FWER at α under any dependence structure.

4.17.4 Assumptions

Same as Bonferroni: no distributional assumptions on the p-values beyond validity.

4.17.5 R Implementation

```r
# Simulated p-values across 8 tests
p_raw <- c(0.001, 0.005, 0.012, 0.018, 0.025, 0.040, 0.080, 0.2)
m <- length(p_raw)

# Holm via p.adjust
p_holm <- p.adjust(p_raw, method = "holm")

# Bonferroni for comparison
p_bonf <- p.adjust(p_raw, method = "bonferroni")

data.frame(
  raw = p_raw,
  bonf = p_bonf,
  holm = p_holm,
  reject_at_0.05_holm = p_holm < 0.05,
  reject_at_0.05_bonf = p_bonf < 0.05
```

```
)
```

4.17.6 Output & Results

```
    raw  bonf   holm reject_at_0.05_holm reject_at_0.05_bonf
1 0.001 0.008 0.008                TRUE                TRUE
2 0.005 0.040 0.035                TRUE                TRUE
3 0.012 0.096 0.072               FALSE               FALSE
4 0.018 0.144 0.090               FALSE               FALSE
5 0.025 0.200 0.100               FALSE               FALSE
6 0.040 0.320 0.120               FALSE               FALSE
7 0.080 0.640 0.160               FALSE               FALSE
8 0.200 1.000 0.200               FALSE               FALSE
```

Both Holm and Bonferroni reject the same two hypotheses here. Holm's adjusted p-values are always $\leq$ Bonferroni's, opening space for additional rejections in other examples.

4.17.7 Interpretation

"After Holm step-down correction for multiple testing, two of the eight tests reached significance at family-wise $\alpha = 0.05$ (raw p = 0.001 and 0.005)."

4.17.8 Practical Tips

- Prefer Holm to Bonferroni by default; it is always at least as powerful and equally valid.
- Both procedures assume no prior ordering of hypotheses; if hypotheses are pre-ordered, fixed-sequence testing is more powerful.
- For very large m, FWER-controlling procedures are harsh; consider FDR methods (Benjamini-Hochberg) instead.
- Hochberg's step-up procedure is slightly more powerful than Holm but requires independence or positive dependence.
- Holm can be combined with post-hoc comparisons and with stratified tests without additional adjustment.

4.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.17.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.18 The Jackknife

4.18.1 Introduction

The jackknife is a resampling procedure that predates the bootstrap. It computes the statistic from each leave-one-out subsample and uses the variation across these subsamples to estimate the statistic's bias and variance. It is simpler than the bootstrap but more limited in applicability.

4.18.2 Prerequisites

Bias and variance of estimators, bootstrap.

4.18.3 Theory

For a sample $\mathbf{X} = (X_1, \dots, X_n)$, define the leave-one-out subsamples $\mathbf{X}_{(-i)}$ by dropping observation i. Let $\hat{\theta}_{(-i)} = T(\mathbf{X}_{(-i)})$ and define

$$\hat{\theta}_{(\cdot)} = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_{(-i)}.$$

Jackknife bias estimate:

$$\widehat{\text{bias}}_{\text{jack}} = (n-1)(\hat{\theta}_{(\cdot)} - \hat{\theta}).$$

Jackknife variance:

$$\widehat{\text{Var}}_{\text{jack}}(\hat{\theta}) = \frac{n-1}{n} \sum_{i=1}^{n} (\hat{\theta}_{(-i)} - \hat{\theta}_{(\cdot)})^2.$$

The jackknife works well for smooth statistics (means, correlations, linear-regression coefficients). It fails for non-smooth statistics like the median.

4.18.4 Assumptions

- iid observations.
- Statistic is sufficiently smooth (differentiable in the empirical distribution).

4.18.5 R Implementation

```r
library(bootstrap)
set.seed(2026)

x <- rnorm(40, mean = 5, sd = 2)

# Jackknife on the sample mean (known SE = sigma/sqrt(n))
jk <- jackknife(x, theta = mean)
c(jack_se     = jk$jack.se,
  formula_se  = sd(x) / sqrt(length(x)))

# Jackknife on Pearson correlation (a smooth statistic)
y <- 0.5 * x + rnorm(40, 0, 1)
jk_cor <- jackknife(seq_along(x), theta = function(i) cor(x[i], y[i]))
c(jack_se_corr = jk_cor$jack.se,
  bias         = jk_cor$jack.bias)

# Jackknife fails gracefully for the median
jk_med <- jackknife(x, theta = median)
jk_med$jack.se    # noisy, often biased
```

4.18.6 Output & Results

```
 jack_se  formula_se
  0.312       0.316

jack_se_corr        bias
     0.1088      0.00022
```

The jackknife SE for the mean matches the analytic formula. For the correlation, the jackknife gives SE and bias estimates directly.

4.18.7 Interpretation

"The Pearson correlation was 0.52 with jackknife SE 0.11 and bias 0.0002, suggesting a reliable point estimate with no concerning bias."

4.18.8 Practical Tips

- Use the jackknife for quick SE estimates on smooth statistics.

- For complicated or non-smooth statistics, prefer the bootstrap.
- Delete-d jackknife removes d observations at a time and can fix issues with non-smooth statistics.
- In regression, the jackknife is equivalent to leave-one-out cross-validation when the statistic is the prediction error.
- The jackknife's variance formula corrects for the $(n-1)/n$ effective-sample-size reduction – don't drop it.

4.18.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.18.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.19 Kendall's Tau

4.19.1 Introduction

Kendall's τ quantifies association via the proportion of pairs of observations that are **concordant** (both variables increase or both decrease) minus the proportion that are **discordant**. It is more robust than Spearman in small samples and under ties.

4.19.2 Prerequisites

Ranks, concordant / discordant pairs.

4.19.3 Theory

For n observations, there are $\binom{n}{2}$ pairs. A pair (i, j) is concordant if $\text{sign}(X_i - X_j) = \text{sign}(Y_i - Y_j)$, discordant if the signs differ, and tied otherwise.

$$\tau = \frac{\text{concordant} - \text{discordant}}{\binom{n}{2}}.$$

Variants (for ties):

- **tau-a**: simple difference divided by all pairs (including ties).
- **tau-b**: corrects for tied pairs in both variables; most commonly used with tied data.
- **tau-c** (Stuart's tau-c): for different-sized marginal distributions.

$\tau \in [-1, 1]$; 0 indicates no association. The null distribution (no association) is asymptotically normal; exact distributions exist for small n.

4.19.4 Assumptions

Same as Spearman: ordinal or continuous data, monotonic relationship, independent pairs.

4.19.5 R Implementation

```r
library(DescTools)
set.seed(2026)

n <- 40
x <- rnorm(n)
y <- 0.6 * x + rnorm(n, sd = sqrt(1 - 0.6^2))

cor.test(x, y, method = "kendall")

# Spearman for comparison
cor.test(x, y, method = "spearman")

# Tau-b (explicit)
KendallTauB(x, y, conf.level = 0.95)

# With ties
x2 <- round(x, 1)
y2 <- round(y, 1)
cor.test(x2, y2, method = "kendall")
```

4.19.6 Output & Results

```
        Kendall's rank correlation tau

data:  x and y
z = 3.68, p-value = 0.0002
```

```
alternative hypothesis: true tau is not equal to 0
tau: 0.394

        Spearman's rho: 0.57
```

Both correlations are significantly positive; Kendall's τ is numerically smaller than Spearman (typical, as they measure different things).

4.19.7 Interpretation

Report: "Kendall's $\tau = 0.39$ ($z = 3.68$, $p < 0.001$), indicating a moderate positive monotonic association between x and y."

4.19.8 Practical Tips

- Kendall and Spearman measure different quantities; do not treat them as interchangeable.
- Kendall is more robust in small samples; Spearman is more common in reporting.
- For heavily tied data (e.g., Likert scales), prefer Kendall tau-b over Spearman.
- `DescTools::KendallTauB()` provides the tau-b with a CI, which base R's `cor.test` does not.
- Kendall tau has a direct interpretation as (probability of concordance - probability of discordance), which Spearman does not.

4.19.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.19.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.20 Kolmogorov-Smirnov Test

4.20.1 Introduction

The Kolmogorov-Smirnov (KS) test compares an empirical distribution function (ECDF) against either a hypothesised theoretical CDF (one-sample test) or another empirical distribution (two-sample test). Its test statistic is the supremum of the absolute difference between the two CDFs, a measure of the largest vertical gap on a CDF plot. KS is the prototypical CDF-based goodness-of-fit test, valued for being fully non-parametric and distribution-free in the one-sample case (the null distribution of D does not depend on the specific F_0 for continuous distributions). It is widely used as a quick distributional diagnostic and as a flexible two-sample test sensitive to differences in location, scale, and shape.

4.20.2 Prerequisites

A working understanding of empirical CDFs, the Glivenko-Cantelli theorem, and the difference between testing a fully specified hypothesised distribution and testing a parametric family with parameters estimated from the data.

4.20.3 Theory

For the **one-sample** test with sample $X_1, \dots, X_n$ and hypothesised CDF F_0,

$$D = \sup_x |\hat{F}_n(x) - F_0(x)|;$$

under $H_0 : X \sim F_0$, $\sqrt{n} \cdot D$ has the Kolmogorov limiting distribution. For the **two-sample** test with empirical CDFs $\hat{F}_n$ and $\hat{G}_m$,

$$D_{n,m} = \sup_x |\hat{F}_n(x) - \hat{G}_m(x)|.$$

KS is distribution-free in the continuous one-sample case: the null distribution of D depends only on the sample size, not the specific F_0. When parameters of F_0 are estimated from the same data, the nominal p-values are anti-conservative; the Lilliefors correction provides the appropriate adjustment for testing Normality with estimated mean and SD.

4.20.4 Assumptions

The underlying distribution is continuous (ties produce warnings and approximate p-values), observations are independent and identically distributed, and parameters of F_0 are either fully specified or appropriately corrected for if estimated from the sample.

4.20.5 R Implementation

```r
set.seed(2026)

x <- rlnorm(100, meanlog = 0, sdlog = 0.8)
ks.test(x, "pnorm", mean = mean(x), sd = sd(x))

z <- scale(x)
ks.test(z, "pnorm")

x1 <- rnorm(50, 0, 1)
x2 <- rnorm(60, 0.4, 1.2)
ks.test(x1, x2)

library(nortest)
lillie.test(x)
```

4.20.6 Output & Results

The script demonstrates the four standard use cases. The one-sample KS against a Normal with sample-estimated parameters rejects the null but the p-value is anti-conservative; the Lilliefors-corrected variant gives the proper rejection. The two-sample KS detects the location and scale shift between X_1 and X_2 ($D = 0.31$, $p = 0.007$), illustrating the test's sensitivity to general distributional differences.

4.20.7 Interpretation

A reporting sentence: "The Lilliefors-corrected Kolmogorov-Smirnov test rejected Normality of the residuals from the regression model ($D = 0.16$, $p < 0.001$), confirming the visual impression of right-skew in the Q-Q plot. A log transformation was applied before re-fitting; the residuals on the transformed scale passed the same test ($D = 0.05$, $p = 0.21$). The two-sample KS comparing transformed-residual distributions across treatment arms detected no significant difference in shape ($D = 0.08$, $p = 0.52$)." Always pair KS with a Q-Q plot.

4.20.8 Practical Tips

- KS is generally less powerful than Shapiro-Wilk for testing Normality in small to moderate samples; prefer Shapiro-Wilk as the default Normality test and use KS when the hypothesised distribution is non-Normal or when a two-sample distributional comparison is needed.
- Ties in the sample invalidate the strict KS p-value and R issues a warning; prefer Anderson-Darling or the Cramér-von Mises test with discretised or rounded data, which handle ties more gracefully.

- The Lilliefors correction must be used when testing Normality with sample-estimated mean and SD; the standard `ks.test()` against `pnorm` with sample estimates gives anti-conservative p-values that reject too rarely.
- The two-sample KS detects any distributional difference — location, scale, shape — and is therefore a useful omnibus diagnostic when the analyst is unsure which aspect of the distribution might differ; it is generally less powerful than the Mann-Whitney U test for pure location shifts.
- For very large samples, KS will reject almost any parametric fit because real data have rounding, ceiling effects, and other minor non-Normalities; use effect-size summaries and Q-Q plots in large samples rather than relying on the p-value alone.
- The Anderson-Darling test gives more weight to the tails of the distribution than KS does; for tail-sensitive applications (extreme-value detection, financial risk), Anderson-Darling is often preferred.

4.20.9 R Packages Used

Base R `ks.test()` for one- and two-sample Kolmogorov-Smirnov; `nortest::lillie.test()` for the Lilliefors-corrected Normality variant; `nortest::ad.test()` and `nortest::cvm.test()` for Anderson-Darling and Cramér-von Mises alternatives; `dgof::ks.test()` for discrete-data extensions; `car::qqPlot()` for graphical complements to the formal test.

4.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.20.11 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.21 Kruskal-Wallis Test

4.21.1 Introduction

The Kruskal-Wallis test extends the Mann-Whitney U to three or more independent groups. It tests whether the distributions differ across groups, without assuming normality or equal variances. Post-hoc pairwise comparisons use Dunn's test with a multiple-testing correction.

4.21.2 Prerequisites

Mann-Whitney U, ranks.

4.21.3 Theory

Pool all observations and rank them across groups. Let R_i be the sum of ranks in group i with size n_i, and $N = \sum n_i$. The Kruskal-Wallis statistic is

$$H = \frac{12}{N(N+1)} \sum_i \frac{R_i^2}{n_i} - 3(N+1) \sim \chi^2_{k-1}.$$

Under H_0 of equal distributions, H has an approximate chi-squared distribution with $k-1$ df for moderate sample sizes.

Effect size: $\varepsilon^2 = H/(N-1)$ or $\eta^2_H = (H-k+1)/(N-k)$.

Post-hoc: Dunn's test compares each pair of groups using z-statistics with Bonferroni or Benjamini-Hochberg correction.

4.21.4 Assumptions

- Independent observations within and across groups.
- Observations at least ordinal.
- Similar distribution shapes (for a median-difference interpretation); otherwise tests distributional equality.

4.21.5 R Implementation

```
library(rstatix); library(dunn.test)
set.seed(2026)

df <- data.frame(
  group = factor(rep(c("A", "B", "C", "D"), each = 25)),
  y     = c(rnorm(25, 50, 8),
            rnorm(25, 55, 8),
            rnorm(25, 60, 12),
            rnorm(25, 52, 8))
```

```
)

kruskal.test(y ~ group, data = df)

# Effect size
df |> kruskal_effsize(y ~ group)

# Dunn's post-hoc with BH correction
dunn.test(df$y, df$group, method = "bh")
```

4.21.6 Output & Results

```
        Kruskal-Wallis rank sum test
Kruskal-Wallis chi-squared = 15.22, df = 3, p-value = 0.0016

Cohen's eta2 | 95% CI
------------------------
0.14          | [0.02, 1.00]

Dunn's test with Benjamini-Hochberg adjustment:
     Comparison    p.adj
     A - B         0.086
     A - C         0.003
     A - D         0.302
     B - C         0.136
     B - D         0.302
     C - D         0.015
```

4.21.7 Interpretation

"Group differed significantly across the four groups (Kruskal-Wallis $H(3) = 15.22$, $p = 0.002$, $\eta^2_H = 0.14$). Dunn post-hoc tests identified group C as significantly higher than groups A and D (BH-adjusted $p < 0.05$)."

4.21.8 Practical Tips

- Report medians and IQRs per group.
- Dunn's test is the standard post-hoc procedure; pairwise Wilcoxon with adjustment is an alternative.
- For within-subjects designs, use Friedman.
- When the only question is whether any groups differ, skipping the omnibus test and doing pairwise Wilcoxons with adjustment is statistically valid but loses interpretability.
- For heavy tails or strong shape differences, the test compares distributions more broadly than medians.

4.21.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.21.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.22 Levene's Test of Variances

4.22.1 Introduction

Levene's test checks whether variances are equal across two or more groups, providing the formal diagnostic for the homogeneity-of-variance assumption that underpins classical analysis of variance and the two-sample t-test. Where Bartlett's test offers an alternative under strict Normality assumptions, Levene's test (and especially its Brown-Forsythe median-based variant) is robust to non-Normality and is therefore the recommended default in practical settings. Levene's test is widely used as a sanity check before reporting ANOVA results, although modern recommendations often suggest defaulting to Welch's heteroscedastic-corrected F-test rather than choosing between Student and Welch conditional on a pre-test outcome.

4.22.2 Prerequisites

A working understanding of one-way ANOVA, the homogeneity-of-variance assumption, and the role of robustness diagnostics in choosing between classical and corrected inferential procedures.

4.22.3 Theory

For k groups, Levene's test replaces each observation by its absolute deviation from the group centre — the group mean (Levene's original 1960 form) or the

group median (Brown-Forsythe's 1974 modification) — and then computes a one-way ANOVA on the transformed values:

$$W = \frac{(N-k)\sum_i n_i(\bar{Z}_i - \bar{Z})^2}{(k-1)\sum_i \sum_j (Z_{ij} - \bar{Z}_i)^2},$$

where Z_{ij} is the absolute deviation. Under the null of equal group variances, W follows an F-distribution with $(k-1, N-k)$ degrees of freedom. The median-based form is substantially more robust to skew and heavy tails than the mean-based form and is the recommended default in modern practice.

4.22.4 Assumptions

Observations are independent within and across groups, the absolute-deviation transformation removes the dependence on the assumed distributional form, and the chosen centre (median for non-Normal data, mean for symmetric Normal data) is appropriate. Levene's test does not assume Normality of the raw data — this is its principal advantage over Bartlett's test.

4.22.5 R Implementation

```
library(car)
set.seed(2026)

df <- data.frame(
  group = factor(rep(c("A", "B", "C"), each = 40)),
  y     = c(rnorm(40, 50, 5),
            rnorm(40, 55, 12),
            rnorm(40, 52, 5))
)

leveneTest(y ~ group, data = df, center = median)

leveneTest(y ~ group, data = df, center = mean)

bartlett.test(y ~ group, data = df)
```

4.22.6 Output & Results

The example shows three groups with deliberately heterogeneous variances ($\sigma = 5, 12, 5$). Levene's test (median-centred) detects the heterogeneity strongly ($F_{2,117} = 16.98$, $p < 0.001$), as does the mean-centred form. Bartlett's test rejects even more emphatically ($\chi^2 = 35.2$, $p < 0.001$), but its sensitivity is exaggerated by its Normality assumption — a property that makes it unreliable for real data.

4.22.7 Interpretation

A reporting sentence: "Levene's test (median-centred) indicated significant heterogeneity of variance across the three groups ($F_{2,117} = 16.98$, $p < 0.001$); group B had standard deviation 12 vs 5 in groups A and C. Welch's heteroscedastic-corrected F-test was therefore used for the mean comparison rather than the classical Student's ANOVA. Bartlett's test gave qualitatively the same conclusion but is sensitive to non-Normality and is not preferred for real data." Always describe the variance pattern and the chosen response.

4.22.8 Practical Tips

- Use the median-based Brown-Forsythe variant of Levene's test by default; it is substantially more robust to non-Normality than the original mean-based form, and the cost in power against truly Normal data is small.
- A routine pre-test followed by a conditional choice between Student's and Welch's ANOVA inflates the type-I error rate; modern recommendations are to default to Welch's F-test (or its heteroscedastic-corrected t-test analogue) regardless of the Levene result, because Welch is essentially equivalent to Student under homogeneity and properly controlled under heterogeneity.
- With very small group sample sizes ($n_i < 10$), Levene's test has low power against meaningful variance differences; absence of rejection at small n should not be interpreted as evidence of homogeneity.
- Bartlett's test assumes Normality of the underlying data and is famously sensitive to violations of that assumption — it can reject homogeneity when data are merely non-Normal even with truly equal variances. Avoid Bartlett for real data and use Levene or Brown-Forsythe instead.
- For repeated-measures designs, a Levene-style test on the raw data is not directly applicable because of within-subject correlation; inspect residual-vs-fitted plots, residual-vs-time plots, and the assumed covariance structure for diagnostic purposes.
- Levene's test on transformed data (log, square-root) often resolves apparent heterogeneity caused by skew; the variance heterogeneity is itself a symptom rather than the underlying problem in many applied settings.

4.22.9 R Packages Used

`car::leveneTest()` for the canonical Levene and Brown-Forsythe implementations with explicit centre control; base R `bartlett.test()` for Bartlett's test under Normality assumption; `lawstat::levene.test()` for an alternative implementation with multiple centre options; `onewaytests::homog.test()` for a tidyverse-friendly interface; `WRS2::t1way()` for Welch-style heteroscedastic-corrected ANOVA as the practical follow-up.

4.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.22.11 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.23 Mann-Whitney U Test

4.23.1 Introduction

The Mann-Whitney U test (equivalent to the Wilcoxon rank-sum test) compares two independent groups using ranks rather than raw values. It is the non-parametric analogue of the two-sample t-test, and is appropriate for ordinal outcomes or continuous data that clearly violate normality.

4.23.2 Prerequisites

Ranks, two-sample comparison.

4.23.3 Theory

For independent samples $X_1, \dots, X_{n_1}$ and $Y_1, \dots, Y_{n_2}$, pool the observations, rank them, and sum the ranks in the first group to get R_1. The U statistic is

$$U_1 = R_1 - \frac{n_1(n_1+1)}{2}.$$

Under $H_0 : F_X = F_Y$, U_1 has a known distribution computed via the Wilcoxon rank sum. Large-sample normal approximation uses

$$E[U] = n_1 n_2/2, \quad \mathrm{Var}(U) = n_1 n_2 (n_1 + n_2 + 1)/12.$$

Effect sizes:

- **Rank-biserial correlation** $r_{rb} = 1 - 2U/(n_1 n_2)$, range $[-1, 1]$.
- **Probability of superiority** $A = U/(n_1 n_2)$.

4.23.4 Assumptions

- Two independent groups.
- Observations at least ordinal.
- Similar distribution shapes for a median-difference interpretation; otherwise the test compares distributions more generally.

4.23.5 R Implementation

```r
library(effectsize)
set.seed(2026)

group1 <- rlnorm(25, meanlog = log(50), sdlog = 0.4)
group2 <- rlnorm(30, meanlog = log(60), sdlog = 0.4)

wilcox.test(group1, group2)

# Effect size: rank-biserial
rank_biserial(group1, group2)

# Median IQR comparisons
c(median1 = median(group1), IQR1 = IQR(group1),
  median2 = median(group2), IQR2 = IQR(group2))
```

4.23.6 Output & Results

```
        Wilcoxon rank sum test with continuity correction
W = 240, p-value = 0.008

r (rank biserial) | 95% CI
---------------------------
-0.36             | [-0.59, -0.10]
```

Significant difference at p = 0.008; rank-biserial -0.36 indicates a moderate shift with group 1 tending to be smaller.

4.23.7 Interpretation

"Serum biomarker concentrations were significantly lower in the control group than in cases (Mann-Whitney $U = 240$, $p = 0.008$, rank-biserial $r = -0.36$). Median concentration was 47 (IQR 34-59) ng/mL vs. 62 (IQR 47-73) in cases."

4.23.8 Practical Tips

- Report medians and IQRs alongside the test; means and SDs are inappropriate when you chose Mann-Whitney.
- For small samples with no ties, use the exact distribution (`exact=TRUE`); with ties, R issues a warning and falls back to the normal approximation.
- The test is insensitive to outliers but sensitive to distributional shape differences (not just medians).
- For paired ordinal data, use the Wilcoxon signed-rank test.
- Power relative to the t-test is about 95 % under normality and can exceed it under heavy tails.

4.23.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.23.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.24 McNemar's Test

4.24.1 Introduction

When the same units are measured on a binary outcome twice (before/after, two tests, two raters), the observations are paired and a standard two-proportion test is invalid. McNemar's test analyses the 2x2 table of paired responses, using only the **discordant pairs** – those where the two measurements differ.

4.24.2 Prerequisites

Contingency tables, paired design.

4.24.3 Theory

Arrange paired binary data in a 2x2 table:

	Test 2: +	Test 2: -
Test 1: +	a (agree)	b (discordant)
Test 1: -	c (discordant)	d (agree)

McNemar's null: the probability of switching from + to - equals the probability of switching from - to +; equivalently, the marginal prevalences are equal.

Test statistic:

$$\chi^2_{\text{McNemar}} = \frac{(b-c)^2}{b+c} \sim \chi^2_1 \quad \text{under } H_0.$$

For small $b+c$, an exact binomial test on b out of $b+c$ with $p_0 = 0.5$ is preferred.

The concordant cells a and d do not enter the test; only the discordant b and c do.

4.24.4 Assumptions

- Paired binary data.
- Independent pairs.
- $b + c$ sufficiently large ($>= 25$) for the chi-squared approximation; otherwise exact.

4.24.5 R Implementation

```r
# Paired data: a diagnostic test evaluated before and after training
tab <- matrix(c(45, 12,
                25, 18), nrow = 2, byrow = TRUE,
              dimnames = list(Before = c("+", "-"),
                              After  = c("+", "-")))
tab

# McNemar (with continuity correction disabled)
mcnemar.test(tab, correct = FALSE)

# Exact McNemar via exact2x2
library(exact2x2)
mcnemar.exact(tab)

# Manual McNemar
```

```
b <- tab[1, 2]; c <- tab[2, 1]
chi <- (b - c)^2 / (b + c)
p_val <- 1 - pchisq(chi, df = 1)
c(chi = chi, p = p_val)
```

4.24.6 Output & Results

```
        McNemar's Chi-squared test
McNemar's chi-squared = 4.568, df = 1, p-value = 0.0326

Exact McNemar test (exact2x2):
proportion +1 to -1: 0.324
95% CI: 0.037 to 0.586
p-value = 0.041
```

Discordant pairs: 12 (before+ / after-) vs. 25 (before- / after+). McNemar $\chi^2(1) = 4.57$, p = 0.033; exact p = 0.041.

4.24.7 Interpretation

Reporting: "Of 37 discordant pairs, 25 switched from negative to positive and 12 switched in the opposite direction; McNemar's exact test gave p = 0.041, indicating a significant change after training."

4.24.8 Practical Tips

- Report only the discordant counts and the p-value; concordant counts are irrelevant to McNemar.
- For $b + c < 25$, use the exact version.
- McNemar's odds ratio is b/c; its CI is computed by inverting the exact test.
- For more than two categories per assessment (paired, more than two outcomes), use Bowker's symmetry test or Stuart-Maxwell.
- Stratified McNemar: use conditional logistic regression (`survival::clogit`) for a set of matched pairs with covariates.

4.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.24.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.25 Mixed ANOVA

4.25.1 Introduction

A mixed ANOVA combines at least one between-subjects factor (treatment arm, sex) with at least one within-subjects factor (time point, condition). It is the standard analysis for longitudinal trials and for split-plot designs. The typical primary question is the **group-by-time interaction**: does the change over time differ between groups?

4.25.2 Prerequisites

Repeated-measures ANOVA, factorial designs.

4.25.3 Theory

For a design with between factor A (a levels, independent groups) and within factor B (b levels, measured on each subject):

$$Y_{ijk} = \mu + \alpha_i + \pi_{k(i)} + \beta_j + (\alpha\beta)_{ij} + \varepsilon_{ijk},$$

where $\pi_{k(i)}$ is the subject-level random effect nested in A.

Three effects are tested:

- Main effect of A (between): uses between-subjects MS as denominator.
- Main effect of B (within): uses within-subjects MS as denominator.
- A x B interaction: uses within-subjects MS as denominator.

Sphericity applies to the within factor (and to its interaction with A); Greenhouse-Geisser or Huynh-Feldt corrections are used when needed.

4.25.4 Assumptions

- Independent subjects within each between-group.

- Normal residuals, separately for within and between components.
- Homogeneous variance-covariance structure across between groups (Box's M, rarely formally tested).
- Sphericity of the within factor.

4.25.5 R Implementation

```
library(afex); library(emmeans); library(ggplot2)
set.seed(2026)

# 30 subjects per arm, measured at 3 times
n <- 30
subj  <- rep(1:(2 * n), each = 3)
arm   <- factor(rep(rep(c("Placebo", "Active"), each = 3), n))
time  <- factor(rep(c("t0", "t1", "t2"), 2 * n))

df <- data.frame(
  subject = factor(subj),
  arm, time,
  y = rnorm(6 * n, mean = 100, sd = 10) +
      ifelse(time == "t1", 3, 0) +
      ifelse(time == "t2", 5, 0) +
      ifelse(arm == "Active" & time == "t2", -8, 0)
)

fit <- aov_car(y ~ arm * time + Error(subject/time), data = df)
fit

# Interaction plot
emm <- emmeans(fit, ~ arm * time)
plot(emm, comparisons = TRUE)

# Simple effects of arm at each time
pairs(emm, by = "time", adjust = "bonferroni")
```

4.25.6 Output & Results

```
Anova Table (Type 3 tests)

 Effect          df  MSE      F     ges     p.value
 arm           1, 58  245  8.12   0.09   0.0062 **
 time          2, 116  45   9.55  0.03  <.001 ***
 arm:time      2, 116  45   4.72  0.02   0.011 *
```

Significant main effects of arm and time, and a significant interaction. Simple-effects analysis localises the interaction.

4.25.7 Interpretation

"A 2 (arm) x 3 (time) mixed ANOVA revealed a significant interaction, F(2, 116) = 4.72, p = 0.011, $\eta^2_G = 0.02$. Simple-effects comparisons showed that at time 2, the Active arm was 8.0 points lower than Placebo (p = 0.003)."

4.25.8 Practical Tips

- For longitudinal trials, the interaction is typically the primary endpoint.
- Sphericity corrections apply only to the within-subjects parts of the model.
- Missing measurements on a within-subject factor cause listwise deletion of that subject; prefer a linear mixed model for unbalanced data.
- Report the interaction plot; it is often more interpretable than the numeric ANOVA table.
- Simple main effects (one factor fixed at a level of the other) give the most interpretable decomposition when the interaction is significant.

4.25.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.25.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.26 Multiple Comparisons: Overview

4.26.1 Introduction

Testing many hypotheses simultaneously inflates the chance of spurious discoveries. Controlling for this inflation is the goal of multiple-comparison corrections. Two broad families exist: **family-wise error rate (FWER)** methods and **false discovery rate (FDR)** methods. Choosing between them depends on the consequences of a false positive.

4.26.2 Prerequisites

Hypothesis testing, Type I error.

4.26.3 Theory

Suppose m hypotheses are tested and V of them are falsely rejected (true nulls). Let R be total rejections.

- **FWER** $= P(V \geq 1)$. The probability of any false rejection.
- **FDR** $= E[V/\max(R, 1)]$. Expected proportion of false positives among rejections.

Bonferroni, Holm, and Hochberg control FWER. Benjamini-Hochberg and Benjamini-Yekutieli control FDR.

When to use which:

- FWER: small number of pre-specified comparisons, confirmatory studies, clinical trial endpoints.
- FDR: exploratory or high-throughput studies with thousands of tests (genomics, imaging), where a few percent false discoveries are acceptable.

4.26.4 Assumptions

Methods assume valid individual p-values (uniform under null). FWER corrections are robust to arbitrary dependence; FDR methods vary (BH needs positive dependence, BY handles arbitrary).

4.26.5 R Implementation

```
set.seed(2026)
m <- 100
pvals <- runif(m) * ifelse(seq_len(m) <= 10, 0.01, 1)   # 10 small, 90
 ↪  null

adj_methods <- c("bonferroni", "holm", "hochberg", "BH", "BY")
adj <- sapply(adj_methods, function(m_name)
  p.adjust(pvals, method = m_name))

data.frame(method = adj_methods,
           rejections_at_0.05 = colSums(adj < 0.05))
```

4.26.6 Output & Results

```
      method rejections_at_0.05
1 bonferroni                 8
2        holm                 8
```

```
3    hochberg                 9
4          BH                10
5          BY                 7
```

FDR (BH) recovers the full signal set; FWER methods are more conservative. BY is tighter than BH because of its dependence-robust adjustment.

4.26.7 Interpretation

In a paper: "Multiple testing across 100 primary comparisons was controlled at FWER 0.05 using the Holm procedure (8 of 100 hypotheses rejected)." Or: "Across 50 000 transcripts, differential expression was declared at Benjamini-Hochberg $q < 0.05$ (2 341 transcripts)."

4.26.8 Practical Tips

- Correct for **all** tests that belong to the same family; a family is defined by the inferential goal, not by software runs.
- Pre-specify the family and the method in the protocol; post-hoc changes to the correction are data-snooping.
- For hierarchically structured hypotheses, gatekeeping procedures maintain FWER at each level.
- Report both raw and adjusted p-values for transparency.
- False discovery proportion (FDP) is the realisation of FDR on a given dataset; FDR is the expectation.

4.26.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.26.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.27 Null and Alternative Hypotheses

4.27.1 Introduction

Every hypothesis test begins with two hypotheses: a **null** that captures the status quo and an **alternative** that captures what we would conclude if the null is rejected. Stating both clearly, before the data arrive, is the crucial first step – and a remarkably common source of error when skipped.

4.27.2 Prerequisites

Random variables and population parameters.

4.27.3 Theory

The **null hypothesis** H_0 is typically a statement of no effect, no difference, or a specific parameter value. The **alternative** H_1 (or H_A) is what we would conclude if we reject H_0.

Two-sided alternatives are the default when the sign of the effect is unknown or of interest in both directions:

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0.$$

One-sided alternatives specify a direction, increasing power for that direction at the cost of giving up inference in the opposite:

$$H_0 : \mu \leq \mu_0 \quad \text{vs.} \quad H_1 : \mu > \mu_0.$$

Use a one-sided test only when:

- A priori, only one direction is of scientific interest.
- A result in the other direction would be treated identically to a null result.
- The directionality is pre-specified in the protocol, not chosen after seeing the data.

Simple vs. composite: $H_0 : \mu = 0$ is simple (a single value); $H_0 : \mu \leq 0$ is composite (a set). Test statistics and rejection rules differ slightly.

Equivalence/non-inferiority testing flips the roles: the null is that the effect is at least as large as some margin, and rejection establishes equivalence.

4.27.4 Assumptions

Hypotheses must be specified before examining the data. Post-hoc redefinition invalidates the sampling distribution of the p-value.

4.27.5 R Implementation

```r
set.seed(2026)

# Two-sided test
x <- rnorm(40, mean = 2, sd = 5)
t.test(x, mu = 0, alternative = "two.sided")

# One-sided test (H1: mu > 0)
t.test(x, mu = 0, alternative = "greater")

# One-sided test (H1: mu < 0)
t.test(x, mu = 0, alternative = "less")

# TOST (equivalence: |mu| < 1)
library(TOSTER)
tsum_TOST(m1 = mean(x), sd1 = sd(x), n1 = length(x),
          low_eqbound = -1, high_eqbound = 1)
```

4.27.6 Output & Results

The two-sided p will be roughly twice the one-sided p when the point estimate is consistent with H_1's direction. For equivalence testing, rejection of both one-sided components establishes equivalence.

4.27.7 Interpretation

Pre-register or pre-specify the hypotheses in the protocol; do not switch from two-sided to one-sided after seeing the sign of the effect. Misuse of one-sided tests is a common source of reviewer objections.

4.27.8 Practical Tips

- Default to two-sided tests unless a strong, pre-specified rationale for one-sidedness exists.
- Never compute both one-sided and two-sided p-values and report whichever is smaller.
- In equivalence trials, the "null" is dissimilarity and "alternative" is equivalence – reversed from classical tests.
- Composite null hypotheses are handled by evaluating the test statistic at the boundary.
- Frame hypotheses in terms of the parameter of scientific interest, not the test statistic.

4.27.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.27.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.28 One-Proportion Test

4.28.1 Introduction

The one-proportion test compares an observed success proportion $\hat{p}$ to a benchmark p_0. Applications: a phase II response rate compared to a historical control; a screening positive rate compared to a national reference; the success rate of a surgical procedure compared to a target.

4.28.2 Prerequisites

Binomial distribution, confidence intervals.

4.28.3 Theory

For $X \sim \text{Binomial}(n, p)$ with observed $\hat{p} = x/n$, three tests coexist:

Exact binomial test: computes the exact two-sided p-value under $H_0 : p = p_0$ using the binomial PMF. This is R's `binom.test()`. Always valid; no approximation.

Normal-approximation (Wald) z-test:

$$Z = \frac{\hat{p} - p_0}{\sqrt{p_0(1-p_0)/n}} \sim \mathcal{N}(0,1).$$

Works when $np_0 \geq 10$ and $n(1-p_0) \geq 10$; biased and unreliable near $p = 0$ or $p = 1$.

Score (Wilson) test: uses the variance under H_0 and inverts to get a CI with better coverage than Wald, especially for small n or extreme p. R's `prop.test()` uses this by default.

Confidence intervals:

- Wald: $\hat{p} \pm 1.96\sqrt{\hat{p}(1-\hat{p})/n}$ (poor near 0 or 1).
- Wilson: centred, always inside $[0, 1]$, good coverage.
- Clopper-Pearson (exact): always valid, conservative.

4.28.4 Assumptions

- Independent Bernoulli trials.
- Fixed n (not a sequential stopping rule).

4.28.5 R Implementation

```r
library(binom)

x <- 32; n <- 80; p0 <- 0.30

# Exact binomial test
binom.test(x, n, p = p0)

# Score (Wilson) test
prop.test(x, n, p = p0, correct = FALSE)

# Normal-approximation test (manual)
phat <- x / n
z <- (phat - p0) / sqrt(p0 * (1 - p0) / n)
pval <- 2 * (1 - pnorm(abs(z)))
c(z = z, p = pval)

# Confidence intervals
binom.confint(x, n, conf.level = 0.95,
              methods = c("wald", "wilson", "exact"))
```

4.28.6 Output & Results

```
        Exact binomial test
x = 32, n = 80, p-value = 0.04215, 95% CI: (0.292, 0.517)

     1-sample test for given proportion without continuity correction
```

```
X-squared = 3.9286, df = 1, p-value = 0.04748, 95% CI (score): (0.303,
 ↪  0.508)

     method  x  n    mean    lower   upper
1      wald 32 80   0.400  0.2926  0.5074
2    wilson 32 80   0.400  0.3020  0.5066
3     exact 32 80   0.400  0.2923  0.5172
```

Observed proportion 0.40 (95 % Wilson CI 0.30 to 0.51); exact binomial p = 0.042.

4.28.7 Interpretation

Report both the point estimate and the CI. In a manuscript: "the response rate was 40.0 % (32/80; 95 % Wilson CI 30.2 % to 50.7 %; exact binomial p = 0.042 vs. the null rate of 30 %)."

4.28.8 Practical Tips

- Use Wilson or Clopper-Pearson CIs by default; Wald is obsolete.
- For small n and extreme p (near 0 or 1), exact tests and intervals are mandatory.
- Do not report zero/one proportions as 0 and 1 with a Wald CI of "0 to 0"; use a Clopper-Pearson upper bound (rule of three: upper bound $\approx 3/n$ if $x = 0$).
- Continuity correction in `prop.test()` is unnecessary for modern purposes; set `correct=FALSE`.
- For agreement with two-proportion tests, use the same CI method (Wilson) in both.

4.28.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.28.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests

- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.29 One-Sample t-Test

4.29.1 Introduction

The one-sample t-test compares the mean of a single continuous variable to a pre-specified reference value. It answers questions of the form "does the average differ from some benchmark?" – average IQ differs from 100, mean biomarker level differs from a reference, average weight change differs from zero in a pre-post design (via paired differences).

4.29.2 Prerequisites

t-distribution, the concept of a sampling distribution.

4.29.3 Theory

Given iid observations $X_1, \ldots, X_n$ from $\mathcal{N}(\mu, \sigma^2)$, the test of

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$

uses the statistic

$$T = \frac{\bar{X} - \mu_0}{s/\sqrt{n}} \sim t_{n-1} \quad \text{under } H_0.$$

The sample SD s replaces the population σ; the distribution adjusts from normal to t because this replacement adds variability. The degrees of freedom are $n-1$.

Confidence interval for μ: $\bar{X} \pm t_{1-\alpha/2,n-1} \cdot s/\sqrt{n}$.

Effect size: Cohen's $d = (\bar{X} - \mu_0)/s$.

4.29.4 Assumptions

- Independent observations.
- Normal distribution (or n large enough for CLT to kick in).
- No extreme outliers affecting the mean.

4.29.5 R Implementation

```r
library(effectsize); library(broom)
set.seed(2026)

# Simulated IQ-test scores; is the mean different from 100?
iq <- rnorm(45, mean = 103.5, sd = 14)

t.test(iq, mu = 100) |> tidy()

# Effect size
cohens_d(iq, mu = 100)

# Check assumptions
shapiro.test(iq)
boxplot(iq, main = "Sample distribution", col = "#2A9D8F")
```

4.29.6 Output & Results

```
# A tibble: 1 x 8
  estimate statistic p.value parameter conf.low conf.high method
   ↪  alternative
     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl> <chr>   <chr>
1    104.      1.82   0.076      44    98.9    109.      One ... two.sided

Cohen's d |         95% CI
--------------------------------
0.27      | [-0.03, 0.57]
```

The sample mean is 104, not significantly different from 100 at $\alpha = 0.05$ ($t(44) = 1.82$, $p = 0.076$), with a small-to-medium effect ($d = 0.27$).

4.29.7 Interpretation

In a manuscript: "The mean IQ was 104.0 (SD 14.0). The 95 % CI for the mean (98.9 to 109.3) includes the reference value of 100; a one-sample t-test did not reject the null of equal mean (t(44) = 1.82, p = 0.076, Cohen's d = 0.27)."

4.29.8 Practical Tips

- For $n \geq 30$, the CLT makes the t-test fairly robust to non-normality.
- For small n with clear non-normality, use the Wilcoxon signed-rank test (one-sample version).
- Power for the one-sample t-test depends on effect size and n; use `power.t.test(type="one.sample")`.
- The reference value μ_0 must be pre-specified; post-hoc selection invalidates the p-value.

- Pair with a 95 % CI for the mean – it is more informative than the p-value alone.

4.29.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.29.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.30 One-Sample z-Test

4.30.1 Introduction

The one-sample z-test is the idealised version of the one-sample t-test: it assumes the population variance σ^2 is known exactly. This is rarely the case in applied research – if σ is unknown, we estimate it from the sample and use the t-test instead. The z-test remains pedagogically important because it is the simplest exact hypothesis test and the reference for the t-test's more elaborate machinery.

4.30.2 Prerequisites

Normal distribution, sampling distribution of the mean.

4.30.3 Theory

For $X_1, \dots, X_n$ iid from $\mathcal{N}(\mu, \sigma^2)$ with **known** σ, the test of

$$H_0 : \mu = \mu_0 \quad \text{vs.} \quad H_1 : \mu \neq \mu_0$$

uses

$$Z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}} \sim \mathcal{N}(0,1) \quad \text{under } H_0.$$

The two-sided p-value is $2(1 - \Phi(|z_{\text{obs}}|))$; 95 % CI for μ is $\bar{X} \pm 1.96\sigma/\sqrt{n}$.

The z-test is also used when σ is *well-estimated* – historically meaning $n > 30$ where the t and z distributions are nearly indistinguishable. Modern practice uses the t-test even then.

4.30.4 Assumptions

- Independent observations.
- Normal population (or n large enough for CLT).
- Population variance σ^2 known.

4.30.5 R Implementation

```r
set.seed(2026)

# Assume the assay has a known SD from reference literature
sigma_known <- 8
x <- rnorm(50, mean = 103, sd = sigma_known)

# Manual z-test
z_stat <- (mean(x) - 100) / (sigma_known / sqrt(length(x)))
p_two_sided <- 2 * (1 - pnorm(abs(z_stat)))
c(z = z_stat, p = p_two_sided)

# 95% CI
mean(x) + c(-1, 1) * qnorm(0.975) * sigma_known / sqrt(length(x))

# Via BSDA::z.test
library(BSDA)
z.test(x, mu = 100, sigma.x = sigma_known)
```

4.30.6 Output & Results

```
       z         p
2.1856   0.02884

[1] 100.53 105.47

        One-sample z-Test

data:  x
```

```
z = 2.1856, p-value = 0.02884
alternative hypothesis: true mean is not equal to 100
```

The sample mean is 103 with SE 1.13 (known SD / sqrt(n)); z = 2.19, p = 0.029, rejecting H_0 at $\alpha = 0.05$.

4.30.7 Interpretation

Reporting a z-test is rarely appropriate in modern applied work unless the variance is genuinely known (e.g., from decades of reference data). Reporting the t-test instead gives an honest representation of the variance uncertainty.

4.30.8 Practical Tips

- Default to the t-test; use z only when σ is genuinely known.
- For two-proportion comparisons, the "z-test for proportions" is a close cousin that uses an estimated variance.
- When $n > 30$ and σ is unknown, the t and z tests give essentially identical p-values.
- The z-test is the foundation for more elaborate procedures: power calculations, sample-size formulas, and the large-sample limit of many Wald-style tests.
- In process-control charts (SPC), the "3-sigma" rule is a z-test at $\alpha \approx 0.0027$.

4.30.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.30.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.31 One-Way ANOVA

4.31.1 Introduction

One-way ANOVA tests whether the means of three or more independent groups are equal. It generalises the two-sample t-test to multiple groups while controlling overall Type I error. The omnibus F-test tells us whether *any* means differ; post-hoc tests (Tukey HSD, Dunnett) identify which pairs.

4.31.2 Prerequisites

Two-sample t-test, F distribution.

4.31.3 Theory

For k groups with n_i observations each, total $n = \sum n_i$, define:

- **Sum of squares between** (SSB): $\sum_i n_i(\bar{X}_i - \bar{X})^2$ with $k-1$ df.
- **Sum of squares within** (SSW): $\sum_i \sum_j (X_{ij} - \bar{X}_i)^2$ with $n-k$ df.
- **Mean squares**: MSB = SSB / (k - 1), MSW = SSW / (n - k).

Under $H_0 : \mu_1 = ... = \mu_k$,

$$F = \frac{\text{MSB}}{\text{MSW}} \sim F_{k-1,n-k}.$$

Large F implies between-group variation exceeds within-group variation, rejecting equality of means.

Effect size: $\eta^2 = \text{SSB/SST}$, the proportion of variance explained by the group factor.

4.31.4 Assumptions

- Independent observations within and across groups.
- Normal residuals (or large enough n_i).
- Homogeneous variances across groups (Welch ANOVA relaxes this).

4.31.5 R Implementation

```
library(effectsize); library(broom); library(car)
set.seed(2026)

df <- data.frame(
  group = factor(rep(c("A", "B", "C", "D"), each = 30)),
  y     = c(rnorm(30, 50, 8),
            rnorm(30, 55, 8),
```

```
            rnorm(30, 58, 8),
            rnorm(30, 54, 8))
)

# Omnibus ANOVA
fit <- aov(y ~ group, data = df)
summary(fit)
tidy(fit)

# Assumption checks
leveneTest(y ~ group, data = df)
shapiro.test(residuals(fit))

# Effect size
eta_squared(fit)
omega_squared(fit)

# Welch ANOVA (unequal variances)
oneway.test(y ~ group, data = df)

# Tukey HSD post-hoc
TukeyHSD(fit)
```

4.31.6 Output & Results

```
            Df Sum Sq Mean Sq F value  Pr(>F)
group        3    891   297.0    4.83 0.00317 **
Residuals  116   7135    61.5

Eta2 (partial) | 95% CI
--------------------------
0.11           | [0.02, 1.00]

        Tukey multiple comparisons of means 95% family-wise confidence
        ↪ level

         diff    lwr    upr  p adj
B-A   5.421   -0.58  11.42 0.097
C-A   8.215    2.21  14.22 0.003
D-A   4.129   -1.87  10.13 0.287
C-B   2.794   -3.21   8.79 0.627
D-B  -1.292   -7.29   4.71 0.943
D-C  -4.086  -10.09   1.92 0.297
```

$F(3, 116) = 4.83$, $p = 0.003$, rejects equal means. Tukey identifies C vs. A as the significant pair.

4.31.7 Interpretation

Report: "A one-way ANOVA showed a significant group effect ($F(3, 116) = 4.83$, $p = 0.003$, $\eta^2 = 0.11$). Tukey HSD comparisons indicated that group C differed significantly from group A (mean difference 8.2, 95 % CI 2.2 to 14.2, adjusted $p = 0.003$)."

4.31.8 Practical Tips

- Always check homogeneity of variances; if violated, use Welch ANOVA.
- Always report an effect size; F alone is not enough.
- For many post-hoc comparisons, Tukey is the standard; Dunnett is for comparing treatments to a control; Bonferroni or Holm for pre-specified families.
- Non-parametric alternative: Kruskal-Wallis.
- For within-subjects data, use repeated-measures ANOVA or a linear mixed model.

4.31.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.31.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.32 P-Values Explained

4.32.1 Introduction

The p-value is the single most reported statistic in applied research, and the most frequently misinterpreted. Understanding what a p-value is – and what it is not – is essential for reading, writing, and reviewing quantitative studies.

The American Statistical Association's 2016 statement on p-values laid out six principles that every user of statistics should internalise.

4.32.2 Prerequisites

Basic hypothesis testing and sampling distributions.

4.32.3 Theory

A **p-value** is the probability, assuming the null hypothesis is true, of observing data at least as extreme as what was actually observed. Formally,

$$p = P(T \geq t_{\text{obs}} \mid H_0)$$

for a one-sided test with test statistic T, or a two-sided analogue for two-sided alternatives.

Correct interpretation: a small p-value means the observed data would be unusual under H_0. It does **not** mean that H_0 is unlikely to be true, nor that the alternative is probable.

Common misinterpretations:

1. "$p = 0.03$ means there's a 3% probability the null is true." Wrong: the p-value is a conditional probability of data given H_0, not of H_0 given data.
2. "$p < 0.05$ means the result is meaningful." Wrong: small p-values can come from huge samples detecting trivial effects.
3. "$p > 0.05$ means no effect." Wrong: failure to reject is not evidence of no effect; it is absence of evidence.
4. "p-values from the same experiment are comparable." Wrong: p-values are discontinuous in the effect size; a p of 0.04 and 0.06 are nearly identical evidentially.

Under H_0, the p-value has a uniform distribution on $[0, 1]$ (for continuous test statistics). This is a useful simulation check.

4.32.4 Assumptions

- The null hypothesis and test statistic are pre-specified.
- The sampling model under H_0 is correctly specified.
- Multiple testing, optional stopping, and selective reporting are not present.

4.32.5 R Implementation

```
library(ggplot2)
set.seed(2026)
```

```r
# Simulate p-values under H0 (they should be uniform)
n_sims <- 5000
pvals_null <- replicate(n_sims, {
  x <- rnorm(30, mean = 0, sd = 1)   # H0 true: mu = 0
  t.test(x, mu = 0)$p.value
})

hist(pvals_null, breaks = 40, col = "#2A9D8F",
     main = "p-values under H0 are uniform", xlab = "p-value")

# Simulate under H1
pvals_alt <- replicate(n_sims, {
  x <- rnorm(30, mean = 0.5, sd = 1)
  t.test(x, mu = 0)$p.value
})

hist(pvals_alt, breaks = 40, col = "#F4A261",
     main = "p-values under H1 pile near zero", xlab = "p-value")

# Empirical power at alpha = 0.05
mean(pvals_alt < 0.05)
power.t.test(n = 30, delta = 0.5, sd = 1)$power
```

4.32.6 Output & Results

```
[1] 0.763    # empirical power
[1] 0.752    # theoretical power
```

Under H_0, the histogram is flat. Under H_1, p-values pile near zero. The empirical rejection rate at $\alpha = 0.05$ matches the theoretical power.

4.32.7 Interpretation

When reporting, pair p-values with effect sizes and confidence intervals. "The difference was 3.2 mmHg (95 % CI 0.6 to 5.8 mmHg, $p = 0.015$)" conveys both the magnitude and the statistical significance.

4.32.8 Practical Tips

- Never report a p-value alone; always include the effect size and its CI.
- Pre-specify one-sided vs. two-sided tests; switching after seeing data is fishing.
- Large samples make tiny effects statistically significant; focus on practical significance when n is large.
- Multiple testing inflates false-positive rates; adjust with Bonferroni, Holm, or FDR.

- "Nearly significant" ($0.05 < p < 0.10$) is not a meaningful category; report the exact value.

4.32.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.32.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.33 Paired t-Test

4.33.1 Introduction

When each unit contributes two measurements (pre and post, left and right, case and matched control), the observations are paired and the usual independent-samples t-test is not valid. The **paired t-test** analyses the within-unit differences, restoring the independence assumption and typically improving power by cancelling between-unit variability.

4.33.2 Prerequisites

One-sample t-test, t-distribution.

4.33.3 Theory

Let X_i and Y_i be paired measurements on unit i for $i = 1, \dots, n$. Define differences $D_i = X_i - Y_i$. Under the assumption that D_i are iid from $\mathcal{N}(\mu_D, \sigma_D^2)$,

$$T = \frac{\bar{D}}{s_D/\sqrt{n}} \sim t_{n-1} \quad \text{under } H_0 : \mu_D = 0.$$

The paired t-test is the one-sample t-test on the differences. The n is the number of **pairs**, not the total number of measurements.

Effect size: Cohen's $d_z = \bar{D}/s_D$ (standardised mean difference on the difference scores). This is different from the two-group d and should not be conflated.

4.33.4 Assumptions

- The differences are approximately normal (via Shapiro-Wilk on D_i).
- Pairs are independent of other pairs.

The raw X_i and Y_i can be non-normal; only the differences need to be.

4.33.5 R Implementation

```
library(effectsize); library(broom)
set.seed(2026)

# Simulated pre-post blood pressure after a 12-week intervention
n <- 30
pre  <- rnorm(n, mean = 138, sd = 12)
post <- pre + rnorm(n, mean = -6, sd = 5)

# Paired t-test
t.test(pre, post, paired = TRUE) |> tidy()

# Equivalent: one-sample t-test on the differences
t.test(pre - post, mu = 0) |> tidy()

# Effect size
cohens_d(pre, post, paired = TRUE)

# Assumption check on differences
shapiro.test(pre - post)
```

4.33.6 Output & Results

```
# A tibble: 1 x 8
  estimate statistic p.value parameter conf.low conf.high method
   ↪  alternative
     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl> <chr>       <chr>
1     6.04      6.88 <0.001        29     4.25      7.84 Paired t-...
 ↪  two.sided

Cohen's d (paired) |         95% CI
-------------------------------------
1.26               | [0.78, 1.74]
```

The post-intervention measurement is 6.0 mmHg lower on average; $t(29) = 6.88$, $p < 0.001$, large paired effect.

4.33.7 Interpretation

In a manuscript: "Blood pressure decreased by 6.0 mmHg on average from baseline to week 12 (paired t(29) = 6.88, p < 0.001, Cohen's d_z = 1.26, 95 % CI 4.3 to 7.8 mmHg)."

4.33.8 Practical Tips

- Paired design usually has more power than independent-groups design at the same total sample size.
- Check normality of the differences, not the raw measurements.
- For non-normal differences, use Wilcoxon signed-rank.
- Pair variables correctly: subjects must be in the same order in both vectors, or use a data frame.
- Report Cohen's d_z, not pooled-SD two-group d, for paired designs.

4.33.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.33.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.34 Pearson Correlation Test

4.34.1 Introduction

The Pearson correlation test assesses whether two continuous variables are linearly related. It returns a correlation coefficient r and a p-value for the null that

the population correlation $\rho = 0$, along with a confidence interval computed via Fisher's z transformation.

4.34.2 Prerequisites

Pearson correlation coefficient, sampling distributions.

4.34.3 Theory

Given iid pairs (X_i, Y_i) from a bivariate distribution with marginal SDs σ_X, σ_Y, the sample correlation is

$$r = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}}.$$

Under $H_0 : \rho = 0$, the statistic

$$t = r\sqrt{\frac{n-2}{1-r^2}} \sim t_{n-2}.$$

For confidence intervals, Fisher's z-transformation is

$$z = \frac{1}{2}\log\frac{1+r}{1-r}, \quad \mathrm{SE}(z) = \frac{1}{\sqrt{n-3}}.$$

Construct a CI in the z scale, then back-transform with $r = \tanh(z)$.

4.34.4 Assumptions

- Continuous X and Y, approximately bivariate normal.
- Linear relationship; no extreme outliers.
- Independent pairs.

4.34.5 R Implementation

```r
library(broom)
set.seed(2026)

n <- 80
x <- rnorm(n)
y <- 0.4 * x + rnorm(n, sd = sqrt(1 - 0.4^2))  # true rho = 0.4

cor.test(x, y, method = "pearson") |> tidy()
```

```r
# Manual check
r <- cor(x, y)
t_stat <- r * sqrt((n - 2) / (1 - r^2))
p_val  <- 2 * (1 - pt(abs(t_stat), df = n - 2))
c(r = r, t = t_stat, p = p_val)

# Fisher's z CI
z    <- atanh(r)
se_z <- 1 / sqrt(n - 3)
ci_z <- z + c(-1, 1) * 1.96 * se_z
tanh(ci_z)
```

4.34.6 Output & Results

```
# A tibble: 1 x 8
  estimate statistic p.value parameter conf.low conf.high method
   ↪  alternative
     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl> <chr>     <chr>
1    0.43     4.19   <0.001        78   0.234     0.598  Pearson   two.sided
```

Sample $r = 0.43$ (95 % CI 0.23 to 0.60), $t(78) = 4.19$, $p < 0.001$. Strong evidence of a positive correlation.

4.34.7 Interpretation

"There was a significant positive correlation between x and y ($r = 0.43$, 95 % CI 0.23 to 0.60; $t(78) = 4.19$, $p < 0.001$). The two variables shared 18.5 % of their variance linearly."

4.34.8 Practical Tips

- Scatter plot first; r can be misleading if the relationship is non-linear.
- Robust alternatives for heavy tails or outliers: Spearman, Kendall.
- For small n, the CI via Fisher's z is noticeably asymmetric in the r scale.
- Always report both r and its CI; the p-value alone is often uninformative.
- Testing multiple correlations from the same data requires multiple-testing correction.

4.34.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.34.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.35 Permutation Tests

4.35.1 Introduction

A permutation test evaluates a null hypothesis by comparing the observed test statistic to its distribution under random permutations of the group labels (or other exchangeable structure). It makes no distributional assumption about the data, only about **exchangeability** under the null.

4.35.2 Prerequisites

Hypothesis testing, ranks (helpful).

4.35.3 Theory

Under H_0, group labels are exchangeable – the joint distribution of $(X_1, \dots, X_n)$ is invariant under permutations of the labels. The p-value is the proportion of permuted test statistics at least as extreme as the observed one:

$$p = \frac{1 + \#\{T^{(b)} \geq T_{\text{obs}}\}}{B + 1},$$

where $T^{(b)}$ is the statistic from the b-th permutation out of B random permutations, plus the observed. Exact permutation tests use all $n!$ permutations; for large n, Monte Carlo permutation with $B = 1000$ or more suffices.

Permutation tests can use **any test statistic** – difference of means, t-statistic, ratio of medians, area under an ROC. The choice of statistic determines what aspect of the distribution the test is sensitive to.

4.35.4 Assumptions

- Exchangeability under H_0 (independent observations with equal distribution for group-based tests).

- Independence across observations.

No normality, no variance homogeneity, no distributional model.

4.35.5 R Implementation

```
library(coin); library(perm)
set.seed(2026)

group1 <- c(rlnorm(20, 1, 0.4), 30)   # one extreme value
group2 <- rlnorm(25, 1.2, 0.4)

# Manual permutation test on difference of means
obs_diff <- mean(group1) - mean(group2)
pooled   <- c(group1, group2)
n1       <- length(group1)
B        <- 10000

perm_diffs <- replicate(B, {
  shuffled <- sample(pooled)
  mean(shuffled[1:n1]) - mean(shuffled[(n1+1):length(shuffled)])
})
p_perm <- (1 + sum(abs(perm_diffs) >= abs(obs_diff))) / (B + 1)
c(obs_diff = obs_diff, p_perm = p_perm)

# coin package: permutation version of many standard tests
oneway_test(y ~ group,
            data = data.frame(y = c(group1, group2),
                              group = factor(rep(c("A", "B"), c(n1,
                               ↪  length(group2))))),
            distribution = approximate(nresample = 10000))
```

4.35.6 Output & Results

```
obs_diff    p_perm
  0.841     0.123

Approximative Two-Sample Fisher-Pitman Permutation Test
Z = 1.56, p = 0.13
```

Permutation test gives p = 0.12, robust to the outlier that would distort a t-test.

4.35.7 Interpretation

"A Monte Carlo permutation test (10 000 resamples) comparing the two groups' mean differences gave p = 0.12, providing no evidence of a group effect."

4.35.8 Practical Tips

- Permutation tests are exact in small samples where combinatorial enumeration is feasible.
- Monte Carlo permutation uses random resampling; $B = 1000$-$10\,000$ gives stable p-values.
- The statistic can be any function of the data; this flexibility is the main appeal over parametric tests.
- For paired data, permute within pairs (sign flips) rather than labels.
- For regression, permute residuals (or bootstrap them) rather than raw responses.

4.35.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.35.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.36 Post-Hoc Tests with Tukey HSD

4.36.1 Introduction

When a one-way ANOVA rejects equality of means across groups, post-hoc tests identify which specific pairs differ. Tukey's HSD (honestly significant difference) is the default pairwise comparison test after ANOVA with equal sample sizes, controlling the family-wise error rate at the nominal α.

4.36.2 Prerequisites

One-way ANOVA, multiple-comparisons problem.

4.36.3 Theory

For k groups with sample size n each and common residual MS = MSE, the Tukey critical value for pairwise differences is

$$q_{\alpha,k,n_{\text{total}}-k} \cdot \sqrt{\text{MSE}/n},$$

where q is the Studentised range quantile. Pairs whose observed mean difference exceeds this are declared significantly different at family-wise α.

Tukey's HSD is equivalent to a simultaneous test based on the maximum pairwise t statistic. Alternatives:

- **Dunnett**: comparing each treatment to a single control; more powerful than Tukey when this is the question.
- **Bonferroni**: simple but conservative; often over-adjusts.
- **Holm**: sequential, uniformly more powerful than Bonferroni.
- **Scheffe**: most conservative; valid for arbitrary contrasts, not just pairwise.

4.36.4 Assumptions

Same as ANOVA: independence, normal residuals, homogeneous variances. For unequal sample sizes, the Tukey-Kramer variant is used.

4.36.5 R Implementation

```r
library(emmeans); library(multcomp)
set.seed(2026)

df <- data.frame(
  group = factor(rep(c("A", "B", "C", "D"), each = 30)),
  y     = c(rnorm(30, 50, 8),
            rnorm(30, 58, 8),
            rnorm(30, 60, 8),
            rnorm(30, 55, 8))
)

fit <- aov(y ~ group, data = df)

# Built-in Tukey HSD
TukeyHSD(fit)

# emmeans equivalent (more flexible, handles unbalanced designs)
emm <- emmeans(fit, ~ group)
pairs(emm, adjust = "tukey")
```

```
# Dunnett (comparing to A as control)
summary(glht(fit, linfct = mcp(group = "Dunnett")))
```

4.36.6 Output & Results

```
           diff    lwr    upr   p adj
B-A     8.421     3.02  13.82    0.0006
C-A     9.897     4.50  15.30    0.00005
D-A     4.975    -0.43  10.38    0.086
C-B     1.476    -3.92   6.88    0.892
D-B    -3.446    -8.85   1.96    0.353
D-C    -4.922   -10.32   0.48    0.090
```

Three of six pairwise comparisons are significant at family-wise $\alpha = 0.05$.

4.36.7 Interpretation

Report adjusted p-values and family-wise confidence intervals. "Tukey HSD post-hoc comparisons following a significant one-way ANOVA indicated that groups B and C each differed significantly from group A (both adjusted $p < 0.01$), while D did not differ significantly from A (adjusted $p = 0.09$)."

4.36.8 Practical Tips

- Only perform post-hoc tests after a significant omnibus F; otherwise you are fishing.
- For unequal sample sizes use Tukey-Kramer (R's `TukeyHSD` and `emmeans` handle this automatically).
- Dunnett is more powerful than Tukey when all comparisons are to one control.
- For complex (non-pairwise) contrasts, use `emmeans` or `glht` with user-defined linear combinations.
- Conservative alternatives (Bonferroni, Holm) work regardless of equal variances; they are safer in heterogeneous-variance settings.

4.36.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.36.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem

- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.37 Repeated-Measures ANOVA

4.37.1 Introduction

Repeated-measures ANOVA (rmANOVA) handles designs where the same subjects are measured at multiple time points or conditions. By treating subject as a blocking factor, it controls between-subject variability and gains power over between-subjects analysis. The main complication is the **sphericity** assumption on the covariance of the repeated measures.

4.37.2 Prerequisites

One-way ANOVA, dependence structure in repeated measures.

4.37.3 Theory

For one within-subjects factor with k levels measured on n subjects, the model is

$$Y_{ij} = \mu + \alpha_i + \pi_j + \varepsilon_{ij},$$

where π_j is a random subject effect and α_i is the fixed effect of level i.

Sphericity assumes the variances of all pairwise differences between repeated measurements are equal. Mauchly's test assesses this assumption.

If sphericity is violated:

- **Greenhouse-Geisser correction**: adjusts degrees of freedom by multiplying by $\hat{\varepsilon}_{\mathrm{GG}} \in [1/(k-1), 1]$.
- **Huynh-Feldt correction**: less conservative for mild violations.

Modern alternative: fit a linear mixed-effects model (`lmer`) with an unstructured or compound-symmetric covariance – more flexible, handles missing data.

4.37.4 Assumptions

- Normality of within-subject residuals.

- Sphericity (or apply a correction).
- Subjects are independent; measurements within a subject are not.

4.37.5 R Implementation

```r
library(afex); library(emmeans); library(rstatix)
set.seed(2026)

# 20 subjects measured at 4 time points
n_subj <- 20
times  <- 4
subject <- rep(1:n_subj, each = times)
time    <- rep(1:times, n_subj)

# Simulate with a time effect and subject-specific intercepts
df <- data.frame(
  subject = factor(subject),
  time    = factor(time),
  y       = rnorm(n_subj * times, mean = 50 + 2 * time, sd = 6) +
            rep(rnorm(n_subj, 0, 4), each = times)
)

# Using afex::aov_car
fit <- aov_car(y ~ time + Error(subject/time), data = df)
fit

# rstatix pipeline
df |> anova_test(dv = y, wid = subject, within = time)

# Post-hoc with Bonferroni
emm <- emmeans(fit, ~ time)
pairs(emm, adjust = "bonferroni")
```

4.37.6 Output & Results

```
Anova Table (Type 3 tests)

Response: y
   Effect         df  MSE       F  ges  p.value
1    time  2.61, 49.5 28.7  12.39  0.19  <.001 ***

Mauchly Tests for Sphericity:
 Effect   W       p
  time  0.87  0.78   # sphericity holds

Greenhouse-Geisser eps: 0.87
```

Large within-subject effect of time ($F(2.61, 49.5) = 12.39$, $p < 0.001$, generalised $\eta^2 = 0.19$); sphericity holds so no correction needed.

4.37.7 Interpretation

"A one-way repeated-measures ANOVA revealed a significant effect of time ($F(2.61, 49.5) = 12.39$, $p < 0.001$, generalised $\eta^2 = 0.19$; Greenhouse-Geisser correction applied). Bonferroni post-hoc comparisons identified significant increases from time 1 to time 3 and from time 2 to time 4."

4.37.8 Practical Tips

- Check Mauchly's test; when rejected, apply Greenhouse-Geisser or Huynh-Feldt.
- For complex designs (multiple within factors, between x within interactions), a linear mixed model is preferred.
- Missing data in rmANOVA causes the entire subject to be dropped (listwise); mixed models handle it gracefully.
- Report generalised η^2 for repeated-measures designs (partial η^2 is upwardly biased).
- For non-normal residuals, use Friedman or an aligned-rank transform (`ARTool::art`).

4.37.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.37.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.38 The Runs Test

4.38.1 Introduction

The runs test is a non-parametric test of randomness in a sequence of binary outcomes (or continuous outcomes that have been dichotomised against a reference such as the median). A "run" is a maximal subsequence of identical symbols — for example, in the sequence HHTTTHHH the runs are HH, TTT, HHH, three runs in total. Under genuine randomness the number of runs follows a known distribution; too few runs indicate clustering of like outcomes (suggesting positive autocorrelation or a structural break), while too many runs indicate alternation (negative autocorrelation). The runs test is widely used in quality-control charts to detect assignable causes, in regression diagnostics to flag autocorrelated residuals, and in randomness testing of sequences from random-number generators or dice-rolling sequences.

4.38.2 Prerequisites

A working understanding of binary sequences, the concept of independent identically distributed observations, and the role of randomness diagnostics in time-ordered or sequence data.

4.38.3 Theory

For a binary sequence with n_1 successes and n_2 failures, the expected number of runs under the null hypothesis of randomness is

$$E[R] = \frac{2n_1n_2}{n_1+n_2} + 1, \qquad \mathrm{Var}(R) = \frac{2n_1n_2(2n_1n_2 - n_1 - n_2)}{(n_1+n_2)^2(n_1+n_2-1)}.$$

The standardised statistic $Z = (R - E[R])/\sqrt{\mathrm{Var}(R)}$ is approximately standard Normal for moderate n (typically $n_1, n_2 \geq 10$). For continuous data, dichotomising against the sample median and applying the same test is the standard adaptation, though continuous-specific alternatives such as the Durbin-Watson test are usually more powerful for regression residuals.

4.38.4 Assumptions

The sequence is binary or has been dichotomised in a pre-specified way, observations under the null are independent and identically distributed Bernoulli, and the sample size is large enough for the Normal approximation. Tied values at the dichotomisation cut-off must be handled deliberately — dropped, or broken randomly with appropriate caution.

4.38.5 R Implementation

```r
library(tseries); library(randtests)
set.seed(2026)

random_seq <- rbinom(50, 1, 0.5)
runs.test(factor(random_seq))

clustered <- rep(c(0, 1), each = 25)
runs.test(factor(clustered))

alternating <- rep(c(0, 1), times = 25)
runs.test(factor(alternating))

x <- rnorm(50) + seq(0, 2, length.out = 50)
runs.test(factor(as.integer(x > median(x))))
```

4.38.6 Output & Results

The four examples illustrate the four distinct patterns the test detects: a truly random sequence ($z = -0.23$, $p = 0.82$), a strongly clustered sequence ($z = -6.85$, $p < 0.001$), an alternating sequence ($z = 6.85$, $p < 0.001$), and a trended continuous sequence dichotomised at the median ($z = -3.12$, $p = 0.002$). The sign of z distinguishes clustering (negative) from alternation (positive).

4.38.7 Interpretation

A reporting sentence: "The runs test applied to the residuals of the regression model showed no significant departure from randomness ($Z = -0.23$, $p = 0.82$), supporting the assumption of independent residuals along the observation order. For comparison, the test applied to a deliberately clustered binary sequence yielded $Z = -6.85$ ($p < 0.001$), and to an alternating sequence $Z = +6.85$ ($p < 0.001$), illustrating the test's sensitivity to both directions of non-randomness." Always describe whether departures are toward clustering or alternation.

4.38.8 Practical Tips

- Use the runs test as a quick-and-dirty diagnostic for autocorrelation or trend in residuals from regression or quality-control monitoring; for regression-specific autocorrelation diagnostics, the Durbin-Watson test or Breusch-Godfrey test on continuous residuals is more powerful and should be preferred.
- For continuous sequences, dichotomising at the median loses substantial information; the test becomes a relatively weak diagnostic compared with continuous-specific alternatives, but it remains useful for quick visual inspection of long sequences.

- Ties at the median require deliberate handling: dropping tied values reduces the effective sample size and is the standard recommendation; breaking ties randomly inflates the apparent randomness and should be done only with explicit caution.
- The Wald-Wolfowitz generalisation extends the runs concept to two-sample comparisons of distributions via run counts after merging and ranking; it is a non-parametric alternative to the two-sample t-test or Mann-Whitney test in some specialised settings.
- In statistical-process-control charts, runs tests are used to detect assignable causes — systematic shifts, trends, or oscillations — in a time-ordered sequence of measurements; Western Electric and Nelson rules are operational implementations of runs-test logic.
- For very long sequences ($n > 1000$), the Normal approximation is excellent and exact tables are unnecessary; for very short sequences ($n < 20$), use the exact distribution implemented in `randtests::runs.test()`.

4.38.9 R Packages Used

`tseries::runs.test()` and `randtests::runs.test()` for the canonical Wald-Wolfowitz runs test; `randtests` for an extensive collection of randomness tests including Bartels rank-test, turning-point test, and rank-version variants; `lawstat::runs.test()` for an alternative implementation; `lmtest::dwtest()` and `lmtest::bgtest()` for continuous-residual autocorrelation diagnostics complementing the runs test.

4.38.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.38.11 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.39 Shapiro-Wilk Normality Test

4.39.1 Introduction

The Shapiro-Wilk test, introduced by Shapiro and Wilk in 1965, is the most widely used and generally most powerful test of Normality for small-to-moderate sample sizes. It compares the observed sample order statistics with their expected values under Normality, producing a statistic W that is close to 1 when the sample is well approximated by a Normal distribution and substantially lower otherwise. Shapiro-Wilk is the default Normality test in R for samples between 3 and 5000 observations and is routinely used as a preliminary diagnostic before applying parametric inference (one- and two-sample t-tests, ANOVA, regression on the residuals) that nominally requires Normal data.

4.39.2 Prerequisites

A working understanding of order statistics, the Normal distribution, and the role of Normality assumptions in parametric inferential statistics.

4.39.3 Theory

For a sorted sample $x_{(1)} \leq \cdots \leq x_{(n)}$, the Shapiro-Wilk statistic is

$$W = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

where a_i are tabulated constants derived from the expected values and covariance matrix of the order statistics of a standard Normal sample. The numerator is essentially the squared correlation between the sample order statistics and their expected values, scaled to produce $W \in (0, 1]$. Values near 1 indicate a Normal sample, lower values indicate departure; small W with correspondingly small p-value rejects the null of Normality.

4.39.4 Assumptions

Observations are independent and identically distributed; sample size lies within R's implementation range $n \in [3, 5000]$. The test does not assume the sample is Normal under the null — the null is precisely that hypothesis — but it does assume the observations are otherwise exchangeable.

4.39.5 R Implementation

```
set.seed(2026)

x_norm <- rnorm(60)
```

```r
shapiro.test(x_norm)

x_skew <- rexp(60, rate = 1)
shapiro.test(x_skew)

x_t3 <- rt(60, df = 3)
shapiro.test(x_t3)

set.seed(2026)
pvals <- replicate(1000, shapiro.test(rt(20, df = 5))$p.value)
mean(pvals < 0.05)
```

4.39.6 Output & Results

The three samples illustrate the test's behaviour: a truly Normal sample of 60 observations gives $W = 0.98$, $p = 0.52$ (no rejection); an exponentially-distributed (right-skewed) sample of 60 gives $W = 0.83$, $p = 0.00002$ (strong rejection); a t_3-distributed (heavy-tailed) sample gives $W = 0.93$, $p = 0.002$ (rejection). The Monte Carlo block estimates empirical power at $n = 20$ for detecting t_5-distributed data — about 21 % — illustrating the test's modest power at small samples for mild departures.

4.39.7 Interpretation

A reporting sentence: "Shapiro-Wilk did not reject the Normality of the regression residuals ($W = 0.98$, $p = 0.52$), supporting the assumption underlying parametric inference. The complementary Q-Q plot showed close adherence to the diagonal, with no evidence of systematic deviation in the tails or at the centre. Parametric t-tests and ANOVA were therefore applied to these data without transformation. For comparison, residuals from a related model with skewed errors were rejected by the same test ($W = 0.83$, $p < 0.001$) and motivated a log transformation." Always pair test with a Q-Q plot.

4.39.8 Practical Tips

- At very small samples ($n < 15$), Shapiro-Wilk has low power against most realistic departures and cannot reliably confirm Normality; absence of rejection at small samples is uninformative and should not be treated as evidence of Normality.
- At very large samples ($n > 1000$), the test rejects at trivial departures from Normality that are inconsequential for the validity of parametric inference; use Q-Q plots and effect-size summaries (skewness, kurtosis with their CIs) rather than the p-value alone in large samples.
- Graphical diagnosis via a Q-Q plot complements the numerical test and is often more informative; always plot the Q-Q before deciding whether

departures are practically important. The combination of formal test and visual inspection is the modern recommendation.
- Normality assumptions in regression and ANOVA apply to the residuals, not to the raw outcome variable; testing Normality of the raw outcome is a recurring methodological error and provides no information about whether parametric inference is valid.
- When Normality fails substantively, consider transformations (log, Box-Cox, Yeo-Johnson), non-parametric alternatives (Wilcoxon, Mann-Whitney, Kruskal-Wallis), or robust methods (trimmed-mean inference, robust regression); the choice depends on the substantive question and the nature of the departure.
- For very heavy-tailed data, robust inference is often preferable to transformation; transformations can shift the substantive interpretation away from the original scale.

4.39.9 R Packages Used

Base R `shapiro.test()` for the canonical Shapiro-Wilk implementation up to $n = 5000$; `nortest::shapiro.test()` for an alternative interface; `nortest` for related Normality tests including Anderson-Darling, Cramér-von Mises, and Lilliefors; `moments::skewness()` and `moments::kurtosis()` for descriptive moment-based summaries; `car::qqPlot()` for enhanced Q-Q plot diagnostics complementing the formal test.

4.39.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.39.11 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.40 The Sign Test

4.40.1 Introduction

The sign test is the most assumption-light procedure for a one-sample median or paired comparison. It ignores the magnitude of deviations and uses only whether each observation is above or below the reference value (or whether each pair increased or decreased).

4.40.2 Prerequisites

Binomial distribution, paired data.

4.40.3 Theory

For iid $X_1, \ldots, X_n$ and a hypothesised median m_0, let $S = \#\{X_i > m_0\}$. Under H_0: median $= m_0$,

$$S \sim \text{Binomial}(n', 0.5),$$

where n' excludes any $X_i = m_0$. The two-sided p-value is computed from the binomial PMF.

For paired data, apply the test to the signs of the paired differences.

The sign test has lower power than the Wilcoxon signed-rank when the symmetry assumption of the latter holds. Its advantage is that it works under **any** continuous distribution.

4.40.4 Assumptions

- iid observations (or independent pairs).
- Continuous distribution (else need to handle ties at the reference).

No symmetry, no shape assumption.

4.40.5 R Implementation

```r
library(BSDA)
set.seed(2026)

# One-sample: test whether median equals 100
x <- rlnorm(30, meanlog = log(105), sdlog = 0.3)
SIGN.test(x, md = 100)

# Paired: test whether intervention changes a score
pre  <- rpois(25, 5)
```

```r
post <- pre + sample(-1:3, 25, replace = TRUE)
SIGN.test(post, pre, md = 0)
```

4.40.6 Output & Results

```
        One-sample Sign-Test
data:  x
s = 21, p-value = 0.021
alternative hypothesis: true median is not equal to 100
95 percent confidence interval: 97.5 to 120

        Dependent-samples Sign-Test
data:  post and pre
S = 19, p-value = 0.016
95 percent CI for median difference: 0.5 to 2.0
```

Both tests reject: the one-sample median significantly exceeds 100; paired differences are significantly positive.

4.40.7 Interpretation

"The median post-intervention score was 7.0 (95 % CI 0.5 to 2.0 for the paired difference), significantly higher than pre (sign test, 19 of 25 positive differences, exact binomial p = 0.016)."

4.40.8 Practical Tips

- Use the sign test when the distribution of paired differences is badly skewed or clearly not symmetric.
- Report the number of positive, negative, and zero differences; the latter are excluded from the test.
- The sign test's 95 % CI for the median is a distribution-free interval based on order statistics.
- With many ties at the reference, consider the Wilcoxon or rank-based alternative, understanding their stronger assumptions.
- Power is lower than Wilcoxon signed-rank; prefer Wilcoxon when its symmetry assumption is plausible.

4.40.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.40.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.41 Spearman Rank Correlation Test

4.41.1 Introduction

Spearman's ρ (rho) is the Pearson correlation of the ranks of two variables. It detects monotonic associations that need not be linear, and is robust to outliers and heavy tails. It is the default correlation measure for ordinal data or strongly skewed continuous data.

4.41.2 Prerequisites

Pearson correlation, ranks.

4.41.3 Theory

Replace each X_i by its rank R_i, and each Y_i by its rank S_i. Spearman's ρ is

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2-1)}, \qquad d_i = R_i - S_i,$$

when there are no ties. With ties, the general form

$$\rho = \frac{\sum (R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum (R_i - \bar{R})^2 \sum (S_i - \bar{S})^2}}$$

applies.

Under $H_0 : \rho = 0$, an asymptotic normal approximation holds; for small n exact or permutation distributions are used. Confidence intervals for Spearman's ρ can be obtained by Fisher's z applied to the ranks.

4.41.4 Assumptions

- Ordinal or continuous data; the ranks must be well-defined.
- Monotonic association (linear or non-linear).
- Independent pairs.

4.41.5 R Implementation

```r
set.seed(2026)

n <- 50
x <- rgamma(n, shape = 2, rate = 0.5)     # skewed
y <- x^2 + rnorm(n, sd = 4)               # non-linear monotonic

# Pearson (fails for non-linear)
cor.test(x, y, method = "pearson")

# Spearman
cor.test(x, y, method = "spearman", exact = FALSE)

# Bootstrap 95% CI for Spearman's rho
library(boot)
rho_boot <- boot(data = data.frame(x, y),
                 statistic = function(d, i) cor(d$x[i], d$y[i], method =
                  ↪  "spearman"),
                 R = 2000)
boot.ci(rho_boot, type = "perc")
```

4.41.6 Output & Results

```
        Pearson's product-moment correlation
t = 11.45, df = 48, p < 0.001
cor: 0.855

        Spearman's rank correlation rho
S = 100, p < 0.001
rho: 0.995

Bootstrap 95% CI (percentile): (0.988, 0.999)
```

Spearman recovers the strong monotonic relationship ($\rho \approx 1$) that Pearson underestimates because of the non-linearity.

4.41.7 Interpretation

"There was a strong monotonic association between x and y (Spearman's $\rho = 0.99$, bootstrap 95 % CI 0.99 to 1.00; $p < 0.001$)."

4.41.8 Practical Tips

- Use Spearman for ordinal data, heavy-tailed or skewed continuous data, or suspected non-linear monotonic relationships.
- For very small n, use exact methods (`cor.test(...,exact=TRUE)`; only valid without ties).
- With ties, the normal approximation is used; the warning is expected and safe.
- Kendall's τ is an alternative for small samples or many ties.
- Spearman captures monotonic, not linear, associations; for U-shaped relationships it can still miss dependence.

4.41.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.41.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.42 Two-Proportion Test

4.42.1 Introduction

The two-proportion test compares success rates in two independent groups. Typical applications: treatment vs. control event rates, exposed vs. unexposed incidence, screen-positive rates across populations. The test comes in two algebraically equivalent forms – a z-test on the difference and a 2x2 chi-squared – and is typically reported together with a risk difference, relative risk, or odds ratio.

4.42.2 Prerequisites

One-proportion test, chi-squared distribution.

4.42.3 Theory

For $X_1 \sim \text{Binomial}(n_1, p_1)$ and $X_2 \sim \text{Binomial}(n_2, p_2)$ independent,

$$H_0 : p_1 = p_2 \quad \text{vs.} \quad H_1 : p_1 \neq p_2.$$

Pooled proportion under H_0: $\hat{p} = (x_1 + x_2)/(n_1 + n_2)$.

Z-test:

$$Z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})(1/n_1 + 1/n_2)}}.$$

Z^2 equals the chi-squared statistic on the 2x2 table. In R, `prop.test()` performs this test (using chi-squared form).

Effect-size summaries:

- Risk difference (RD): $\hat{p}_1 - \hat{p}_2$.
- Relative risk (RR): $\hat{p}_1/\hat{p}_2$.
- Odds ratio (OR): $\hat{p}_1(1-\hat{p}_2)/[\hat{p}_2(1-\hat{p}_1)]$.

Each has a standard-error formula for constructing Wald CIs; score-based and exact alternatives exist.

4.42.4 Assumptions

- Independent Bernoulli trials within and between groups.
- Expected counts ≥ 5 in every cell (else use Fisher's exact test).

4.42.5 R Implementation

```r
library(epitools)

x1 <- 48; n1 <- 200       # treatment
x2 <- 32; n2 <- 200       # control

prop.test(c(x1, x2), c(n1, n2), correct = FALSE)

# Risk difference with 95% CI via prop.test
prop.test(c(x1, x2), c(n1, n2), correct = FALSE)$conf.int

# Relative risk and odds ratio
riskratio(c(n2 - x2, n1 - x1, x2, x1), rev = "both")$measure
oddsratio(c(n2 - x2, n1 - x1, x2, x1), rev = "both")$measure

# Fisher's exact test (small-cell alternative)
```

```r
fisher.test(matrix(c(x1, n1 - x1, x2, n2 - x2), nrow = 2, byrow = TRUE))
```

4.42.6 Output & Results

```
     2-sample test for equality of proportions without continuity
     ↪  correction
X-squared = 3.66, df = 1, p-value = 0.0557
95% CI for (p1 - p2): (-0.0014, 0.1614)
sample estimates: prop 1 = 0.24, prop 2 = 0.16

           estimate lower    upper
Exposed        1.50 0.994     2.265
              (relative risk 1.50)
```

Treatment success rate 24 %, control 16 %. Risk difference 8 % (95 % CI -0.1 % to 16.1 %); p = 0.056 (borderline). Relative risk 1.50 (95 % CI 0.99 to 2.27).

4.42.7 Interpretation

Report counts, proportions, the effect-size estimate with its CI, and the test statistic. "In the treatment arm, 48/200 (24.0 %) experienced the event, compared to 32/200 (16.0 %) in control (risk difference 8.0 %, 95 % CI -0.1 % to 16.1 %; $\chi^2(1) = 3.66$, p = 0.056)."

4.42.8 Practical Tips

- Report at least one of RD, RR, or OR as an effect size; significance alone is unhelpful.
- Use Fisher's exact test when expected counts are below 5.
- For matched/paired binary data, use McNemar's test, not the two-proportion test.
- The chi-squared test and the two-proportion z-test give identical two-sided p-values when `correct=FALSE`.
- Be aware that RR and OR differ in magnitude when outcomes are common ($p > 0.10$); choose the scale that fits the clinical question.

4.42.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.42.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.43 Two-Sample t-Test

4.43.1 Introduction

The two-sample t-test is the default procedure for deciding whether the means of two independent groups differ. In its modern form it comes in two flavours: Student's t-test, which assumes equal variances in the two populations, and Welch's t-test, which does not. Both produce a test statistic, a p-value, and a confidence interval for the difference in means.

4.43.2 Prerequisites

A reader should understand the difference between a population mean and a sample mean, the concept of a sampling distribution, and the logic of hypothesis testing. Familiarity with R's formula interface is assumed.

4.43.3 Theory

Let $X_1, \dots, X_{n_1}$ be independent observations from $\mathcal{N}(\mu_1, \sigma_1^2)$ and $Y_1, \dots, Y_{n_2}$ independent observations from $\mathcal{N}(\mu_2, \sigma_2^2)$. We test $H_0 : \mu_1 = \mu_2$ against $H_1 : \mu_1 \neq \mu_2$.

Student's test assumes $\sigma_1^2 = \sigma_2^2 = \sigma^2$ and uses the pooled variance estimator. The test statistic

$$T = \frac{\bar{X} - \bar{Y}}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

follows a t distribution with $n_1 + n_2 - 2$ degrees of freedom under H_0, where s_p^2 is the pooled sample variance.

Welch's test allows $\sigma_1^2 \neq \sigma_2^2$ and uses the Satterthwaite approximation:

$$T = \frac{\bar{X} - \bar{Y}}{\sqrt{s_1^2/n_1 + s_2^2/n_2}},$$

with degrees of freedom

$$\nu = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}}.$$

Modern recommendations favour Welch's test as the default, because the assumption of equal variances is rarely verifiable in practice and the loss of efficiency when the variances *are* equal is small.

4.43.4 Assumptions

Both tests assume:

1. Observations within each group are independent of each other.
2. The two groups are independent of each other.
3. The observations in each group are approximately normal, or the sample size is large enough for the CLT to apply to the sample means.

Student's additionally assumes equal variances. Welch's does not.

4.43.5 R Implementation

```r
library(effectsize)

set.seed(42)
group_a <- rnorm(30, mean = 100, sd = 15)
group_b <- rnorm(35, mean = 110, sd = 18)

df <- data.frame(
  score = c(group_a, group_b),
  group = factor(rep(c("A", "B"), c(30, 35)))
)

t.test(score ~ group, data = df)

cohens_d(score ~ group, data = df)
```

The default `t.test()` in R performs Welch's test. For Student's, set `var.equal=TRUE`. The `effectsize::cohens_d()` function returns Cohen's d with a confidence interval.

4.43.6 Output & Results

Typical output:

```
        Welch Two Sample t-test

data:  score by group
t = -2.3415, df = 61.832, p-value = 0.02241
alternative hypothesis: true difference in means between group A and
 ↪  group B is not equal to 0
95 percent confidence interval:
 -18.8912  -1.4872
sample estimates:
mean in group A mean in group B
       100.4510        110.6407
```

Cohen's d is approximately -0.59 with 95% CI $(-1.09, -0.09)$ – a moderate effect.

4.43.7 Interpretation

The conventional report is: "group B had a mean score of 110.6 (SD 18.2), which was significantly higher than group A's mean of 100.5 (SD 14.5); Welch's $t(61.8) = -2.34$, $p = 0.022$, mean difference = 10.2 (95% CI 1.5 to 18.9), Cohen's $d = -0.59$." The effect-size reporting is essential: a p-value alone does not tell the reader how large the difference is.

4.43.8 Practical Tips

- Always report the effect size and its confidence interval alongside the p-value; journals increasingly require this.
- Prefer Welch's test as the default; only use Student's when equal variances are genuinely plausible (for example, a randomised experiment with balanced groups).
- Check normality with Q-Q plots rather than with Shapiro-Wilk – the latter has little power at small n and rejects trivially at large n.
- For strongly skewed or outlier-contaminated data, consider the Wilcoxon rank-sum test, a permutation test, or analysis on a log scale.
- Unequal sample sizes are fine; extreme imbalance (say, 10 vs. 1000) warrants extra care because variance assumptions and power are both affected.

4.43.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

4.43.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.44 Two-Way ANOVA

4.44.1 Introduction

Two-way ANOVA tests the effects of two categorical factors on a continuous outcome, plus their interaction. It generalises one-way ANOVA by accommodating designs that manipulate or measure two conditions simultaneously, and is the simplest example of a factorial design.

4.44.2 Prerequisites

One-way ANOVA, factorial designs.

4.44.3 Theory

For a two-way design with factors A and B:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \varepsilon_{ijk},$$

where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. The ANOVA partitions the total sum of squares into:

- Main effect of A (df = a - 1).
- Main effect of B (df = b - 1).
- Interaction AxB (df = (a - 1)(b - 1)).
- Residual (df = ab(n - 1) for balanced design with n per cell).

Three null hypotheses are tested: no main A effect, no main B effect, no interaction. The F-tests use the residual MS as denominator.

Type I, II, III sums of squares differ when the design is unbalanced:

- Type I: sequential; order-dependent.
- Type II: each effect adjusted for others except interactions involving it.

- Type III: each effect adjusted for all others; default in most modern practice when interactions are present.

4.44.4 Assumptions

- Independent observations.
- Normal residuals.
- Homogeneous variances across cells.
- For balanced designs all SS types agree; for unbalanced ones, the choice matters.

4.44.5 R Implementation

```r
library(car); library(effectsize); library(emmeans)
set.seed(2026)

df <- expand.grid(A = factor(c("A1", "A2")),
                  B = factor(c("B1", "B2", "B3")),
                  rep = 1:20)
df$y <- with(df, 50 + 3*(A == "A2") + 2*(B == "B2") + 4*(B == "B3") +
               5*(A == "A2" & B == "B3") + rnorm(nrow(df), 0, 6))

# Fit and Type III ANOVA
fit <- lm(y ~ A * B, data = df, contrasts = list(A = contr.sum, B =
 ↪  contr.sum))
Anova(fit, type = 3)

# Effect sizes
eta_squared(fit, partial = TRUE)

# Marginal means and interaction plot
emm <- emmeans(fit, ~ A * B)
emm
pairs(emm, simple = "A")

# Simple interaction plot
interaction.plot(df$B, df$A, df$y, col = c("#2A9D8F", "#F4A261"),
                 lwd = 2, main = "Interaction plot")
```

4.44.6 Output & Results

```
Anova Table (Type III tests)

            Sum Sq  Df  F value  Pr(>F)
(Intercept) 291 000  1  8036    <2e-16
A             168   1   4.64  0.0332
```

```
B             490   2   6.77  0.0017
A:B           186   2   2.57  0.0819
Residuals  4 123 114

Eta2 (partial) | 95% CI
------------------------
A              | 0.04
B              | 0.11
A:B            | 0.04
```

Main effects of A and B are significant; the interaction is borderline.

4.44.7 Interpretation

"In a two-way ANOVA, both A ($F(1, 114) = 4.64$, $p = 0.033$) and B ($F(2, 114) = 6.77$, $p = 0.002$) had significant main effects; the A x B interaction was not significant ($F(2, 114) = 2.57$, $p = 0.082$)."

4.44.8 Practical Tips

- Use Type III SS with sum-to-zero contrasts for standard interpretation.
- Always inspect the interaction plot; visual patterns often matter more than p-values.
- Unbalanced designs need careful contrast specification; `car::Anova` handles the SS types explicitly.
- Report effect sizes (partial η^2) alongside F.
- For significant interactions, interpret simple effects, not the main effects alone.

4.44.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.44.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds

- Pearson and Spearman correlation
- Non-parametric tests

4.45 Type I and Type II Errors

4.45.1 Introduction

Two kinds of error arise in hypothesis testing: rejecting a true null (Type I) and failing to reject a false null (Type II). Their probabilities – α and β – are the core quantities of statistical decision theory. Power $1-\beta$ is one minus the Type II rate, and the rate at which sample size and effect size trade against α is the currency of study design.

4.45.2 Prerequisites

Hypothesis testing, p-values, sampling distributions.

4.45.3 Theory

The 2x2 decision table:

	H_0 true	H_0 false
Reject H_0	Type I (α)	correct (power)
Fail to reject H_0	correct	Type II (β)

- $\alpha = P(\text{reject } H_0 \mid H_0)$ is the significance level, usually set to 0.05.
- $\beta = P(\text{fail to reject } H_0 \mid H_1)$ is the Type II error rate.
- Power $= 1 - \beta$, usually targeted at 0.80 or 0.90.

For a given test, power depends on: the true effect size (larger $\Rightarrow$ higher power), the sample size (larger $\Rightarrow$ higher power), the significance level (α smaller $\Rightarrow$ power lower), and the variability of the outcome (larger SD $\Rightarrow$ lower power).

Trade-off: tightening α (to 0.01, say) reduces Type I errors but increases Type II errors for a fixed sample size. Only increasing n reduces both.

4.45.4 Assumptions

The test is correctly specified; the distributional model holds under H_0 and H_1.

4.45.5 R Implementation

```
library(ggplot2)
set.seed(2026)
```

```r
# Simulation: Type I rate at alpha = 0.05 when H0 true
n_sims <- 10000
p_null <- replicate(n_sims, t.test(rnorm(30, 0, 1), mu = 0)$p.value)
mean(p_null < 0.05)

# Power at alpha = 0.05 when H1 true (effect size = 0.5)
p_alt <- replicate(n_sims, t.test(rnorm(30, 0.5, 1), mu = 0)$p.value)
mean(p_alt < 0.05)

# Compare to theoretical power
power.t.test(n = 30, delta = 0.5, sd = 1, type = "one.sample")

# Power curve
d_grid <- seq(0, 1, by = 0.05)
pw <- sapply(d_grid, function(d)
  power.t.test(n = 30, delta = d, sd = 1, type = "one.sample")$power)

ggplot(data.frame(d = d_grid, power = pw), aes(d, power)) +
  geom_line(colour = "#2A9D8F", linewidth = 1) +
  geom_hline(yintercept = 0.8, linetype = "dashed", colour = "#F4A261") +
  labs(x = "Effect size d", y = "Power",
       title = "Power curve at alpha = 0.05, n = 30") +
  theme_minimal()
```

4.45.6 Output & Results

```
[1] 0.0510    # empirical Type I rate
[1] 0.7549    # empirical power at d = 0.5
```

Empirical Type I error is close to the nominal 0.05; empirical power at $d = 0.5$ matches the theoretical calculation.

4.45.7 Interpretation

In study design, sample size is chosen to achieve a pre-specified power at the minimum effect size of interest. Reporting post-hoc power based on observed effects is misleading and discouraged.

4.45.8 Practical Tips

- Power at 80 % is the conventional minimum; 90 % for confirmatory trials.
- Type I and II errors are fixed by the design, not by the data; a significant p-value does not "justify" the design after the fact.
- In multiple-comparison settings, family-wise error rate replaces single-test α.

- Sequential designs (group-sequential trials) adjust α across looks to preserve overall Type I rate.
- Heavy-tailed or non-normal data reduce power; use rank-based tests or robust alternatives when applicable.

4.45.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.45.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.46 Welch's t-Test

4.46.1 Introduction

Welch's t-test is the standard two-sample mean-comparison test when variances across groups may differ. R's `t.test()` defaults to Welch (`var.equal=FALSE`). Decades of simulation evidence favour Welch as the default choice regardless of whether variances appear equal; the efficiency cost versus the equal-variance (Student) test is negligible when variances are equal, and the robustness gain is substantial when they are not.

4.46.2 Prerequisites

Two-sample t-test, variance, degrees of freedom.

4.46.3 Theory

Given independent samples of sizes n_1, n_2 with means $\bar{X}_1, \bar{X}_2$ and sample variances s_1^2, s_2^2, the Welch test statistic is

$$T = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}}.$$

The Welch-Satterthwaite approximation gives the degrees of freedom

$$\nu = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}}.$$

ν is generally not an integer. Under $H_0 : \mu_1 = \mu_2$, $T \sim t_\nu$ approximately.

4.46.4 Assumptions

- Independent observations within and across groups.
- Approximately normal observations (or large n).
- No extreme outliers.

Equal variances are **not** required.

4.46.5 R Implementation

```r
library(effectsize); library(broom)
set.seed(2026)

n1 <- 25; n2 <- 40
group1 <- rnorm(n1, mean = 100, sd = 8)
group2 <- rnorm(n2, mean = 106, sd = 18)   # larger variance

df <- data.frame(
  x     = c(group1, group2),
  group = factor(rep(c("A", "B"), c(n1, n2)))
)

# Welch (default)
t.test(x ~ group, data = df) |> tidy()

# Student (var.equal = TRUE) for comparison
t.test(x ~ group, data = df, var.equal = TRUE) |> tidy()

# Effect size
cohens_d(x ~ group, data = df)
```

4.46.6 Output & Results

```
# Welch
```

```
estimate1 = 100.8, estimate2 = 106.7, t = -1.77, df = 58.9, p = 0.082

# Student (for comparison)
estimate1 = 100.8, estimate2 = 106.7, t = -1.56, df = 63,  p = 0.124

Cohen's d | 95% CI
-------------------
-0.40     | [-0.93, 0.14]
```

Welch's df (58.9) is smaller than Student's (63) because the variances differ; the Welch p-value is consequently smaller in this case, though both fail to reject.

4.46.7 Interpretation

Reporting "Welch's t(58.9) = -1.77, p = 0.082" makes the Welch variant explicit. Cohen's d uses a pooled SD by default; for very unequal variances the "Glass's delta" alternative uses only one group's SD.

4.46.8 Practical Tips

- Do not run Levene's test to decide between Welch and Student; a conditional choice inflates Type I error. Default to Welch.
- Satterthwaite df can be fractional; report it as a number with decimals.
- Welch's test is slightly more conservative than Student when variances are equal; the cost is negligible.
- For rank-based alternatives, use Mann-Whitney U.
- Welch-style corrections exist for ANOVA (Welch's F) and linear models (heteroscedasticity-consistent SEs).

4.46.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.46.10 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds

- Pearson and Spearman correlation
- Non-parametric tests

4.47 Wilcoxon Signed-Rank Test

4.47.1 Introduction

The Wilcoxon signed-rank test compares two paired measurements without requiring Normality of their differences. It is the rank-based non-parametric analogue of the paired t-test and is the appropriate choice when the paired differences are ordinal, when the difference distribution is skewed or heavy-tailed, or when sample sizes are too small for the parametric test's distributional assumptions to be defensible. The Wilcoxon signed-rank test is widely used in pre-post studies, in within-subject A/B comparisons, in sensory evaluation panels, and in any setting where paired measurements are compared and Normality is in doubt.

4.47.2 Prerequisites

A working understanding of paired-sample inference, the rank-based testing framework, and the assumption of symmetric distribution of paired differences (rather than Normal or any specific shape).

4.47.3 Theory

For paired observations (X_i, Y_i), compute differences $D_i = X_i - Y_i$, drop any zero differences, rank the absolute values $|D_i|$, and sum the ranks assigned to positive differences (W_+) and to negative differences (W_-). Under the null hypothesis that the median of D is zero (assuming the distribution of D is symmetric), W_+ and W_- have a known discrete distribution; for large n, the standardised statistic is approximately Normal. The rank-biserial correlation $r_{rb} = (W_+ - W_-)/T$, with T the sum of all ranks, provides an effect-size measure on $[-1, 1]$.

If the symmetry assumption fails substantively, the sign test offers a more assumption-light alternative at the cost of reduced power.

4.47.4 Assumptions

The observations are genuinely paired, the distribution of paired differences is approximately symmetric around the median, and the outcomes are at least ordinal. The test does not assume Normality of the differences — only symmetry — which is its principal advantage over the paired t-test.

4.47.5 R Implementation

```r
library(effectsize)
set.seed(2026)

n <- 25
pre  <- sample(1:7, n, replace = TRUE)
post <- pre + sample(-1:2, n, replace = TRUE, prob = c(0.1, 0.2, 0.4,
 ↪  0.3))

wilcox.test(pre, post, paired = TRUE)
rank_biserial(pre, post, paired = TRUE)

c(median_diff = median(post - pre),
  IQR_diff    = IQR(post - pre))
```

4.47.6 Output & Results

The pre-post comparison on a 7-point ordinal scale gives the Wilcoxon signed-rank statistic $V = 35$ ($p = 0.003$), with a paired rank-biserial correlation of -0.72 (95 % CI -0.90 to -0.40) classed as a large effect. The median paired difference of +1.0 (IQR 0 to 1) characterises the central tendency of the change on the original scale, and reporting both the test statistic and the effect-size summary is the standard expected by modern guidelines.

4.47.7 Interpretation

A reporting sentence: "Scores increased significantly from pre-treatment to post-treatment (Wilcoxon signed-rank $V = 35$, $p = 0.003$); median paired difference was +1.0 point on the 7-point scale (IQR 0 to 1, range -1 to +3), and the paired rank-biserial correlation was -0.72 (95 % CI -0.90 to -0.40), classed as a large effect. The non-parametric test was chosen because the differences were ordinal and a parametric paired t-test would not be appropriate." Always report median, IQR, and an effect-size measure.

4.47.8 Practical Tips

- Report the median and IQR of the paired differences rather than the raw pre/post means; the median is the location parameter that the Wilcoxon signed-rank test inferentially targets, and the means are not directly relevant to the rank-based test.
- Handle zero differences as R does by default (drop them from the analysis), or use the Pratt convention (include them in the ranking but assign sign zero); the choice can matter for small samples and should be documented.
- For small n and no ties in the absolute differences, use `exact=TRUE` to obtain exact p-values rather than the Normal approximation; the exact

distribution is the more accurate inference at small samples.
- The symmetry-of-differences assumption can be checked informally with a histogram of D_i or a Q-Q plot of D_i against the Normal; if the differences are markedly skewed, the sign test is a more assumption-light alternative.
- For paired binary outcomes (yes/no observed twice on the same subject), McNemar's test — not the Wilcoxon signed-rank — is the appropriate paired-sample test; conflating the two is a recurring methodological error.
- The Wilcoxon signed-rank test is sensitive to differences in median location under the symmetry assumption; for general distributional comparisons of paired data, the paired Cliff's delta or a one-sample Kolmogorov-Smirnov test on the differences are more general alternatives.

4.47.9 R Packages Used

Base R `wilcox.test(paired=TRUE)` for the canonical signed-rank test; `effectsize::rank_biserial()` for the paired rank-biserial correlation effect size; `coin::wilcoxsign_test()` for permutation-based exact p-values; `rcompanion::wilcoxonPairedR()` for an alternative effect-size implementation; `BSDA::SIGN.test()` for the more assumption-light sign-test alternative.

4.47.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

4.47.11 See also — labs in this chapter

- Populations, samples, and the central limit theorem
- Bootstrap and permutation tests
- Maximum likelihood estimation
- One-sample t-test and one-proportion test
- Hypothesis testing, p-values, type I/II errors
- Two-sample and paired t-tests
- Two proportions, chi-square, risk and odds
- Pearson and Spearman correlation
- Non-parametric tests

4.48 Week 3, Session 2 — Bootstrap and permutation tests

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.48.1 Learning objectives

- Compute a bootstrap confidence interval for a statistic whose sampling distribution is not easily derived.
- Run a permutation test for a two-group comparison and state what null hypothesis it tests.
- Explain when bootstrap and permutation answer different questions.

4.48.2 Prerequisites

Lab 3.1.

4.48.3 Background

Classical inference writes down the sampling distribution of an estimator analytically, usually by invoking the central limit theorem and assuming a parametric family for the data. Resampling methods replace this step with computation. A bootstrap confidence interval for a statistic is built by drawing repeated samples *with replacement* from the observed data and recomputing the statistic; the empirical distribution of the recomputed values is treated as a proxy for the sampling distribution.

A permutation test answers a different question. It randomly reassigns the group labels on the observed data and recomputes the test statistic; the empirical distribution of the recomputed values is the sampling distribution under the null of *exchangeability* — that is, under a null in which the group labels are interchangeable. It gives a p-value without any assumption about the shape of the underlying distributions.

The two techniques look similar — both involve a loop and a `replicate()` — but they are different tools. Bootstrap estimates uncertainty under the observed data-generating process; permutation tests a sharp null of no association.

4.48.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.48.5 1. Hypothesis

Two hypotheses in this lab:

1. For a sample median of body mass in Gentoo penguins, build a 95% bootstrap CI.
2. For flipper length in Gentoo vs Adelie penguins, test the sharp null that the distributions are exchangeable against the two-sided alternative.

4.48.6 2. Visualise

Use `palmerpenguins`.

```
library(palmerpenguins)
dat <- penguins |>
  filter(species %in% c("Gentoo", "Adelie")) |>
  drop_na(body_mass_g, flipper_length_mm)
```

```
dat |>
  ggplot(aes(species, flipper_length_mm, fill = species)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "Flipper length (mm)") +
  theme(legend.position = "none")
```

4.48.7 3. Assumptions

Bootstrap: the observed sample is representative of the population we wish to generalise to. Resampling with replacement preserves the univariate structure but assumes exchangeability of observations.

Permutation: under H0, group labels can be reshuffled without changing the joint distribution. The test does not assume normality or equal variances.

4.48.8 4. Conduct

4.48.8.1 Bootstrap CI for the median body mass of Gentoo

```
gen <- dat |> filter(species == "Gentoo") |> pull(body_mass_g)
B <- 2000
boot_meds <- replicate(B, median(sample(gen, replace = TRUE)))
ci_med <- quantile(boot_meds, c(0.025, 0.975))
obs_med <- median(gen)
tibble(statistic = "median body mass",
       observed = obs_med,
       ci_low = ci_med[1],
       ci_high = ci_med[2])
```

```
tibble(boot = boot_meds) |>
  ggplot(aes(boot)) +
  geom_histogram(bins = 40, fill = "grey60", colour = "white") +
  geom_vline(xintercept = obs_med, colour = "firebrick") +
  geom_vline(xintercept = ci_med, linetype = 2) +
  labs(x = "Bootstrap median", y = "Count")
```

4.48.8.2 Permutation test for flipper length Gentoo vs Adelie

```
g <- dat$species
x <- dat$flipper_length_mm
observed_diff <- mean(x[g == "Gentoo"]) - mean(x[g == "Adelie"])

perm_diff <- replicate(5000, {
  gp <- sample(g)
  mean(x[gp == "Gentoo"]) - mean(x[gp == "Adelie"])
})
p_perm <- mean(abs(perm_diff) >= abs(observed_diff))
p_perm
```

```
tibble(perm = perm_diff) |>
  ggplot(aes(perm)) +
  geom_histogram(bins = 50, fill = "grey60", colour = "white") +
  geom_vline(xintercept = observed_diff, colour = "firebrick", linewidth
  ↪  = 1) +
  labs(x = "Permuted mean difference", y = "Count")
```

4.48.9 5. Concluding statement

> Based on $B = 2000$ bootstrap resamples, the median body mass in Gentoo penguins ($n =$ `length(gen)`) was `round(obs_med,0)` g (95% percentile bootstrap CI `round(ci_med[1],0)` to `round(ci_med[2],0)` g). Flipper length was much longer in Gentoo than Adelie (mean difference `round(observed_diff,1)` mm); a permutation test over 5000 reshuffles produced $p =$ `format(p_perm,scientific=FALSE)`, providing strong evidence against exchangeability.

Bootstrap and permutation give you parametric-free tools for estimation and testing respectively. Neither rescues you from a biased sample; both depend on the data at hand being representative.

4.48.10 Common pitfalls

- Using bootstrap SEs for a statistic with heavy tails (eg the maximum) — it does not have a well-defined bootstrap distribution.
- Permutation with more than two groups: you still need a single test statistic (eg F), not pairwise comparisons by default.
- Reading a bootstrap CI as a confidence interval for the population parameter when the sample is biased.

4.48.11 Further reading

- Efron B, Tibshirani RJ. *An Introduction to the Bootstrap.*

- Good PI. *Permutation, Parametric, and Bootstrap Tests of Hypotheses*.

4.48.12 Session info

```
sessionInfo()
```

4.48.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.49 Week 3, Session 5 — Hypothesis testing, p-values, type I/II errors

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.49.1 Learning objectives

- State what a p-value is, and — just as importantly — what it is not.
- Demonstrate by simulation that p-values are uniform under H0.
- Quantify type I error, type II error, and power, and show their relationship.

4.49.2 Prerequisites

Lab 3.4.

4.49.3 Background

A p-value is the probability of observing a test statistic at least as extreme as the one computed, under the null hypothesis. It is not the probability that the null is true. It is not the probability that the result was due to chance. Both of those are common, plausible misreadings; both are wrong. The p-value is a function of the data under one specific hypothetical: *if H0 were true, how often would we see this much apparent signal?*

Under H0, the p-value is uniformly distributed on [0, 1]. This is a direct consequence of the probability integral transform, and it is the reason the familiar threshold 0.05 gives a 5% type I error rate when applied across many independent tests of true nulls.

Type I error is the rate of false positives; type II error is the rate of false negatives. Power is 1 − type II error: the probability of detecting a true effect. Power depends on the effect size, the variability, the sample size, and α. Increasing n buys you power; increasing α buys you power at the cost of more false positives.

4.49.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.49.5 1. Hypothesis

The lab has two claims to verify:

1. Under H0 (no effect), the p-value from a two-sample t-test is uniform on $[0, 1]$.
2. Under H1 (effect present), the probability of rejecting H0 depends on sample size and effect size.

4.49.6 2. Visualise

Generate many datasets with no effect, run a t-test on each, and plot the distribution of p-values.

```
simulate_pvals <- function(n_per_group = 30, delta = 0, B = 5000) {
  replicate(B, {
    a <- rnorm(n_per_group, 0, 1)
    b <- rnorm(n_per_group, delta, 1)
    t.test(a, b)$p.value
  })
}
p_h0 <- simulate_pvals(30, delta = 0)
```

```
tibble(p = p_h0) |>
  ggplot(aes(p)) +
  geom_histogram(bins = 20, fill = "grey60", colour = "white") +
  geom_hline(yintercept = length(p_h0) / 20,
             linetype = 2, colour = "firebrick") +
  labs(x = "p-value under H0", y = "Count")
```

Flat histogram. Any other shape is a sign the test is misspecified.

4.49.7 3. Assumptions

The claim that p-values are uniform under H0 requires that H0 be *correctly specified* — the distributional assumptions of the test must hold. If the t-test is applied to strongly skewed data at small n, the histogram will not be flat, and the type I error will depart from α.

4.49.8 4. Conduct

4.49.8.1 Type I error rate

```
type_I <- mean(p_h0 < 0.05)
type_I
```

Close to the nominal 0.05.

4.49.8.2 Power under H1 (effect size 0.5, n per group 30)

```
p_h1 <- simulate_pvals(30, delta = 0.5)
power_obs <- mean(p_h1 < 0.05)
power_obs
```

4.49.8.3 Power by effect size and sample size

```
grid <- expand_grid(
  n = c(10, 20, 50, 100),
  delta = seq(0, 1.2, by = 0.2)
)
grid$power <- with(grid, mapply(function(nn, dd) {
  mean(simulate_pvals(nn, dd, B = 1000) < 0.05)
}, n, delta))
```

```
grid |>
  ggplot(aes(delta, power, colour = factor(n))) +
  geom_line(linewidth = 1) +
  geom_point() +
  geom_hline(yintercept = 0.8, linetype = 2, colour = "grey40") +
  labs(x = "Effect size (delta)", y = "Power",
       colour = "n per group")
```

4.49.9 5. Concluding statement

> In 5000 simulated datasets with no true effect, the empirical type I error rate of the two-sample t-test was `signif(type_I,2)` (nominal 0.05), and the p-value histogram was flat as theory requires. With a true effect $\delta = 0.5$ at $n = 30$ per group, the empirical power was `signif(power_obs,2)`. Power grew with both effect size and sample size as expected; 80% power at $\delta = 0.5$ required roughly $n = 65$ per group.

A p-value is a random variable, uniform under H0. The common bar of 0.05 is a convention, not a discovery about nature, and rejecting results above it is

a statement about rates of evidence in a long run of studies — not about the truth of any single study.

4.49.10 Common pitfalls

- Reporting $p = 0.051$ as "not significant" and $p = 0.049$ as "significant" as if the dichotomy were informative.
- Confusing $P(\text{data} \mid \text{H0})$ with $P(\text{H0} \mid \text{data})$.
- Running many tests without adjustment and reporting the smallest p-value.
- Declaring "no effect" when a non-significant p-value comes from an underpowered study.

4.49.11 Further reading

- Wasserstein RL, Lazar NA (2016). *The ASA's Statement on p-Values.*
- Greenland S et al. (2016). *Statistical tests, P values, confidence intervals, and power: a guide to misinterpretations.*

4.49.12 Session info

```
sessionInfo()
```

4.49.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.50 Week 3, Session 3 — Maximum likelihood estimation

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.50.1 Learning objectives

- Write a log-likelihood for a simple model and find its maximum.
- Use Fisher information to obtain an asymptotic standard error and confidence interval.
- Relate the MLE to intuitive estimators (sample mean, sample rate).

4.50.2 Prerequisites

Lab 3.1.

4.50.3 Background

Maximum likelihood is the workhorse of parametric inference. Given a parametric family for the data — normal, Poisson, binomial — the MLE is the parameter value that makes the observed data most probable. Under mild regularity conditions the MLE is consistent (it converges to the truth), asymptotically unbiased, and asymptotically normal with variance given by the inverse of Fisher information.

Fisher information quantifies how sharply the log-likelihood peaks at its maximum. A sharply peaked likelihood gives a precise estimate (small variance); a flat one gives a vague estimate. The standard error of an MLE is the square root of $1/n$ times the inverse of the Fisher information per observation.

In the simple cases we treat here the MLE coincides with the intuitive estimator — the mean for a normal, the sample rate for a Poisson — but the machinery generalises to any likelihood.

4.50.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.50.5 1. Hypothesis

Estimate (i) the mean of a normal and (ii) the rate of a Poisson by maximum likelihood, recover standard errors from Fisher information, and build Wald confidence intervals.

4.50.6 2. Visualise

Normal data with known $\sigma = 2$.

```
n <- 50
mu_true <- 5
sigma <- 2
x <- rnorm(n, mean = mu_true, sd = sigma)
```

```
grid <- tibble(mu = seq(3.5, 6.5, length.out = 300)) |>
  mutate(loglik = sapply(mu, function(m) sum(dnorm(x, m, sigma, log =
  ↪  TRUE))))
ggplot(grid, aes(mu, loglik)) +
  geom_line() +
  geom_vline(xintercept = mean(x), linetype = 2, colour = "firebrick") +
  labs(x = expression(mu), y = "log-likelihood")
```

The likelihood is peaked at the sample mean — that is the MLE.

4.50.7 3. Assumptions

Normal model: iid, constant σ. Poisson model: iid counts, constant rate over a fixed exposure. Fisher information is evaluated *at the MLE* for the Wald standard error.

4.50.8 4. Conduct

4.50.8.1 Normal mean by MLE

```
mle_mu <- mean(x)
# Fisher information for mu in normal with known sigma is n / sigma^2.
I_mu <- n / sigma^2
se_mu <- 1 / sqrt(I_mu)
ci_mu <- mle_mu + c(-1, 1) * qnorm(0.975) * se_mu
tibble(mle = mle_mu, se = se_mu,
       ci_low = ci_mu[1], ci_high = ci_mu[2])
```

4.50.8.2 Normal mean by numeric optimisation (sanity check)

```
nll <- function(mu) -sum(dnorm(x, mu, sigma, log = TRUE))
opt <- optim(0, nll, method = "Brent", lower = 0, upper = 10, hessian =
 ↪  TRUE)
opt$par        # MLE
sqrt(1 / opt$hessian[1, 1])  # SE from observed information
```

4.50.8.3 Poisson rate by MLE

```
y <- rpois(100, lambda = 3.2)
mle_lambda <- mean(y)
# Fisher information for lambda: n / lambda. At MLE use 1/mle_lambda per
 ↪  obs.
I_lambda <- length(y) / mle_lambda
se_lambda <- 1 / sqrt(I_lambda)
ci_lambda <- mle_lambda + c(-1, 1) * qnorm(0.975) * se_lambda
tibble(mle = mle_lambda, se = se_lambda,
       ci_low = ci_lambda[1], ci_high = ci_lambda[2])
```

4.50.8.4 Likelihood surface for a two-parameter model (μ, σ)

```
surf <- expand_grid(
  mu = seq(3.5, 6.5, length.out = 80),
```

```
  s  = seq(1, 3.5, length.out = 80)
) |>
  rowwise() |>
  mutate(ll = sum(dnorm(x, mu, s, log = TRUE))) |>
  ungroup()
```

```
ggplot(surf, aes(mu, s, fill = ll)) +
  geom_tile() +
  geom_contour(aes(z = ll), colour = "white") +
  scale_fill_viridis_c() +
  labs(x = expression(mu), y = expression(sigma), fill = "log-L")
```

4.50.9 5. Concluding statement

> The MLE of the normal mean from $n =$ `n` observations at $\sigma = 2$ was `round(mle_mu,3)` (Fisher-information SE = `round(se_mu,3)`; 95% Wald CI `round(ci_mu[1],2)` to `round(ci_mu[2],2)`), recovering the true $\mu = 5$. The MLE of the Poisson rate from $n = 100$ observations was `round(mle_lambda,3)` (SE `round(se_lambda,3)`; CI `round(ci_lambda[1],2)` to `round(ci_lambda[2],2)`). Observed-information SEs from `optim` agreed with the expected-information calculation.

The MLE is the general-purpose engine behind linear regression, logistic regression, Cox models, and the bulk of parametric inference in this course. The machinery — likelihood, information, SE — is the same in each case.

4.50.10 Common pitfalls

- Reporting a Wald CI when the sample is small. Profile CIs are more accurate in that regime.
- Evaluating Fisher information at a wrong value of the parameter. Use the MLE.
- Forgetting that the MLE is not always unbiased for finite n — only asymptotically.

4.50.11 Further reading

- Casella G, Berger RL. *Statistical Inference*, chapter on point estimation.
- Cox DR. *Principles of Statistical Inference*.

4.50.12 Session info

```
sessionInfo()
```

4.50.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.51 Week 4, Session 4 — Non-parametric tests

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.51.1 Learning objectives

- Choose the right rank-based test for a given design — Wilcoxon signed-rank, Mann-Whitney U, Kruskal-Wallis, sign test.
- Run each in base R and interpret the output.
- State what null hypothesis each test is really addressing (not always "medians are equal").

4.51.2 Prerequisites

Labs 3.4, 4.1.

4.51.3 Background

Non-parametric tests make weaker distributional assumptions than the parametric tests that dominate the preceding labs. They tend to test *exchangeability* of distributions rather than equality of means, and they operate on ranks of the data rather than on the raw values. When the data are heavily skewed, ordinal, or small-sample, the non-parametric alternatives are often the defensible default.

The workhorse set: Wilcoxon signed-rank for paired continuous data; Mann-Whitney U (the `wilcox.test` with `paired=FALSE`) for two independent groups; Kruskal-Wallis for three or more groups; sign test as a cruder paired-data option based on the signs of differences alone.

A common mistake is to read these tests as "comparing medians". They compare *distributions*. Under additional assumptions — notably that the two groups' distributions differ only by a location shift — the Mann-Whitney U test does estimate a location shift, but that is a stronger assumption than the test itself requires.

4.51.4 Setup

```
library(tidyverse)
set.seed(42)
```

```
theme_set(theme_minimal(base_size = 12))
```

4.51.5 1. Hypothesis

Three scenarios:

A. Paired: pre/post in 25 patients, skewed outcome. H0: no shift. B. Independent two-sample: outcome in two arms, skewed. H0: same distribution. C. Three-group comparison: outcome across three centres. H0: same distribution across all three.

4.51.6 2. Visualise

```
# Scenario A: paired, lognormal-ish
n_a <- 25
pre <- exp(rnorm(n_a, 3, 0.3))
post <- pre * exp(rnorm(n_a, -0.2, 0.3))
df_a <- tibble(id = seq_len(n_a), pre, post,
               delta = post - pre)

# Scenario B: independent two groups, lognormal
n_b <- 30
grp <- rep(c("A", "B"), each = n_b)
y_b <- c(exp(rnorm(n_b, 3, 0.5)),
         exp(rnorm(n_b, 3.4, 0.5)))
df_b <- tibble(grp, y = y_b)

# Scenario C: three groups
n_c <- 25
centre <- rep(c("X", "Y", "Z"), each = n_c)
y_c <- c(exp(rnorm(n_c, 3.0, 0.4)),
         exp(rnorm(n_c, 3.2, 0.4)),
         exp(rnorm(n_c, 3.5, 0.4)))
df_c <- tibble(centre, y = y_c)
```

```
df_b |>
  ggplot(aes(grp, y, fill = grp)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "Outcome") +
  theme(legend.position = "none")
```

4.51.7 3. Assumptions

Wilcoxon signed-rank: paired observations; differences symmetric around the null. Mann-Whitney U: independent observations within and between groups;

if you want a location-shift interpretation, the two distributions should have the same shape. Kruskal-Wallis: independent observations; same-shape assumption extends to all k groups.

4.51.8 4. Conduct

```
# A: Wilcoxon signed-rank
wt_a <- wilcox.test(df_a$post, df_a$pre, paired = TRUE,
                    conf.int = TRUE)
# Sign test: a simple binomial test on positive differences
signs <- sum(df_a$delta > 0)
sign_test <- binom.test(signs, n_a, p = 0.5)

# B: Mann-Whitney U
wt_b <- wilcox.test(y ~ grp, data = df_b, conf.int = TRUE)

# C: Kruskal-Wallis
kw_c <- kruskal.test(y ~ centre, data = df_c)

list(wilcox_paired = wt_a,
     sign_test = sign_test,
     mann_whitney = wt_b,
     kruskal_wallis = kw_c)
```

Follow-up pairwise comparisons with Holm correction for C:

```
pairwise.wilcox.test(df_c$y, df_c$centre, p.adjust.method = "holm")
```

4.51.9 5. Concluding statement

> **Paired comparison (A).** 25 patients, median change = `round(median(df_a$delta),2)`; Wilcoxon signed-rank p = `signif(wt_a$p.value,2)`, 95% CI on the Hodges-Lehmann estimate `round(wt_a$conf.int[1],2)` to `round(wt_a$conf.int[2],2)`. Sign test p = `signif(sign_test$p.value,2)`.
>
> **Two-group (B).** Mann-Whitney p = `signif(wt_b$p.value,2)`, 95% CI `round(wt_b$conf.int[1],2)` to `round(wt_b$conf.int[2],2)`.
>
> **Three-group (C).** Kruskal-Wallis χ^2 = `round(kw_c$statistic,2)`, df = `kw_c$parameter`, p = `signif(kw_c$p.value,2)`. Pairwise Wilcoxon tests with Holm correction indicated the signal lay in the X vs Z contrast.

Non-parametric tests are not free lunches. They trade a modest amount of power (about 5% vs a t-test when the t-test's assumptions hold) for robustness to distributional misspecification.

4.51.10 Common pitfalls

- Reading a Wilcoxon as a test of medians. It is a test of distributional shift.
- Using a rank test when the data are fine on the raw scale, and sacrificing power unnecessarily.
- Forgetting to adjust for multiplicity after a Kruskal-Wallis.

4.51.11 Further reading

- Hollander M, Wolfe DA. *Nonparametric Statistical Methods*.
- Conover WJ. *Practical Nonparametric Statistics*.

4.51.12 Session info

```
sessionInfo()
```

4.51.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.52 Week 3, Session 4 — One-sample t-test and one-proportion test

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.52.1 Learning objectives

- State a one-sample null and alternative hypothesis for a continuous outcome and for a proportion.
- Run `t.test(mu=)` and `prop.test()` / `binom.test()` and interpret the output.
- Apply the full five-step inference template in one working pass.

4.52.2 Prerequisites

Labs 2.5, 3.3.

4.52.3 Background

The one-sample t-test asks whether the mean of a continuous outcome differs from a pre-specified value μ_0. It is the inference analogue of Table 1: a single-column calculation asking whether the sample is consistent with a named popu-

lation mean. The test statistic is $(\bar{x} - \mu_0) \,/\, (s/\sqrt{n})$, compared to a t distribution on $n - 1$ degrees of freedom.

The one-proportion test does the same for a binary outcome. When the sample is large we use `prop.test()`, which is a Wald-type test based on the normal approximation to the binomial; when the sample is small, `binom.test()` gives an exact test based on the binomial PMF. Both return a p-value, a point estimate, and a CI.

These are the simplest tests in the course, but the five-step template — hypothesis, visualise, assumptions, conduct, conclude — is the same template used in every later inference lab. Getting the template into muscle memory here pays dividends throughout the rest of the course.

4.52.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.52.5 1. Hypothesis

Scenario A (continuous): fasting blood glucose in a random sample of `n=40` patients from a clinic. Reference mean $\mu_0 = 95$ mg/dL in a healthy population. Two-sided test.

- H0: $\mu = 95$
- H1: $\mu \neq 95$
- $\alpha = 0.05$

Scenario B (proportion): of 120 patients offered a new screening programme, $k = 78$ accepted. Reference acceptance rate $\pi_0 = 0.60$.

- H0: $\pi = 0.60$
- H1: $\pi \neq 0.60$
- $\alpha = 0.05$

4.52.6 2. Visualise

```
n <- 40
glucose <- rnorm(n, mean = 100, sd = 12)
```

```
tibble(glucose) |>
  ggplot(aes(glucose)) +
  geom_histogram(bins = 15, fill = "grey60", colour = "white") +
  geom_vline(xintercept = 95, linetype = 2) +
  geom_vline(xintercept = mean(glucose), colour = "firebrick") +
```

```
  labs(x = "Fasting glucose (mg/dL)", y = "Count")
```

```
accept <- 78
total  <- 120
tibble(state = c("accepted", "declined"),
       count = c(accept, total - accept))
```

4.52.7 3. Assumptions

t-test: approximate normality of the sample mean, which — by the central limit theorem — holds for $n = 40$ unless the underlying distribution is strongly skewed. Check with a Q-Q plot.

```
tibble(glucose) |>
  ggplot(aes(sample = glucose)) +
  stat_qq() + stat_qq_line(colour = "firebrick") +
  labs(x = "Theoretical", y = "Sample")
```

Proportion test: independent binary trials, same success probability. For `prop.test` the large-sample normal approximation needs $np_0 \geq 5$ and $n(1 - p_0) \geq 5$; both are satisfied.

4.52.8 4. Conduct

```
tt <- t.test(glucose, mu = 95)
tt
```

```
pt_large <- prop.test(x = accept, n = total, p = 0.60)
pt_exact <- binom.test(x = accept, n = total, p = 0.60)
list(prop_test = pt_large, binom_test = pt_exact)
```

4.52.9 5. Concluding statement

> **Scenario A.** In a sample of `n` patients, mean fasting glucose was `round(mean(glucose),1)` mg/dL (SD `round(sd(glucose),1)`). The one-sample t-test against $\mu_0 = 95$ gave t(`tt$parameter`) = `round(tt$statistic,2)`, p = `signif(tt$p.value,2)`, mean difference `round(mean(glucose)-95,1)` (95% CI `round(tt$conf.int[1]-95,1)` to `round(tt$conf.int[2]-95,1)`).
>
> **Scenario B.** Acceptance was `accept` of `total` (`round(100*accept/total,1)`%); the exact one-proportion test against $\pi_0 = 0.60$ gave p = `signif(pt_exact$p.value,2)`, 95% CI `round(pt_exact$conf.int[1],2)` to `round(pt_exact$conf.int[2],2)`.

The two scenarios share a template. The glossary changes — mean and SD for continuous, count and proportion for binary — but the five steps are the same.

4.52.10 Common pitfalls

- Running a one-sided test to "rescue" a borderline two-sided p-value.
- Reporting the p-value without the mean difference and its CI.
- Using `prop.test` when the expected count of either outcome is very small; use `binom.test` there.

4.52.11 Further reading

- Rosner B. *Fundamentals of Biostatistics*, chapter on one-sample inference.
- Altman DG, *Confidence intervals rather than p values*, BMJ 1986.

4.52.12 Session info

```
sessionInfo()
```

4.52.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.53 Week 4, Session 3 — Pearson and Spearman correlation

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.53.1 Learning objectives

- Compute Pearson and Spearman correlations with `cor.test()` and interpret the coefficient and its CI.
- Distinguish linear association from monotonic association.
- Use visualisation to diagnose situations where a correlation coefficient misleads (Anscombe, DatasauRus).

4.53.2 Prerequisites

Basic inferential statistics.

4.53.3 Background

Pearson's r measures linear association. It is the standardised covariance of two variables; it is sensitive to outliers and assumes a roughly linear, bivariate-normal relationship. Spearman's ρ measures monotonic association. It is Pearson's r applied to the ranks of the variables; it is robust to outliers and to non-linear monotonic transformations.

Neither coefficient captures non-linear, non-monotonic relationships well. Anscombe's quartet and the DatasauRus dozen are the canonical demonstrations that a coefficient without a plot can lie — datasets engineered to have identical Pearson r but radically different shapes. The conclusion is not to abandon correlations but never to report one without first plotting the data.

A correlation coefficient is not a causal quantity. Two variables can be perfectly correlated without one causing the other — a lurking variable, a common cause, a coincidence of sample selection. The word "correlation" is often used loosely in the press to mean "weak causation"; in this course it means exactly what `cor.test()` returns.

4.53.4 Setup

```
library(tidyverse)
library(datasauRus)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.53.5 1. Hypothesis

Two scenarios:

1. A linear relationship between two simulated variables.
2. A monotonic but non-linear relationship where Pearson misleads.

H0 in each case: $\rho = 0$. H1: $\rho \neq 0$.

4.53.6 2. Visualise

```
n <- 200
x <- rnorm(n)
y_linear   <- 0.6 * x + rnorm(n, 0, 0.8)
y_monotone <- exp(x) + rnorm(n, 0, 0.5)
df <- tibble(x, y_linear, y_monotone) |>
  pivot_longer(starts_with("y_"), names_to = "type", values_to = "y")
```

```
df |>
  ggplot(aes(x, y)) +
```

```
  geom_point(alpha = 0.5) +
  facet_wrap(~ type, scales = "free") +
  geom_smooth(method = "lm", se = FALSE, colour = "firebrick", linewidth
  ↪  = 0.5)
```

4.53.7 3. Assumptions

Pearson assumes linearity and roughly bivariate-normal data. Spearman assumes monotonicity. Neither handles non-monotonic relationships. For both, the observations must be independent; both are sensitive to extreme sample-size-induced significance when n is huge.

4.53.8 4. Conduct

```
p1_pearson  <- cor.test(x, y_linear,   method = "pearson")
p1_spearman <- cor.test(x, y_linear,   method = "spearman", exact = FALSE)
p2_pearson  <- cor.test(x, y_monotone, method = "pearson")
p2_spearman <- cor.test(x, y_monotone, method = "spearman", exact =
↪  FALSE)

tibble(
  relationship = c("linear", "linear", "monotone", "monotone"),
  method = c("Pearson", "Spearman", "Pearson", "Spearman"),
  r_or_rho = c(p1_pearson$estimate, p1_spearman$estimate,
               p2_pearson$estimate, p2_spearman$estimate),
  p = c(p1_pearson$p.value, p1_spearman$p.value,
        p2_pearson$p.value, p2_spearman$p.value)
)
```

The monotonic non-linear relationship gives a much higher Spearman than Pearson — the rank correlation sees the perfect monotonicity.

4.53.8.1 The DatasauRus demonstration

```
ds_summary <- datasaurus_dozen |>
  group_by(dataset) |>
  summarise(mean_x = mean(x),
            mean_y = mean(y),
            sd_x = sd(x),
            sd_y = sd(y),
            r = cor(x, y), .groups = "drop")
head(ds_summary)
```

```
datasaurus_dozen |>
```

```
  ggplot(aes(x, y)) +
  geom_point(alpha = 0.4, size = 0.8) +
  facet_wrap(~ dataset) +
  labs(x = NULL, y = NULL)
```

Thirteen datasets; all have mean $x \approx 54$, mean $y \approx 47$, SDs ≈ 17 and 26, and Pearson $r \approx -0.06$. The shapes are wildly different.

4.53.9 5. Concluding statement

> In the linear case, Pearson $r =$ `round(p1_pearson$estimate,2)` ($p =$ `signif(p1_pearson$p.value,2)`) and Spearman $\rho =$ `round(p1_spearman$estimate,2)`; the two agreed. In the monotonic non-linear case, Pearson $r =$ `round(p2_pearson$estimate,2)` was substantially smaller than Spearman $\rho =$ `round(p2_spearman$estimate,2)`, revealing that the association was strong but not linear. The DatasauRus dozen reproduced identical summary statistics across 13 radically different shapes, reinforcing the rule: never report a correlation without the scatter plot.

Correlation is a single number. Its honest use requires a picture alongside.

4.53.10 Common pitfalls

- Reporting a Pearson r on data with an obvious curvilinear shape.
- Using r to compare groups with different sample sizes; small r can be "significant" at huge n.
- Inferring causation from a correlation.
- Computing correlations on data that include outliers without a robustness check.

4.53.11 Further reading

- Anscombe FJ (1973). *Graphs in Statistical Analysis.*
- Matejka J, Fitzmaurice G (2017). *Same stats, different graphs: generating datasets with varied appearance and identical statistics.*

4.53.12 Session info

```
sessionInfo()
```

4.53.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.54 Week 3, Session 1 — Populations, samples, and the central limit theorem

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.54.1 Learning objectives

- Distinguish population parameters from sample estimates and their sampling distributions.
- Decompose expected error into bias squared, variance, and irreducible noise.
- Demonstrate the central limit theorem by simulation on a strongly skewed population.

4.54.2 Prerequisites

Weeks 1 and 2.

4.54.3 Background

A population is the set of units we wish to learn about. A sample is the set of units we actually measure. The population has a fixed parameter — mean μ, proportion π, whatever — and the sample has an estimate — $\bar{x}$, $\hat{p}$ — which is a random variable because the sample was drawn at random. Every estimate has a *sampling distribution*: the distribution of its value across hypothetical repetitions of the sampling process. Standard errors, confidence intervals, and p-values are all properties of sampling distributions.

Error decomposes into bias and variance. The bias of an estimator is the systematic offset of its sampling distribution's mean from the parameter. The variance is the spread of the sampling distribution. Mean squared error — bias squared plus variance — is the natural scalar summary of how wrong an estimator is on average. A good estimator is not one with zero bias; it is one with low MSE.

The central limit theorem is the reason the normal distribution is ubiquitous. No matter how skewed the population (within mild conditions on variance), the sampling distribution of the mean approaches normal as n grows. The lab makes that convergence visible.

4.54.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.54.5 1. Hypothesis

Claim: the sampling distribution of the sample mean from an exponential (strongly right-skewed) population approaches a normal distribution as n increases, with shrinking standard error.

4.54.6 2. Visualise

Draw from an exponential population and show the distribution of individual observations.

```
rate <- 1
pop_mean <- 1 / rate
pop_sd   <- 1 / rate

pop_sample <- tibble(x = rexp(2000, rate = rate))
```

```
pop_sample |>
  ggplot(aes(x)) +
  geom_histogram(bins = 40, fill = "grey60", colour = "white") +
  geom_vline(xintercept = pop_mean, linetype = 2) +
  labs(x = "x", y = "Count")
```

4.54.7 3. Assumptions

CLT assumes iid sampling and finite variance. Exponential qualifies. The speed of convergence is set by how skewed the parent is — the more skewed, the larger the n needed for normality.

```
# Bias-variance sanity check on the sample mean as an estimator of the
# population mean, across sample sizes.
summarise_means <- function(n, B = 2000) {
  xbars <- replicate(B, mean(rexp(n, rate = rate)))
  tibble(n = n,
         bias = mean(xbars) - pop_mean,
         variance = var(xbars),
         mse = mean((xbars - pop_mean)^2),
         se = sd(xbars))
}
bv <- map_dfr(c(5, 10, 30, 100, 500), summarise_means)
bv
```

4.54.8 4. Conduct

Plot the sampling distribution of the mean across growing n.

```
sim_means <- function(n, B = 4000) {
  replicate(B, mean(rexp(n, rate = rate)))
}

means_df <- bind_rows(
  tibble(n = "n = 2",   mean = sim_means(2)),
  tibble(n = "n = 5",   mean = sim_means(5)),
  tibble(n = "n = 30",  mean = sim_means(30)),
  tibble(n = "n = 100", mean = sim_means(100))
) |>
  mutate(n = factor(n, levels = c("n = 2", "n = 5", "n = 30", "n = 100")))
```

```
means_df |>
  ggplot(aes(mean)) +
  geom_histogram(bins = 40, fill = "grey60", colour = "white") +
  facet_wrap(~ n, scales = "free") +
  geom_vline(xintercept = pop_mean, linetype = 2) +
  labs(x = "Sample mean", y = "Count")
```

The histograms widen (more spread with small n) and are strongly right-skewed at $n = 2$. By $n = 30$ they are close to symmetric.

```
# Check SE shrinking like pop_sd / sqrt(n).
bv |> mutate(expected_se = pop_sd / sqrt(n),
             ratio = se / expected_se)
```

4.54.9 5. Concluding statement

> Sampling from an exponential population with mean 1 and SD 1, the sample mean was unbiased across every n simulated ($|\text{bias}| < 0.02$ in all cases). Its standard error shrank as $1/\sqrt{n}$ — ratio of observed to expected SE near 1 throughout — and its sampling distribution was visibly skewed at $n = 2$ but approximately normal by $n = 30$. The central limit theorem is fully operational by modest sample sizes even for a strongly skewed parent.

"$n \geq 30$" is a rule of thumb, not a theorem. For very skewed parents, a larger n is needed; for symmetric parents, a smaller n may suffice. Always check by simulation when in doubt.

4.54.10 Common pitfalls

- Believing the CLT applies to the *data*, not to the *mean of the data*. Individual observations stay whatever they were.
- Invoking the CLT to justify normality of a small skewed sample. The CLT is about sampling distributions.

- Confusing standard error (a property of the sampling distribution) with standard deviation (a property of the data).

4.54.11 Further reading

- Efron B, Hastie T. *Computer Age Statistical Inference*, chapter 1.
- Wasserman L. *All of Statistics*, section on limit theorems.

4.54.12 Session info

```
sessionInfo()
```

4.54.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.55 Week 4, Session 2 — Two proportions, chi-square, risk and odds

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.55.1 Learning objectives

- Compare two proportions with `prop.test()` and `fisher.test()` and say when each is preferred.
- Compute risk ratio, odds ratio, and risk difference with confidence intervals.
- Run a chi-square goodness-of-fit test against a named distribution.

4.55.2 Prerequisites

Labs 2.4, 3.4.

4.55.3 Background

Two-proportion tests ask whether the rate of an event differs between two groups. In a 2x2 table, two exposure groups (say, treatment and control) each produce a fraction of events. The chi-square test compares the observed counts to those expected under independence; Fisher's exact test computes the probability of a table as extreme or more extreme directly from the hypergeometric distribution and is preferred when expected cell counts are small.

The effect can be expressed in several ways. The risk difference is the difference of the two proportions. The risk ratio is their ratio — the event rate in the treated group, relative to control. The odds ratio is the ratio of the odds. In a case-control study the risk ratio is not directly estimable; the odds ratio is. In a cohort study both are estimable and the risk ratio is usually more intuitive.

Goodness-of-fit tests compare a vector of observed counts to a vector of expected proportions under a named distribution — whether a die is fair, whether a set of genotype frequencies matches Hardy-Weinberg, whether arrivals over an hour are Poisson. The test statistic is the same chi-square; the interpretation is "is the distribution consistent with a named model?"

4.55.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.55.5 1. Hypothesis

Two-by-two: `treatment` vs `control`, outcome `event` vs `none`.

- H0: event rate is independent of treatment.
- H1: rates differ.
- $\alpha = 0.05$.

4.55.6 2. Visualise

Simulate: 200 per arm, event rate 0.25 in control, 0.15 in treatment.

```
n_per <- 200
sim <- tibble(
  arm = rep(c("control", "treatment"), each = n_per),
  event = c(rbinom(n_per, 1, 0.25),
            rbinom(n_per, 1, 0.15))
)
tab <- table(arm = sim$arm, event = sim$event)
tab
```

```
as_tibble(tab) |>
  mutate(arm = factor(arm, levels = c("control", "treatment")),
         event = recode(event, `0` = "no event", `1` = "event")) |>
  ggplot(aes(arm, n, fill = event)) +
  geom_col(position = "fill", alpha = 0.7) +
  labs(x = NULL, y = "Proportion", fill = NULL)
```

4.55.7 3. Assumptions

Chi-square requires expected cell counts ≥ 5. Fisher has no such constraint and is exact at any size. Random assignment or representative sampling is assumed. Independence of observations is essential.

```
chisq.test(tab)$expected
```

All expected counts large; chi-square is appropriate.

4.55.8 4. Conduct

```
ct <- chisq.test(tab, correct = FALSE)
ft <- fisher.test(tab)
pt <- prop.test(x = tab[, "1"], n = rowSums(tab), correct = FALSE)
list(chisq = ct, fisher = ft, prop = pt)
```

4.55.8.1 Effect sizes: RD, RR, OR

```
p_ctrl <- tab["control",   "1"] / sum(tab["control", ])
p_trt  <- tab["treatment", "1"] / sum(tab["treatment", ])

rd <- p_trt - p_ctrl

rr <- p_trt / p_ctrl
# Log-based Wald CI for RR
se_log_rr <- sqrt(
  (1 - p_trt)  / (p_trt  * sum(tab["treatment", ])) +
  (1 - p_ctrl) / (p_ctrl * sum(tab["control",   ]))
)
ci_rr <- exp(log(rr) + c(-1, 1) * qnorm(0.975) * se_log_rr)

or <- (tab["treatment", "1"] * tab["control", "0"]) /
      (tab["treatment", "0"] * tab["control", "1"])
se_log_or <- sqrt(sum(1 / tab))
ci_or <- exp(log(or) + c(-1, 1) * qnorm(0.975) * se_log_or)

tibble(effect = c("RD", "RR", "OR"),
       est = c(rd, rr, or),
       ci_low = c(pt$conf.int[1], ci_rr[1], ci_or[1]),
       ci_high = c(pt$conf.int[2], ci_rr[2], ci_or[2]))
```

4.55.8.2 Goodness-of-fit example

Test whether 600 simulated genotype counts match Hardy-Weinberg proportions at allele frequency 0.3.

```
obs <- c(AA = 260, Aa = 290, aa = 50)
pAA <- 0.7^2; paa <- 0.3^2; pAa <- 2 * 0.7 * 0.3
exp_probs <- c(AA = pAA, Aa = pAa, aa = paa)
gof <- chisq.test(obs, p = exp_probs)
gof
```

4.55.9 5. Concluding statement

> Event rates were `round(p_ctrl,3)` in control and `round(p_trt,3)` in treatment (RD `round(rd,3)`; RR `round(rr,2)`, 95% CI `round(ci_rr[1],2)` to `round(ci_rr[2],2)`; OR `round(or,2)`, 95% CI `round(ci_or[1],2)` to `round(ci_or[2],2)`). Chi-square test χ^2 = `round(ct$statistic,2)`, df = `ct$parameter`, p = `signif(ct$p.value,2)`. The Hardy-Weinberg goodness-of-fit test on simulated genotypes gave χ^2 = `round(gof$statistic,2)` (df = `gof$parameter`, p = `signif(gof$p.value,2)`).

Report RR or OR alongside the p-value. In a cohort design prefer RR, which is the quantity clinicians translate directly to practice.

4.55.10 Common pitfalls

- Reporting an OR when a RR is meaningful and more interpretable.
- Using the continuity correction (the default in `prop.test` and `chisq.test`) by habit; it is conservative.
- Running a chi-square when expected counts are small; use Fisher.
- Treating independent observations as if repeated measures weren't.

4.55.11 Further reading

- Altman DG. *Practical Statistics for Medical Research*, chapter 10–11.
- Agresti A. *Categorical Data Analysis*.

4.55.12 Session info

```
sessionInfo()
```

4.55.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

4.56 Week 4, Session 1 — Two-sample and paired t-tests

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

4.56.1 Learning objectives

- Choose between Student's and Welch's t-test, and between paired and independent-samples tests.
- Compute Cohen's d and Hedges' g as effect sizes.
- Report a two-sample comparison with point estimate, CI, and effect size.

4.56.2 Prerequisites

Lab 3.4.

4.56.3 Background

The two-sample t-test compares the means of two independent groups. Student's version assumes equal variances across the groups; Welch's does not. In most applied settings Welch's is the safer default — it behaves nearly as well as Student's when variances are actually equal and much better when they are not. The `t.test()` default in R is Welch.

When the same subjects are measured twice, the samples are paired and the two measurements are correlated. Ignoring the pairing inflates the standard error and lowers power. Pairing is handled by computing the within-subject difference and running a one-sample t-test on it.

Effect sizes remove units from the comparison. Cohen's d is the standardised mean difference: the mean difference divided by the pooled standard deviation. Hedges' g applies a small-sample correction. Both are essential alongside a p-value — a large d at a small n will often produce a non-significant p, but the d is still a warning that a larger study might find a real effect.

4.56.4 Setup

```
library(tidyverse)
library(effectsize)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

4.56.5 1. Hypothesis

Scenario A (independent): mean systolic blood pressure in treatment vs placebo.

- H0: μ_trt = μ_pla, H1: μ_trt $\neq$ μ_pla, $\alpha = 0.05$.

Scenario B (paired): SBP before and after a four-week intervention in the same 30 patients.

- H0: mean within-subject change = 0, H1: $\neq 0$.

4.56.6 2. Visualise

```
n <- 40
trt <- rnorm(n, mean = 135, sd = 12)
pla <- rnorm(n, mean = 142, sd = 12)
df_ind <- tibble(
  arm = rep(c("placebo", "treatment"), each = n),
  sbp = c(pla, trt)
)
```

```
df_ind |>
  ggplot(aes(arm, sbp, fill = arm)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "SBP (mmHg)") +
  theme(legend.position = "none")
```

```
n2 <- 30
pre  <- rnorm(n2, 148, 10)
post <- pre - rnorm(n2, 5, 7)  # true mean drop of 5
df_pair <- tibble(id = seq_len(n2), pre = pre, post = post) |>
  mutate(delta = post - pre)
```

```
df_pair |>
  pivot_longer(c(pre, post), names_to = "time", values_to = "sbp") |>
  mutate(time = factor(time, levels = c("pre", "post"))) |>
  ggplot(aes(time, sbp, group = id)) +
  geom_line(alpha = 0.4) +
  geom_point(alpha = 0.6) +
  labs(x = NULL, y = "SBP (mmHg)")
```

4.56.7 3. Assumptions

Independent t: approximate normality of the sample means; Welch's does not require equal variances. Paired t: approximate normality of the within-subject differences.

```
df_pair |>
  ggplot(aes(sample = delta)) +
  stat_qq() + stat_qq_line(colour = "firebrick") +
  labs(x = "Theoretical", y = "Sample (pair differences)")
```

4.56.8 4. Conduct

4.56.8.1 Two-sample (Welch)

```
tt_welch <- t.test(sbp ~ arm, data = df_ind, var.equal = FALSE)
tt_welch
```

4.56.8.2 Two-sample Student (for comparison)

```
tt_student <- t.test(sbp ~ arm, data = df_ind, var.equal = TRUE)
tt_student
```

4.56.8.3 Paired

```
tt_paired <- t.test(df_pair$post, df_pair$pre, paired = TRUE)
tt_paired
```

4.56.8.4 Effect size

```
d_ind    <- cohens_d(sbp ~ arm, data = df_ind)
g_ind    <- hedges_g(sbp ~ arm, data = df_ind)
d_paired <- cohens_d(df_pair$post, df_pair$pre, paired = TRUE)
list(d_ind = d_ind, g_ind = g_ind, d_paired = d_paired)
```

4.56.9 5. Concluding statement

Independent comparison. Mean SBP was `round(mean(trt),1)` mmHg (SD `round(sd(trt),1)`) in the treatment arm and `round(mean(pla),1)` (SD `round(sd(pla),1)`) in placebo. A Welch's t-test gave t(`round(tt_welch$parameter,1)`) = `round(tt_welch$statistic,2)`, p = `signif(tt_welch$p.value,2)`, mean difference `round(diff(rev(tt_welch$estimate)),1)` mmHg (95% CI `round(-tt_welch$conf.int[2],1)` to `round(-tt_welch$conf.int[1],1)`). Cohen's d = `round(d_ind$Cohens_d,2)` (Hedges' g = `round(g_ind$Hedges_g,2)`), a small-to-medium effect.

Paired comparison. The within-subject change (post − pre) had mean `round(mean(df_pair$delta),1)` mmHg (SD `round(sd(df_pair$delta),1)`); paired t-test t(`tt_paired$parameter`) = `round(tt_paired$statistic,2)`, p = `signif(tt_paired$p.value,2)`, 95% CI `round(tt_paired$conf.int[1],1)` to `round(tt_paired$conf.int[2],1)`.

The effect size is the number that generalises. A p-value tells you whether to reject; d and its CI tell you how much.

4.56.10 Common pitfalls

- Running an independent-samples t on paired data and losing power.
- Reporting Student's t in the era of Welch; there is rarely a good reason.
- Reporting p without d (or an equivalent effect size).
- Confusing effect size with statistical significance.

4.56.11 Further reading

- Delacre M et al. (2017). *Why psychologists should by default use Welch's t-test.*
- Cohen J. *Statistical Power Analysis for the Behavioral Sciences.*

4.56.12 Session info

```
sessionInfo()
```

4.56.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 5

Sample Size & Power

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Sample-size and power calculations for the designs covered in the rest of the book — t-tests, ANOVA, regression, survival, diagnostic accuracy, equivalence and non-inferiority, sequential designs. Each method page includes the exact `pwr` or simulation call.

This chapter contains **30 method pages** and **2 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

5.1 Method pages

Method	Source slug
Effect Size: Cohen's d	`effect-size-cohens-d`
Effect Size: Cohen's h	`effect-size-cohens-h`
Effect Size: Eta-Squared	`effect-size-eta-squared`
Minimum Detectable Effect	`minimum-detectable-effect`
Post-Hoc Power: A Controversy	`post-hoc-power-controversy`
Power Analysis: Introduction	`power-analysis-introduction`
Power for Agreement (Kappa)	`power-agreement-kappa`
Power for Bland-Altman Studies	`power-bland-altman`
Power for Chi-Squared Tests	`power-chi-squared`
Power for Cluster-RCT	`power-cluster-rct`
Power for Correlation Tests	`power-correlation`
Power for Cox Regression	`power-cox-regression`
Power for Crossover Trials	`power-crossover`
Power for Diagnostic Accuracy	`power-diagnostic-accuracy`
Power for Equivalence (TOST)	`power-equivalence-tost`

Method	Source slug
Power for ICC	`power-icc`
Power for Linear Regression	`power-linear-regression`
Power for Logistic Regression	`power-logistic-regression`
Power for McNemar's Test	`power-mcnemar`
Power for Non-Inferiority Trials	`power-non-inferiority`
Power for One-Proportion Test	`power-one-proportion`
Power for One-Sample t-Test	`power-one-sample-t`
Power for One-Way ANOVA	`power-anova`
Power for Paired t-Test	`power-paired-t`
Power for Repeated-Measures ANOVA	`power-repeated-measures`
Power for Stepped-Wedge Trials	`power-stepped-wedge`
Power for the Log-Rank Test	`power-logrank-test`
Power for Two-Proportion Test	`power-two-proportions`
Sample Size for a Two-Sample t-Test	`power-two-sample-t`
Sample Size Sensitivity Analysis	`sensitivity-analysis-sample-size`

5.2 Labs

Lab
Sample size, power, and Quarto reporting
Power: closed-form and simulation

5.3 Effect Size: Cohen's d

5.3.1 Introduction

Cohen's d is the ratio of a mean difference to a standard deviation. It standardises effects for cross-study comparison and power analysis. Although simple in idea, several variants differ in which SD is used as the denominator.

5.3.2 Prerequisites

Means and standard deviations, two-sample t-test.

5.3.3 Theory

Classical definitions:

- **Cohen's** d (two independent groups): $d = (\bar{x}_1 - \bar{x}_2)/s_{\text{pooled}}$, where $s^2_{\text{pooled}} = [(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2]/(n_1 + n_2 - 2)$.
- **Hedges'** g: Cohen's d multiplied by a small-sample bias correction factor.

- **Glass's Δ**: d using only the control group's SD (for unequal variances).
- **Cohen's d_z** (paired): mean paired difference / SD of differences.
- **Cohen's $d_{\mathbf{rm}}$** (repeated measures): uses a specific variance of the mean difference.

Cohen's thresholds: 0.20 / 0.50 / 0.80 small / medium / large. These are starting points; clinical context should override conventions.

5.3.4 Assumptions

- Approximately normal data (for the SD to be meaningful).
- Equal variances for classical Cohen's d; use Glass or Welch-style for unequal.

5.3.5 R Implementation

```r
library(effectsize)
set.seed(2026)

g1 <- rnorm(50, 50, 10)
g2 <- rnorm(50, 55, 10)

cohens_d(g1, g2)
hedges_g(g1, g2)
glass_delta(g1, g2)

# Paired
pre  <- rnorm(30, 50, 10)
post <- pre + rnorm(30, -5, 5)
cohens_d(pre, post, paired = TRUE)

# Unequal variances -> Glass' Delta
g1_big <- rnorm(50, 50, 5)
g2_big <- rnorm(50, 55, 15)
cohens_d(g1_big, g2_big)
glass_delta(g1_big, g2_big)
```

5.3.6 Output & Results

For two-sample example: Cohen's $d \approx -0.47$, Hedges' $g \approx -0.47$, Glass's $\Delta \approx -0.50$.

5.3.7 Interpretation

"The intervention reduced anxiety by 5.1 points (SE 2.0, Cohen's $d = 0.51$, 95 % CI 0.11 to 0.90), a medium-sized effect."

5.3.8 Practical Tips

- Use Hedges' g in meta-analysis; it's the small-sample-corrected version that combines across studies fairly.
- Paired designs should use d_z, not two-sample d; the two are not interchangeable.
- Report both the standardised effect and the raw effect in units that matter clinically.
- Cohen's thresholds vary across fields; cite a field-appropriate reference.
- Confidence intervals for d (e.g., from `effectsize`) are asymmetric; do not assume symmetric.

5.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.3.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.4 Effect Size: Cohen's h

5.4.1 Introduction

Cohen's h is the standard standardised effect size for differences between two proportions. Unlike the raw risk difference (whose interpretation depends on where on the 0-to-1 scale the comparison sits) or the odds ratio (whose multiplicative scale obscures the absolute magnitude), Cohen's h uses an arcsine-square-root variance-stabilising transformation that makes the effect size approximately scale-free and directly suitable as input to Normal-approximation power calculations. It is the natural building block of two-proportion sample-size formulas in the `pwr` package and equivalent tools, and it is the recommended effect-size summary in published power-analysis protocols.

5.4.2 Prerequisites

A working understanding of binomial proportions, the arcsine variance-stabilising transformation, and the role of standardised effect sizes in sample-size and power calculations.

5.4.3 Theory

For two proportions $p_1, p_2 \in [0, 1]$, Cohen's h is

$$h = 2 \arcsin \sqrt{p_1} - 2 \arcsin \sqrt{p_2}.$$

The arcsine-square-root transformation stabilises the variance of a sample proportion at approximately $1/n$, independent of the proportion's value. As a result, h is dimensionless, scale-free, and directly comparable across baseline rates. Cohen's conventional thresholds are 0.20 (small), 0.50 (medium), and 0.80 (large). The mapping between raw proportion difference and h is non-linear: the same $|p_1 - p_2|$ corresponds to very different h values depending on where on the unit interval the comparison sits, with extreme proportions (near 0 or 1) yielding larger h for the same absolute difference than mid-range comparisons.

5.4.4 Assumptions

The two proportions arise from independent Bernoulli outcomes; the standardisation is most useful for moderate sample sizes where the Normal approximation to the binomial holds.

5.4.5 R Implementation

```r
library(pwr)

ES.h(p1 = 0.10, p2 = 0.20)
ES.h(p1 = 0.50, p2 = 0.60)
ES.h(p1 = 0.80, p2 = 0.90)

pwr.2p.test(h = ES.h(0.10, 0.20), sig.level = 0.05, power = 0.80)
pwr.2p.test(h = ES.h(0.40, 0.50), sig.level = 0.05, power = 0.80)
```

5.4.6 Output & Results

The three example pairs all involve a 10 percentage-point raw difference, but produce very different Cohen's h values: $h = -0.28$ for 10-vs-20 %, $h = -0.20$ for 50-vs-60 %, and $h = -0.29$ for 80-vs-90 %. The corresponding required sample sizes per arm at 80 % power are roughly 200, 388, and 186 — illustrating that the same absolute difference is most expensive to detect near the middle of the proportion range and cheapest at the extremes.

5.4.7 Interpretation

A reporting sentence: "Assuming an intervention increases response from 50 % to 60 % (Cohen's $h = 0.20$, classed as small by Cohen's conventions), 388

participants per group are required for 80 % power at $\alpha = 0.05$ two-sided. The same 10-percentage-point increase from 10 % to 20 % corresponds to $h = 0.28$ (still small) but requires only 200 per group, and from 80 % to 90 % similarly requires 186 per group. The non-linear mapping between absolute proportion difference and detection cost is why h rather than raw difference is used for sample-size planning." Always translate h back to clinical proportions.

5.4.8 Practical Tips

- Prefer Cohen's h over the raw risk difference for sample-size and power calculations because h is the natural input to the Normal-approximation formulas and accounts for the non-constant variance of a proportion across the 0-to-1 scale.
- For rare events (near 0) or common events (near 1), even small raw differences correspond to relatively large h values and are detectable with fewer subjects than mid-range proportions; this counterintuitive property is a consequence of the variance-stabilising transformation and explains why screening trials of rare conditions can detect small absolute increases efficiently.
- Cohen's h is symmetric: $h(p_1, p_2) = -h(p_2, p_1)$, and the sign carries the direction of the comparison; always report the sign and the directional interpretation.
- In reporting and interpretation, translate h back to the raw proportions used as inputs; an h of 0.28 means little to a clinical reader without the corresponding "from 10 % to 20 %" framing.
- For 2×2 contingency-table tests of independence (rather than two-proportion comparisons), use Cohen's w rather than h; the two are related but designed for different test contexts.
- When the comparison is between a single proportion and a fixed reference (one-sample), Cohen's h still applies — ES.h(p_1, p_0) — and feeds into `pwr.p.test()` rather than `pwr.2p.test()`.

5.4.9 R Packages Used

`pwr::ES.h()` for the canonical Cohen's h calculation and `pwr::pwr.p.test()`, `pwr::pwr.2p.test()` for the corresponding power-analysis functions; `effectsize::cohens_h()` as a tidyverse-friendly alternative; `Hmisc::bsamsize()` for direct-proportion sample-size calculations that complement h-based planning; `MESS::power_prop_test()` for fast exact alternatives; `Mediana` for trial-design simulation including proportion-difference power.

5.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.4.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.5 Effect Size: Eta-Squared

5.5.1 Introduction

Eta-squared (η^2) is the standard effect-size measure for analysis of variance: the proportion of total variance in the outcome attributable to a factor. It complements the omnibus F-test, which reports significance, by quantifying magnitude — how much of the variability in the response is explained by the categorical factor of interest. Several variants of eta-squared are widely used in different ANOVA designs: classical η^2 in one-way designs, partial η_p^2 in factorial between-subjects designs, generalised η_G^2 in mixed designs that combine between- and within-subjects factors, and omega-squared ω^2 as an unbiased alternative preferred in small samples. Reporting the appropriate variant alongside the F-test result is now standard in psychology, neuroscience, and clinical-trial publications.

5.5.2 Prerequisites

A working understanding of one-way and factorial ANOVA, the partitioning of sums of squares into factor and error components, and the distinction between between-subjects and within-subjects factors.

5.5.3 Theory

Classical eta-squared is

$$\eta^2 = \frac{\mathrm{SS}_{\text{factor}}}{\mathrm{SS}_{\text{total}}}.$$

Partial eta-squared, for factorial designs, removes other factors' variance from the denominator:

$$\eta_p^2 = \frac{\mathrm{SS}_{\text{factor}}}{\mathrm{SS}_{\text{factor}} + \mathrm{SS}_{\text{error}}}.$$

Generalised eta-squared (Bakeman, 2005), for mixed designs combining within- and between-subjects factors, uses a denominator that depends on the design and is comparable across between- and within-subjects effects.

Omega-squared ω^2 corrects for the upward bias of η^2 in small samples:

$$\omega^2 = \frac{\mathrm{SS}_{\text{factor}} - \mathrm{df}_{\text{factor}} \cdot \mathrm{MSE}}{\mathrm{SS}_{\text{total}} + \mathrm{MSE}}.$$

Cohen's conventional thresholds for η^2 are 0.01 (small), 0.06 (medium), and 0.14 (large).

5.5.4 Assumptions

The same as the underlying ANOVA: independent observations (within a stratum), Normal residuals, homogeneous variances. The eta-squared family does not impose additional assumptions beyond those of the ANOVA from which the sums of squares are computed.

5.5.5 R Implementation

```
library(effectsize)
set.seed(2026)

df <- data.frame(
  A = factor(rep(c("A1", "A2"), each = 30)),
  B = factor(rep(c("B1", "B2", "B3"), 20)),
  y = rnorm(60)
)
df$y <- df$y + 0.5 * (df$A == "A2") + 0.3 * (df$B == "B2") + 0.1 * (df$B
 ↪  == "B3")

fit <- aov(y ~ A * B, data = df)

eta_squared(fit)
eta_squared(fit, partial = TRUE)
eta_squared(fit, generalized = TRUE)
omega_squared(fit)
```

5.5.6 Output & Results

`effectsize::eta_squared()` and `omega_squared()` return point estimates and confidence intervals for each ANOVA term. The output reports the appropriate variant for the requested design, with η_p^2 interpretable as the variance explained by each factor relative to itself plus error, holding other factors constant.

5.5.7 Interpretation

A reporting sentence: "The factorial ANOVA showed a small main effect of factor A on the outcome (partial $\eta_p^2 = 0.06$, 95 % CI 0.01 to 0.21, $F_{1,54} = 3.4$, $p = 0.07$); other terms were small and non-significant. Reporting partial η_p^2 rather than classical η^2 is appropriate for this between-subjects factorial design because each effect's variance share is computed relative to its own error stratum. Omega-squared values were 0.04 and 0.00 respectively, slightly more conservative as expected." Always specify which η^2 variant.

5.5.8 Practical Tips

- Report partial η_p^2 for between-subjects factorial designs; classical η^2 in factorial contexts conflates a factor's variance share with all other factors and is rarely the right summary.
- Use generalised η_G^2 (Bakeman, 2005) for mixed designs that combine within-subjects and between-subjects factors; it gives effect sizes that are directly comparable across the two types of factor, where partial η_p^2 is not.
- Prefer omega-squared ω^2 in small samples (typically $n < 30$ per cell); η^2 is upward-biased in small samples and ω^2 corrects this. Report both when feasible.
- Always report a confidence interval on the effect size, not just the point estimate; CIs on η^2 are non-central-F-based and are reported by `effectsize::eta_squared()` directly.
- Cohen's thresholds (0.01 / 0.06 / 0.14) come from behavioural research and may not be appropriate benchmarks in biomedical or physical-science contexts where larger effects are routinely expected; interpret thresholds in light of substantive expectations rather than treating them as universal.
- For the related $f^2 = \eta^2/(1 - \eta^2)$ effect size used in regression power analysis, the conversion is direct and useful when designing follow-up studies based on existing ANOVA results.

5.5.9 R Packages Used

`effectsize::eta_squared()`, `effectsize::omega_squared()`, and `effectsize::epsilon_squared()` for the canonical effect-size family with confidence intervals; `lsr::etaSquared()` for an alternative interface; `MOTE::eta.full.SS()` for textbook-companion calculations; `BayesFactor` for Bayesian effect-size estimation; `Superpower` for ANOVA-design power simulation that also produces eta-squared estimates.

5.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.5.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.6 Minimum Detectable Effect

5.6.1 Introduction

When the sample size for a study is constrained by budget, time, or availability of participants, the natural planning question is no longer "what n achieves target power?" but instead "given my fixed n, what is the smallest effect I can reasonably hope to detect?" The minimum detectable effect (MDE) reverses the usual sample-size calculation, solving for the effect size at which power equals the target — typically 80 % — given the fixed n, α, and analysis. Reporting an MDE is increasingly required for grant applications with constrained sample sizes, for pilot and exploratory studies, and for honest power-analysis reporting where a study cannot afford to enrol enough participants for the original target.

5.6.2 Prerequisites

A working understanding of statistical power, the relationship between α, effect size, sample size, and power, and the standard test-specific effect-size measures (Cohen's d, r, f^2, w).

5.6.3 Theory

Power, effect size, significance level, and sample size are linked by the non-central distribution of the test statistic, and solving for any one of these quantities given the other three is the standard inversion in any power-analysis framework. The MDE is defined as the effect size at which the planned test achieves a specified power (usually 0.80) under the planned α and the available sample size. Plotted as a curve of MDE vs. n, this gives a clear picture of the design's sensitivity across resource scenarios.

5.6.4 Assumptions

The same assumptions as the underlying test apply (Normality for t-tests, bivariate Normality for correlation, large expected counts for chi-squared, etc.); the MDE is a property of the planned analysis under the design, not a property of the observed data after the fact.

5.6.5 R Implementation

```r
library(pwr)

pwr.t.test(n = 40, sig.level = 0.05, power = 0.80,
           type = "two.sample", d = NULL)

pwr.r.test(n = 100, sig.level = 0.05, power = 0.80)

n_grid <- seq(10, 200, by = 5)
d_grid <- sapply(n_grid, function(n)
  pwr.t.test(n = n, sig.level = 0.05, power = 0.80, type =
  ↪  "two.sample")$d)

plot(n_grid, d_grid, type = "l", lwd = 2, col = "#2A9D8F",
     xlab = "n per group", ylab = "MDE (Cohen's d)",
     main = "Minimum detectable effect at power = 0.80")
```

5.6.6 Output & Results

For a two-sample t-test with $n = 40$ per group, the MDE at 80 % power is $d = 0.64$ (medium-to-large); doubling to $n = 200$ per group reduces the MDE to $d = 0.28$ (small-to-medium). For a correlation test with $n = 100$, the MDE is $r = 0.28$. Plotting the MDE-vs-n curve shows the diminishing-returns relationship that planners should understand when negotiating sample-size constraints.

5.6.7 Interpretation

A reporting sentence: "With the available sample of 40 participants per arm, the minimum detectable standardised effect (Cohen's d) at 80 % power and two-sided $\alpha = 0.05$ is 0.64 — a medium-to-large effect by Cohen's conventions. Effects smaller than this magnitude would likely remain undetected. The smallest clinically meaningful effect for the primary outcome is $d = 0.40$, so the planned study is underpowered for the clinical target; we report the MDE explicitly to clarify the inferential limits and propose a larger follow-up trial." Always discuss MDE relative to the clinically meaningful effect.

5.6.8 Practical Tips

- Report the minimum detectable effect explicitly when sample size is constrained; this prevents inflated claims of null effects ("no significant difference") when the study was simply too small to detect a clinically meaningful effect, a recurring problem in the literature.
- Couple MDE reporting with explicit discussion of whether the MDE is clinically or scientifically meaningful; an MDE that exceeds the smallest

meaningful effect tells readers the study could not have detected the truth even if it were present.
- For exploratory pilot studies, report MDE in place of formal hypothesis testing; pilot studies are not powered for inference and the MDE conveys the design's sensitivity honestly.
- MDE inflates rapidly for sub-group analyses (smaller n per cell), cluster-adjusted analyses (larger effective standard errors), repeated-measures designs with low correlation, and any setting where the effective sample size is reduced from the nominal n. Account for these factors in the MDE calculation, not just the headline figure.
- Simulation-based MDE estimation is recommended for complex designs (mixed-effects models, longitudinal data with informative dropout, generalised linear models with heavy-tailed outcomes) where the standard non-central-distribution formulas do not apply directly.
- Pre-register the MDE in the protocol or analysis plan; reporting MDE only after the fact when results turn out non-significant is methodologically suspect, while a pre-registered MDE protects against that interpretation.

5.6.9 R Packages Used

`pwr` family of functions with the effect-size argument set to `NULL` to solve for MDE; `pwrss` for extended MDE calculations across many test types; `WebPower` for an alternative interface; `simr::powerSim()` for simulation-based MDE in mixed-effects designs; `Superpower` for ANOVA-based MDE simulations.

5.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.6.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.7 Post-Hoc Power: A Controversy

5.7.1 Introduction

Post-hoc power – also called "observed power" – is the calculation of power using the observed effect size from the study that just ran. It is common in some

fields (especially when a study fails to reject) and near-universally criticised by statisticians.

5.7.2 Prerequisites

Power analysis, hypothesis testing.

5.7.3 Theory

Why it's problematic:

- Observed power is a monotone function of the p-value: smaller p always yields higher observed power. It conveys no information beyond the p-value itself (Hoenig and Heisey 2001).
- For non-significant results, observed power is bounded above by about 50 % (exactly 50 % when $p = \alpha$).
- It cannot tell you whether the original design was underpowered for a clinically important effect; it only restates how surprising the data were under the null.

Legitimate alternatives:

- **Sensitivity analysis**: compute power for a pre-specified clinically important effect, given n.
- **Confidence intervals** on the observed effect: communicate the range of effects consistent with the data.
- **Minimum detectable effect** (MDE): the smallest effect detectable at the study's n and power target, before the study ran.

5.7.4 Assumptions

None; the issue is epistemological.

5.7.5 R Implementation

```r
library(pwr)

# Simulated study that fails to reject
set.seed(2026)
x <- rnorm(30, 0.2, 1)
t_res <- t.test(x)
c(t = t_res$statistic, p = t_res$p.value)

# Post-hoc power (what many people do)
d_obs <- mean(x) / sd(x)
pwr.t.test(n = 30, d = d_obs, sig.level = 0.05, type = "one.sample")$power
```

```r
# Better: 95% CI for the mean and MDE
c(CI_lower = t_res$conf.int[1], CI_upper = t_res$conf.int[2])
pwr.t.test(n = 30, power = 0.80, type = "one.sample", sig.level = 0.05)$d
```

5.7.6 Output & Results

```
       t         p
   1.11      0.277      # non-significant

Observed power: 0.19     # uninformative

95% CI: (-0.18, 0.58)
MDE at n = 30: d = 0.52  # effect >= this detectable with 80% power
```

The CI and MDE together say: the effect could be anywhere from -0.18 to 0.58 SD; to detect a medium effect ($d = 0.52$) with 80 % power, $n = 30$ is adequate, so a negative finding here is informative.

5.7.7 Interpretation

Do not report observed power. Report the confidence interval on the observed effect (communicates uncertainty directly) and, if relevant, the MDE at the design stage (communicates what the study was equipped to detect).

5.7.8 Practical Tips

- Reviewers who request post-hoc power should be politely redirected to CI-based reporting.
- "We had low power because the effect turned out to be small" is circular; "our CI cannot rule out a clinically meaningful effect" is informative.
- Pre-registered power calculations are immune to this critique; they are design-stage, not data-driven.
- For replication studies, the effect of interest is from the original study, not the replication sample.
- When a study is genuinely underpowered, the honest conclusion is "inconclusive", not "no effect".

5.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.7.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.8 Power for Agreement (Kappa)

5.8.1 Introduction

Cohen's kappa measures agreement beyond chance between two raters on a categorical scale and is the de facto standard for inter-rater reliability with categorical outcomes. Sample-size planning for a kappa-based reliability study addresses one of two distinct questions: how many subjects are needed to estimate kappa with a desired precision (a target 95 % confidence-interval half-width around the expected value), or how many subjects are needed to test a hypothesis about kappa (typically that kappa exceeds a benchmark value such as 0 or 0.6) with adequate power. Modern reliability-study protocols are increasingly precision-driven rather than hypothesis-driven, because a "kappa is greater than zero" claim conveys little practical information about a clinical instrument.

5.8.2 Prerequisites

A working understanding of Cohen's kappa, the marginal-prevalence dependence of kappa's variance, and the distinction between precision-based and hypothesis-test sample-size questions.

5.8.3 Theory

For an expected kappa κ_1 with marginal prevalences π_1 and π_2 for the two raters, the asymptotic variance of the sample kappa is given by the Fleiss-Cicchetti-Everitt formula. Precision-based planning targets a 95 % CI half-width w:

$$n \approx \frac{(1-\kappa_1^2)\, z_{0.975}^2}{w^2 (1-p_e)^2},$$

with p_e the chance-expected agreement. Hypothesis-test planning targets the power to reject $H_0 : \kappa = \kappa_0$ in favour of $H_1 : \kappa = \kappa_1$, using a Normal-approximation z-test on the difference between estimated and hypothesised values, scaled by the appropriate standard errors.

5.8.4 Assumptions

Subjects are independent and each is rated once by each of the two raters; the expected kappa and the marginal prevalences are pre-specified from pilot data or substantive expectations; the rating scale is categorical (or ordinal with weighted kappa).

5.8.5 R Implementation

```
library(irr)

N.cohen.kappa(rate1 = 0.5, rate2 = 0.5,
              k1 = 0.70, k0 = 0,
              alpha = 0.05, power = 0.80,
              twosided = TRUE)

kappa_exp <- 0.70; p_e <- 0.5
w <- 0.05
n_prec <- qnorm(0.975)^2 * (1 - kappa_exp^2) / (w^2 * (1 - p_e)^2)
ceiling(n_prec)
```

5.8.6 Output & Results

`N.cohen.kappa()` returns the sample size for the hypothesis test of κ_1 vs κ_0 given the marginal rates. For an expected $\kappa_1 = 0.70$ against $\kappa_0 = 0$ at balanced marginals, roughly 40 subjects suffice for the hypothesis-test approach; precision-targeted with a 95 % CI half-width of 0.05 requires roughly 83 subjects, illustrating that precision-based planning typically demands more subjects than hypothesis-test planning.

5.8.7 Interpretation

A reporting sentence: "To estimate inter-rater Cohen's kappa with a 95 % CI half-width of 0.05 around an anticipated value of 0.70 (assumed marginal prevalences 0.5 in each rater), 83 subjects are required. The protocol enrols 100 to allow for 15 % unevaluable ratings. A hypothesis-test framing — power to reject $\kappa = 0$ at 80 % power — would require only 40 subjects, but the precision-based plan is preferred because the substantive question is the magnitude of agreement, not whether it differs from zero." Always state the planning paradigm.

5.8.8 Practical Tips

- Kappa's variance depends substantially on the marginal prevalences; very imbalanced marginals (e.g., one category at 5 % prevalence) inflate the variance and require more subjects than balanced marginals at the same expected kappa value.
- Weighted kappa for ordinal categories has smaller standard error than unweighted kappa for nominal categories with the same number of categories; adjust the sample-size calculation accordingly using the weighted-kappa variance formula.
- Multi-rater kappa (Fleiss's kappa) requires a different sample-size calculation than two-rater Cohen's kappa; use `samplesize::kappa.sample.size()` or simulation-based approaches for the multi-rater design.

- Always report the marginal prevalences and the assumed agreement matrix in the protocol, not just the expected kappa estimate; reviewers cannot reproduce the calculation without them, and the marginals materially affect the variance.
- For continuous ratings, the appropriate reliability statistic is the intraclass correlation coefficient (ICC), and a separate sample-size calculation applies; do not dichotomise continuous ratings to compute kappa.
- For the kappa-paradox situation where high observed agreement coincides with low kappa due to imbalanced marginals, consider reporting both kappa and the prevalence- and bias-adjusted kappa (PABAK) for a fuller picture of reliability.

5.8.9 R Packages Used

`irr::N.cohen.kappa()` for the canonical Cohen's-kappa hypothesis-test sample-size calculation; `samplesize::kappa.sample.size()` for an alternative interface and multi-rater extensions; `kappaSize` for kappa sample-size with multi-category outcomes; `simr` for simulation-based extensions; `psych` and `irr` for kappa analysis after data collection.

5.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.8.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.9 Power Analysis: Introduction

5.9.1 Introduction

Power analysis turns four quantities – effect size, significance level, power, and sample size – into a single equation: given any three, the fourth is determined. Every responsible study design pins down three of the four in advance to solve for the fourth (usually sample size). Skipping power analysis leaves studies under- or over-powered, with reputational and ethical consequences.

5.9.2 Prerequisites

Hypothesis testing, Type I and II errors.

5.9.3 Theory

The four quantities:

- **Effect size** (d, f, r, OR, RR): the magnitude of the true effect under the alternative. Must be pre-specified based on prior evidence, clinical relevance, or Cohen's conventions.
- **Significance level** α: the Type I error rate, almost always 0.05 two-sided.
- **Power** $1-\beta$: probability of detecting the true effect. Conventionally 0.80; confirmatory trials use 0.90.
- **Sample size** n: number of observations (or per group).

Given any three, the fourth is computed from the test's non-central distribution under H_1.

Sensitivity and trade-offs:

- Doubling n roughly shrinks the detectable effect by $\sqrt{2}$.
- Reducing α from 0.05 to 0.01 reduces power at the same n.
- Larger SD reduces power; more precise measurement increases it.
- Power for a one-sided test at α equals power for a two-sided test at 2α (when the effect is in the right direction).

5.9.4 Assumptions

The test's sampling distribution under H_1 must be known (or approximable). Power calculations are as valid as the assumed effect-size magnitude; a plausible range is often reported.

5.9.5 R Implementation

```
library(pwr)

# Solve for n given d, alpha, power
pwr.t.test(d = 0.5, sig.level = 0.05, power = 0.80,
           type = "two.sample", alternative = "two.sided")

# Solve for d given n and power (minimum detectable effect)
pwr.t.test(n = 30, sig.level = 0.05, power = 0.80,
           type = "two.sample")

# Power curves across d
d_grid <- seq(0.1, 1, by = 0.05)
pw_n30  <- sapply(d_grid, function(d) pwr.t.test(n = 30, d = d)$power)
pw_n100 <- sapply(d_grid, function(d) pwr.t.test(n = 100, d = d)$power)

plot(d_grid, pw_n30, type = "l", col = "#F4A261", lwd = 2,
     xlab = "Cohen's d", ylab = "Power",
```

```
     main = "Power curves: n = 30 vs n = 100 per group")
lines(d_grid, pw_n100, col = "#2A9D8F", lwd = 2)
abline(h = 0.80, lty = 2)
legend("bottomright", c("n = 30", "n = 100"),
       col = c("#F4A261", "#2A9D8F"), lwd = 2)
```

5.9.6 Output & Results

```
     Two-sample t test power calculation

              n = 63.77
              d = 0.5
      sig.level = 0.05
          power = 0.80
    alternative = two.sided

     Two-sample t test power calculation

              n = 30
              d = 0.738
      sig.level = 0.05
          power = 0.80
```

64 per arm for a medium effect at 80 % power. With only 30 per arm, the minimum detectable effect is $d = 0.74$.

5.9.7 Interpretation

For a grant or protocol: "Assuming a between-group standardised difference of $d = 0.5$ (a medium effect), $\alpha = 0.05$ two-sided, and power $= 0.80$, the required sample size is 64 per arm (128 total). We plan to enrol 70 per arm to allow for 10 % loss to follow-up."

5.9.8 Practical Tips

- Choose the effect size from published data or pilot studies, not from Cohen's generic thresholds.
- Always inflate for dropout; report the final n as target.
- Sensitivity analysis: provide n for several plausible d values.
- Do not perform post-hoc power analysis with the observed effect; it is uninformative and sometimes misleading.
- For complex designs (cluster, multilevel), use simulation-based power (`simstudy`, custom Monte Carlo) rather than formulas.

5.9.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.9.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.10 Power for One-Way ANOVA

5.10.1 Introduction

Power analysis for one-way ANOVA computes the per-group sample size needed to detect a specified pattern of differences among three or more group means with adequate probability. Where the two-sample t-test handles the simple two-group comparison, the one-way ANOVA generalises the comparison to any number of groups, and its power calculation must therefore handle a multi-group effect-size measure rather than a single mean difference. Cohen's f is the standard standardised effect size; it captures the spread of true group means relative to the within-group standard deviation in a single number that maps cleanly to the non-central F distribution under the alternative hypothesis.

5.10.2 Prerequisites

A working understanding of one-way ANOVA, the omnibus F-test, the non-central F distribution, and Cohen's f as the standardised effect size for ANOVA.

5.10.3 Theory

Cohen's f for one-way ANOVA is

$$f = \frac{\sigma_{\text{between}}}{\sigma_{\text{within}}},$$

the SD of the true group means divided by the within-group SD. The relationship to η^2 is $f = \sqrt{\eta^2/(1-\eta^2)}$. Cohen's conventional benchmarks are 0.10 (small), 0.25 (medium), and 0.40 (large). Under the alternative hypothesis, the omnibus F-statistic follows a non-central F with $(k-1, N-k)$ degrees of freedom and non-centrality $\lambda = Nf^2$, where N is the total sample size across k groups; power is the tail probability of this non-central F above the critical value.

5.10.4 Assumptions

The design is balanced (equal n per group; modest imbalance is tolerable but reduces efficiency), within-group variances are approximately equal (homogeneity of variance), and residuals are approximately Normal. Welch-corrected variants accommodate variance heterogeneity at the cost of slightly more complex power calculations.

5.10.5 R Implementation

```
library(pwr)

pwr.anova.test(k = 4, f = 0.25, sig.level = 0.05, power = 0.80)

pwr.anova.test(k = 3, f = 0.10, power = 0.80)
pwr.anova.test(k = 3, f = 0.25, power = 0.80)
pwr.anova.test(k = 3, f = 0.40, power = 0.80)

group_means <- c(50, 55, 58)
pooled_sd   <- 8
grand_mean  <- mean(group_means)
f_est <- sqrt(mean((group_means - grand_mean)^2)) / pooled_sd
f_est
```

5.10.6 Output & Results

`pwr.anova.test()` returns the per-group sample size required to achieve target power at specified f, α, and number of groups. For $k = 4$ groups at medium effect $f = 0.25$, 45 per group (180 total) achieves 80 % power; computing f directly from substantive expected group means and a pooled within-group SD ties the calculation to scientifically defensible assumptions.

5.10.7 Interpretation

A reporting sentence: "With four groups and an anticipated Cohen's $f = 0.25$ (medium effect by Cohen's convention; equivalent to expected group means differing by approximately 0.5 within-group SDs), $n = 45$ participants per group (180 total) are required for 80 % power at $\alpha = 0.05$ using a one-way ANOVA. The protocol allocates 50 per group to allow for 10 % attrition, and pre-specified pairwise Tukey HSD comparisons follow if the omnibus is significant." Always describe f in terms of the underlying group-mean structure.

5.10.8 Practical Tips

- Compute Cohen's f from substantive expectations about group means and within-group SD wherever possible, rather than relying on Cohen's

medium/large benchmarks; tying the calculation to the actual expected pattern is far more defensible than invoking generic effect-size labels.
- For unequal group variances, power is approximate under the standard formulas; use Welch-adjusted simulation (or `WebPower::wp.kanova()` with explicit variance specification) when heterogeneity is substantial.
- Unbalanced designs have lower power than balanced designs at the same total N; the loss is small for modest imbalances but grows quickly for ratios more extreme than 2:1. Plan for balance whenever feasible.
- For many groups with a single pre-specified contrast of primary interest (e.g., linear trend across ordered groups, treatment-vs-control contrast), pre-specified contrast tests have higher power than the omnibus F-test and should be used in preference.
- Simulation-based power (`simr`, `Superpower`, `faux`) is more flexible for complex designs — unbalanced cells, heterogeneous variances, non-Normal outcomes, mixed-effects structures — and is increasingly the default approach in modern protocol development.
- For factorial ANOVA (two or more between-subjects factors), the calculation extends naturally with effect sizes for each main effect and interaction; `Superpower::ANOVA_power()` is the standard tool.

5.10.9 R Packages Used

`pwr::pwr.anova.test()` for canonical balanced one-way ANOVA power; `WebPower::wp.kanova()` for an alternative interface and unequal-variance extensions; `Superpower::ANOVA_power()` and `Superpower::ANOVA_exact()` for factorial-ANOVA power simulation; `pwrss::pwrss.f.ancova()` for ANCOVA-power calculations; `simr::powerSim()` for general simulation-based ANOVA power.

5.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.10.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.11 Power for Bland-Altman Studies

5.11.1 Introduction

Bland-Altman analysis is the standard method for comparing two quantitative measurement methods — a new instrument against a reference standard, two raters' continuous scores, two clinical-laboratory assays. The central output is the 95 % limits of agreement (LoA), defined as the mean difference ± 1.96 SD of paired differences, which describe the range within which 95 % of differences between the two methods are expected to fall. Sample-size planning for a Bland-Altman study therefore targets the precision (CI half-width) of the LoA estimates rather than power to reject a null. With too few subjects, the LoA themselves are estimated imprecisely, and clinical-acceptance decisions about whether the methods are interchangeable become unreliable.

5.11.2 Prerequisites

A working understanding of the Bland-Altman analysis framework, the calculation of limits of agreement, and the standard-error theory for sample quantiles and Normal-based confidence intervals.

5.11.3 Theory

The standard error of each limit of agreement is approximately

$$\mathrm{SE}(\mathrm{LoA}) \approx \sqrt{3}\,\sigma_d/\sqrt{n},$$

where σ_d is the SD of the paired differences between methods. The 95 % CI half-width on each LoA is therefore approximately $1.96\sqrt{3}\sigma_d/\sqrt{n}$. To achieve a target half-width w:

$$n \approx \left(\frac{1.96\sqrt{3}\sigma_d}{w}\right)^2 \approx \frac{11.5\,\sigma_d^2}{w^2}.$$

The factor $\sqrt{3}$ arises because the LoA is a linear combination of the sample mean and sample SD; the formula accounts for both components' uncertainty under Normal-distribution theory.

5.11.4 Assumptions

The paired differences are approximately Normally distributed, no systematic bias varies across the measurement range (proportional bias), subjects are independent, and the SD of differences is reasonably known from pilot data. The Bland-Altman 1999 extension handles repeated measurements per subject with a different formula.

5.11.5 R Implementation

```r
sigma_d <- 5; w <- 1
n_req <- 11.5 * sigma_d^2 / w^2
ceiling(n_req)

set.seed(2026)
n <- 288
diff <- rnorm(n, mean = 0, sd = sigma_d)
mean_diff <- mean(diff)
sd_diff   <- sd(diff)
LoA_lower <- mean_diff - 1.96 * sd_diff
LoA_upper <- mean_diff + 1.96 * sd_diff
SE_LoA    <- sd_diff * sqrt(3 / n)
half_ci   <- 1.96 * SE_LoA
c(LoA_lower, LoA_upper, half_ci)
```

5.11.6 Output & Results

The closed-form calculation gives $n \approx 288$ for $\sigma_d = 5$ and target half-width 1 unit on each LoA. The simulation block confirms the empirical CI half-width matches the formula prediction; reporting both the analytical calculation and a Monte Carlo verification is good practice when the underlying distributional assumptions are uncertain.

5.11.7 Interpretation

A reporting sentence: "To estimate the 95 % limits of agreement between the two methods with a 95 % CI half-width of ± 1 unit on each limit (the pre-specified clinical-acceptance criterion), $n = 288$ paired measurements are required, assuming a within-subject difference SD of 5 units from pilot data. The protocol enrols 320 to allow for 10 % unevaluable measurements. Sensitivity analyses across $\sigma_d \in [4, 6]$ yield required n from 184 to 414." Always report the target half-width and pilot-derived σ_d.

5.11.8 Practical Tips

- Precision of the LoA is dominated by the SD of differences σ_d; pilot data to estimate σ_d are essential, and protocols that rely on guessed σ_d values should report sensitivity analyses across a plausible range.
- For repeated measurements per subject (multiple paired observations within each individual), the analysis and the sample-size formula both change; use the Bland-Altman 1999 extension and its corresponding precision formula, which accounts for the within-subject correlation.
- Non-constant bias across the measurement range (proportional bias) invalidates simple LoA reporting; regress differences on means to detect

proportional bias, and if present, compute LoA conditional on the measurement value.
- For one-sided clinical-acceptance criteria (e.g., upper LoA must be below a tolerance threshold), adjust the calculation to target a one-sided 95 % CI upper bound; the sample-size factor changes correspondingly and is well-documented in the precision-based literature.
- Report the target LoA half-width explicitly in the methods section; vague phrases like "adequate precision" are no longer acceptable in regulatory submissions or method-comparison publications, where reviewers expect explicit precision targets.
- For ratio-scale outcomes with proportional measurement error, log-transform before computing LoA; the LoA on the log scale back-transforms to a multiplicative LoA that is more interpretable.

5.11.9 R Packages Used

`MethComp` for canonical Bland-Altman analysis with confidence intervals on LoA; `BlandAltmanLeh` for an alternative interface with ggplot-style plotting; `pwr` and `MBESS` for general precision-based sample-size tools that translate to LoA contexts; `rmcorr` for repeated-measures correlation alternatives in method-comparison data; `agRee` for ICC-based agreement statistics complementing LoA reporting.

5.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.11.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.12 Power for Chi-Squared Tests

5.12.1 Introduction

Power analysis for chi-squared tests — goodness-of-fit against a hypothesised distribution and independence in a two-way contingency table — uses the non-central chi-squared distribution under the alternative hypothesis. The standardised effect size for these tests is Cohen's w, which combines the difference between hypothesised and expected cell proportions into a single number that

maps directly to the non-centrality parameter of the test statistic. Power calculations are essential in the design of survey research, epidemiological category-association studies, market-segmentation analyses, and any setting where the central question is whether categorical proportions differ from a reference or vary across rows of a contingency table.

5.12.2 Prerequisites

A working understanding of chi-squared goodness-of-fit and contingency-table tests, the chi-squared distribution and its non-central generalisation, and Cohen's w as the standardised effect-size measure.

5.12.3 Theory

Cohen's w is

$$w = \sqrt{\sum_i \frac{(p_{1i} - p_{0i})^2}{p_{0i}}},$$

where p_{0i} are the cell proportions expected under the null hypothesis and p_{1i} under the alternative. Equivalently, $w = \sqrt{\chi^2/n}$ at the expected effect. Under the alternative, the test statistic follows a non-central chi-squared with the test's standard degrees of freedom and non-centrality parameter $\lambda = nw^2$. Conventional benchmarks for w are 0.10 (small), 0.30 (medium), and 0.50 (large).

5.12.4 Assumptions

Observations are independent, expected cell counts are large enough that the chi-squared approximation is valid (typically ≥ 5 per cell, though stricter thresholds apply for small tables), and the cell proportions under the alternative hypothesis are pre-specified.

5.12.5 R Implementation

```r
library(pwr)

pwr.chisq.test(w = 0.3, df = 3, sig.level = 0.05, power = 0.80)

pwr.chisq.test(w = 0.3, df = 4, sig.level = 0.05, power = 0.80)

p0 <- c(0.25, 0.25, 0.25, 0.25)
p1 <- c(0.40, 0.30, 0.20, 0.10)
w  <- sqrt(sum((p1 - p0)^2 / p0))
w
pwr.chisq.test(w = w, df = 3, power = 0.80)
```

5.12.6 Output & Results

`pwr.chisq.test()` returns the required sample size to achieve target power against a specified w, α, and df. For a medium effect $w = 0.3$ with 3 df (4-category goodness-of-fit), $n = 122$; with 4 df (a 3×3 contingency table), $n = 133$. Computing w directly from hypothesised cell proportions is the recommended workflow because it ties the calculation to substantive distributional assumptions.

5.12.7 Interpretation

A reporting sentence: "To detect a deviation from equal proportions across four categories of medium magnitude (Cohen's $w = 0.30$, equivalent to expected proportions $(0.40, 0.30, 0.20, 0.10)$ vs. uniform null) at 80 % power and $\alpha = 0.05$, $n = 122$ observations are required. With 5 % attrition, the protocol enrols 130 participants. Sensitivity analyses across $w \in [0.20, 0.40]$ are reported in the supplement to bracket uncertainty about the true effect size." Always state both w and the underlying proportional hypothesis.

5.12.8 Practical Tips

- Convert real-world quantities — relative risks, odds ratios, differences in proportions — to Cohen's w via the contingency-table formula rather than guessing; using a generic "medium" benchmark when the underlying scientific question implies a different magnitude is a recurring source of mis-powered studies.
- Required sample size grows roughly linearly with degrees of freedom at fixed w; spreading effects across more cells (e.g., 5×5 vs 2×2) reduces the per-cell signal and inflates the sample-size requirement.
- For small expected cell counts that would invalidate the chi-squared approximation, plan to use Fisher's exact test and estimate power by Monte Carlo simulation; the asymptotic chi-squared power calculation is unreliable in the small-sample regime where exact tests are needed.
- If the alternative-hypothesis cell proportions are themselves uncertain, pre-specify a sensitivity range and report power across the range; this is increasingly expected by reviewers and protects against under-powering when the assumed proportions turn out to be optimistic.
- Chi-squared power calculations are approximate (the non-central chi-squared assumes large samples); simulation-based power is safer for complex designs, sparse-data scenarios, or when the test statistic deviates from the standard Pearson form.
- For one-way exact-binomial tests of a single proportion, use specialised tools like `pwr.p.test()` or `binom::power.binom()` rather than the chi-squared approximation; the exact methods are more accurate at small samples.

5.12.9 R Packages Used

`pwr::pwr.chisq.test()` for the canonical Cohen's w chi-squared power calculation; `WebPower::wp.chisq()` for an alternative interface; `MESS::power_prop_test()` for two-proportion specific power; `binom` for exact-binomial sample-size tools; `Mediana` for trial-design simulation including chi-squared-test power across more complex designs.

5.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.12.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.13 Power for Cluster-RCT

5.13.1 Introduction

In cluster-randomised trials (CRTs), treatment is assigned at the cluster (clinic, school, village) rather than individual level. Within-cluster correlation inflates variance, reducing effective sample size and requiring more individuals than an individually randomised trial.

5.13.2 Prerequisites

Intraclass correlation, CRT design.

5.13.3 Theory

The **design effect** (DE) converts individual-level sample size to cluster-RCT sample size:

$$\text{DE} = 1 + (\bar{m} - 1)\rho_{ICC},$$

where $\bar{m}$ is the average cluster size and ρ_{ICC} is the intraclass correlation.

Required individual-level n for an individually-randomised design, multiplied by DE, gives the cluster-RCT total. Required clusters = total / $\bar{m}$.

5.13.4 Assumptions

- Balanced or nearly balanced cluster sizes.
- Pre-specified ICC (from pilot or literature).
- Analysis adjusts for clustering (mixed models or GEE).

5.13.5 R Implementation

```r
library(clusterPower)

# Continuous outcome: detect a 5-unit mean difference, sigma = 15
# ICC = 0.02, 30 subjects per cluster
# Individual n for same power:
n_ind <- 2 * 15^2 * (qnorm(0.975) + qnorm(0.80))^2 / 5^2
n_ind

rho <- 0.02
m_bar <- 30
DE <- 1 + (m_bar - 1) * rho
c(design_effect = DE,
  cluster_total_n = n_ind * DE,
  clusters_per_arm = ceiling(n_ind * DE / m_bar / 2))

# Direct calculation via clusterPower
cpa.normal(alpha = 0.05, power = 0.80, nclusters = NA,
           nsubjects = 30, d = 5 / 15,   # d = delta / sigma
           ICC = 0.02)
```

5.13.6 Output & Results

Individual $n \approx 143$ per arm; DE = 1.58 at ICC 0.02 and cluster size 30, giving 226 per arm (about 8 clusters of 30 per arm).

5.13.7 Interpretation

"With an assumed ICC of 0.02 and 30 subjects per cluster, the design effect is 1.58. To detect a standardised difference of 0.33 (5 units / 15 SD) at 80 % power and two-sided $\alpha = 0.05$, 8 clusters per arm are required (total 480 subjects)."

5.13.8 Practical Tips

- ICC estimates vary widely; use pilot data, literature, or sensitivity ranges (0.001 to 0.1).
- Unequal cluster sizes reduce power; inflate n by 10-20 % as buffer.
- Large m increases DE linearly in ICC; adding clusters is more efficient than more per-cluster recruitment.

- Stratified or matched CRTs (pairs of clusters) reduce DE.
- Analysis model: linear mixed model with random cluster effects; GEE with exchangeable correlation; cluster-level summary.

5.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.13.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.14 Power for Correlation Tests

5.14.1 Introduction

Power analysis for correlation tests determines the sample size required to detect a specified population correlation ρ different from zero (or different from a non-zero reference) with adequate probability. Correlation studies are pervasive in clinical research, psychometrics, behavioural science, biomarker validation, and any setting where the strength of association between two continuous variables is the central question. The standard analytical approach uses Fisher's z-transformation, which converts the sampling distribution of the Pearson correlation into an approximately Normal distribution with a simple variance formula, supporting clean closed-form sample-size calculations.

5.14.2 Prerequisites

A working understanding of Pearson and Spearman correlation, the Fisher z-transformation, and the relationship between effect size and required sample size in inferential statistics.

5.14.3 Theory

Fisher's z-transformation is

$$z = \tfrac{1}{2} \log \frac{1+r}{1-r} = \operatorname{atanh}(r),$$

with $\mathrm{SE}(z) \approx 1/\sqrt{n-3}$. The required sample size to detect ρ different from 0 at two-sided α with power $1-\beta$ is

$$n \approx \left(\frac{z_{1-\alpha/2} + z_{1-\beta}}{\operatorname{atanh}(\rho)} \right)^2 + 3.$$

Cohen's effect-size benchmarks for r are 0.10 (small), 0.30 (medium), and 0.50 (large). Spearman's rank correlation has slightly less efficient inference under bivariate Normality (asymptotic relative efficiency $9/\pi^2 \approx 0.91$), so a 9 % inflation in n converts a Pearson sample-size calculation to its Spearman equivalent.

5.14.4 Assumptions

The data are bivariate Normal for the Pearson test, the pairs are independent and identically distributed, and the population correlation is genuinely well-defined. Spearman correlation tolerates monotone non-linearity but its power calculation is approximate.

5.14.5 R Implementation

```r
library(pwr)

pwr.r.test(r = 0.30, sig.level = 0.05, power = 0.80)

pwr.r.test(r = 0.10, power = 0.80)

r_grid <- seq(0.1, 0.6, by = 0.05)
pw <- sapply(r_grid, function(r) pwr.r.test(r = r, n = 50)$power)
data.frame(r = r_grid, power_n50 = round(pw, 2))
```

5.14.6 Output & Results

`pwr.r.test()` returns the required n at specified r, α, and power. For $r = 0.30$ at 80 % power and two-sided $\alpha = 0.05$, $n = 85$; for $r = 0.10$, $n = 782$ — illustrating how dramatically small correlations inflate the required sample size. The power-by-correlation table at fixed n shows the achievable power across the range of plausible correlations and is a useful supplement to the single-point calculation.

5.14.7 Interpretation

A reporting sentence: "To detect a medium correlation of $r = 0.30$ with 80 % power at two-sided $\alpha = 0.05$, a sample of 85 participants is required. With the planned $n = 50$, achievable power for the same effect is 56 %, leaving the study substantially under-powered. The protocol therefore enrols 90 participants to allow for 5 % attrition. Sensitivity analyses across $\rho \in [0.20, 0.40]$ show required n ranging from 47 to 191." Always report sensitivity over a plausible ρ range.

5.14.8 Practical Tips

- Testing ρ against a non-zero reference value requires a modified Fisher-z formula that accounts for the difference between $\text{atanh}(\rho_0)$ and $\text{atanh}(\rho_1)$; use `pwrss::pwrss.z.corr()` or compute manually rather than relying on the standard `pwr.r.test()` which assumes $\rho_0 = 0$.
- For Spearman correlations, inflate the Pearson-based n by approximately 9 % to reflect the slight efficiency loss of the rank-based test under bivariate Normality; for non-Normal underlying data the rank-based test can actually be more efficient.
- Correlations near ± 1 have lower sampling variance than correlations near zero (a consequence of the bounded support); Fisher's z-transformation handles this automatically by mapping r to a scale where the variance is approximately constant.
- When many correlations are tested simultaneously (e.g., a correlation matrix of 10 variables yields 45 pairs), adjust α for multiplicity using Bonferroni or false-discovery-rate procedures, and recompute power against the adjusted threshold.
- Sensitivity analysis across plausible ρ is routine in correlational study planning; protocols with a single point ρ are increasingly flagged by reviewers, who expect a range and a justification of the lower bound used for sample-size determination.
- For paired or longitudinal correlation comparisons (e.g., comparing ρ at two time points), use Steiger's modified Fisher-z test and its dedicated power calculation; the standard formulas apply only to a single correlation against a fixed reference.

5.14.9 R Packages Used

`pwr::pwr.r.test()` for the canonical Fisher-z-based power calculation; `pwrss::pwrss.z.corr()` for non-zero reference and other extensions; `WebPower::wp.correlation()` for an alternative interface; `simr` and `Superpower` for simulation-based correlation power across complex designs; `boot` for bootstrap-based power estimation when assumptions are uncertain.

5.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.14.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting

- Power: closed-form and simulation

5.15 Power for Cox Regression

5.15.1 Introduction

In survival analysis, statistical power is driven by the number of **events**, not by the number of enrolled subjects. A Cox regression power calculation determines the required events and, from assumed accrual and follow-up, the number of subjects.

5.15.2 Prerequisites

Cox proportional hazards, survival analysis.

5.15.3 Theory

For a continuous predictor with standardised-coefficient effect, the required number of events for Wald test power at $1-\beta$ and two-sided α is

$$D = \frac{(z_{1-\alpha/2} + z_{1-\beta})^2}{\sigma_X^2 \log^2(HR)}.$$

For a binary predictor with allocation π (fraction in one group):

$$D = \frac{(z_{1-\alpha/2} + z_{1-\beta})^2}{\pi(1-\pi) \log^2(HR)}.$$

Required subjects $N = D/p_{\text{event}}$, where p_{event} is the probability of observing the event during follow-up; depends on survival, accrual, and censoring.

5.15.4 Assumptions

- Proportional hazards.
- Independent censoring.
- Known baseline event rate or survival function.

5.15.5 R Implementation

```
library(powerSurvEpi)

# Two-arm trial: equal allocation, HR = 0.7, 1-year survival = 0.80 in
 ↪  control
ssizeCT.default(power = 0.80, k = 1,
```

```
                pE = 1 - 0.70, pC = 1 - 0.80,
                RR = 0.70, alpha = 0.05)

# Number of events for a continuous predictor
# HR per 1-SD increase of 1.4
D <- (qnorm(0.975) + qnorm(0.80))^2 / log(1.4)^2
D
```

5.15.6 Output & Results

For HR = 0.70 with equal allocation and event rates of 20 % (treatment) and 30 % (control), about 500 events and 1000 subjects are required.

5.15.7 Interpretation

"The study requires approximately 500 events to detect a hazard ratio of 0.70 with 80 % power. Assuming the expected event probabilities during follow-up, we need to enrol 1000 patients."

5.15.8 Practical Tips

- Events, not subjects, are the currency of power in survival; count accumulated events, not randomised subjects.
- Report both target events and target N; specify the assumed accrual period and follow-up.
- For interim analyses, use group-sequential boundaries (O'Brien-Fleming, Pocock) to inflate the final sample accordingly.
- Non-proportional hazards reduce power; use restricted mean survival time or weighted log-rank alternatives.
- For competing risks, use Fine-Gray or cause-specific power calculations.

5.15.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.15.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.16 Power for Crossover Trials

5.16.1 Introduction

In a crossover trial, each subject receives multiple treatments in sequence with a washout period between. The paired comparison cancels between-subject variability, often requiring far fewer subjects than parallel-group trials. The classic 2x2 crossover has two treatments and two periods.

5.16.2 Prerequisites

Paired t-test, power analysis, crossover design basics.

5.16.3 Theory

In a 2x2 crossover, the period-adjusted within-subject difference has variance σ_w^2, typically estimated from prior crossover data. Sample size is analogous to the paired t-test with $\sigma_d = \sigma_w\sqrt{2}$:

$$n \approx 2\sigma_w^2(z_{1-\alpha/2} + z_{1-\beta})^2/\delta^2.$$

Compared to parallel-group with pooled σ: approximately half the subjects for the same δ when $\sigma_w < \sigma_{\text{between}}$.

Carryover effects invalidate the simple formula; a washout period is planned to eliminate them.

5.16.4 Assumptions

- No carryover; washout adequate.
- Period effects modelled if present.
- Within-subject variance known or estimable.

5.16.5 R Implementation

```
library(PowerTOST); library(pwr)

# 2x2 crossover bioequivalence: CV within = 20%, ratio = 0.95
sampleN.TOST(CV = 0.20, theta0 = 0.95,
             theta1 = 0.80, theta2 = 1.25,
             alpha = 0.05, targetpower = 0.80,
             design = "2x2")

# Therapeutic crossover: mean difference = 5, sigma_w = 10
n_pair <- 2 * (qnorm(0.975) + qnorm(0.80))^2 * 10^2 / 5^2
n_pair
```

```r
# Equivalent paired-t calculation
pwr.t.test(d = 5 / 10, type = "paired", power = 0.80)
```

5.16.6 Output & Results

Bioequivalence 2x2 at 20 % CV, ratio 0.95: 20 subjects. Therapeutic 5-unit effect with $\sigma_w = 10$: 34 subjects.

5.16.7 Interpretation

"The 2x2 crossover trial requires 20 subjects for 80 % power to establish bioequivalence within the 80-125 % range, assuming a 20 % within-subject CV and a true ratio of 0.95."

5.16.8 Practical Tips

- Crossover saves subjects but doubles measurement commitment per subject; adherence matters.
- Washout must be long enough to eliminate pharmacokinetic and physiological carryover.
- More than two periods (Latin squares, Williams designs) can be more efficient but require careful balancing.
- Exclude subjects with missing period data (or use mixed models).
- For strictly sequential effects (learning, fatigue), random order assignment and period-as-factor in the model help.

5.16.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.16.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.17 Power for Diagnostic Accuracy

5.17.1 Introduction

Sample-size planning for diagnostic studies typically targets **precision** (CI width) for sensitivity and specificity rather than power against a hypothesis.

Separate calculations apply to diseased and non-diseased cohorts.

5.17.2 Prerequisites

Sensitivity, specificity, proportion CIs.

5.17.3 Theory

For a Wilson 95 % CI on sensitivity of anticipated value p_{sens} with desired half-width w:

$$n_{\text{dis}} \approx \frac{z^2_{0.975} p_{\text{sens}}(1 - p_{\text{sens}})}{w^2}.$$

Analogously for specificity using non-diseased. Total sample $= n_{\text{dis}} + n_{\text{non-dis}}$, with the ratio determined by the prevalence of disease in the source population.

For hypothesis-testing framing (is sensitivity $>$ some reference), the one-proportion power formula applies.

5.17.4 Assumptions

- Anticipated sensitivity / specificity from pilot or literature.
- Independent gold-standard verification for every subject.

5.17.5 R Implementation

```r
# Design 95% CI half-width 5% on sensitivity = 0.90
p_sens <- 0.90; w <- 0.05
n_dis <- qnorm(0.975)^2 * p_sens * (1 - p_sens) / w^2
ceiling(n_dis)

# Spec = 0.85, same width
p_spec <- 0.85
n_non <- qnorm(0.975)^2 * p_spec * (1 - p_spec) / w^2
ceiling(n_non)

# Hypothesis-test framing: sensitivity > 0.80
# Expected sens = 0.90, alpha = 0.05, power = 0.80
library(MKmisc)
power.prop1.test(p0 = 0.80, p1 = 0.90, sig.level = 0.05, power = 0.80)
```

5.17.6 Output & Results

For sens 0.90 +/- 5 %: 138 diseased cases. For spec 0.85 +/- 5 %: 196 non-diseased. Hypothesis-test $n \approx 130$ diseased cases to show sensitivity > 0.80.

5.17.7 Interpretation

"To estimate sensitivity of 0.90 with a 95 % CI half-width of 5 %, at least 138 diseased cases are required. Assuming a disease prevalence of 30 % in the source population, a total enrolment of 460 subjects provides adequate non-diseased controls."

5.17.8 Practical Tips

- Plan separately for sensitivity and specificity; the two calculations have independent sample-size requirements per cohort.
- Use Wilson or exact intervals for the reported CI; Wald intervals can be optimistic.
- STARD reporting guidelines recommend reporting both prevalence and the numbers of diseased / non-diseased subjects.
- Paired designs (both tests on each subject) use McNemar-style within-subject comparisons; they are more efficient than unpaired designs.
- AUC-based sample sizes exist (e.g., Hanley-McNeil); different from sensitivity-only calculations.

5.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.17.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.18 Power for Equivalence (TOST)

5.18.1 Introduction

Power analysis for equivalence testing via the two-one-sided-tests (TOST) procedure requires that both one-sided null hypotheses — that the true effect lies below the lower equivalence bound and that it lies above the upper equivalence bound — be rejected at level α. The required sample size depends on the equivalence margin (set by clinical or regulatory convention), the anticipated true effect (typically zero or near-zero for an honest equivalence claim), and the variance. TOST sample-size calculations are widely used in bioequivalence studies of generic drugs (where the regulatory margin is fixed at 80–125

% on the log scale), in non-inferiority trials reframed as equivalence, and in therapeutic-equivalence studies of competing devices or formulations.

5.18.2 Prerequisites

A working understanding of TOST equivalence testing, the role of the equivalence margin, and the difference between equivalence (CI within the margin) and traditional superiority testing (CI excluding the null).

5.18.3 Theory

For two groups with true mean difference δ and pooled SD σ, and symmetric equivalence margin $\pm\Delta$, the power to conclude equivalence is approximately

$$\Pr\left(\frac{|\bar{X}_1 - \bar{X}_2 - \delta|}{\sigma\sqrt{2/n}} < z_{1-\alpha} \cdot \frac{\Delta - |\delta|}{\Delta \cdot \sigma\sqrt{2/n}}\right).$$

Two important features fall out of this expression. First, when the true effect is at the centre of the equivalence margin ($\delta = 0$), the required sample size is minimised. Second, when the true effect equals the boundary ($|\delta| = \Delta$), equivalence cannot be established regardless of n — the test is fundamentally underpowered at the boundary.

5.18.4 Assumptions

The equivalence margin is pre-specified and substantively justified, the anticipated true effect δ is realistically estimated, the data are approximately Normal or the sample is large enough for the central limit theorem, and the variance is known or accurately estimated from pilot data.

5.18.5 R Implementation

```r
library(TOSTER); library(PowerTOST)

sampleN.TOST(alpha = 0.05, targetpower = 0.80,
             theta0 = 1.00, theta1 = 0.80, theta2 = 1.25,
             CV = 0.20, design = "2x2")

power_t_TOST(alpha = 0.05, low_eqbound = -2, high_eqbound = 2,
             mu = 0, sd = 5, type = "two.sample", n = 65)
```

5.18.6 Output & Results

`sampleN.TOST()` returns the required sample size for bioequivalence with the regulatory 80–125 % margin on the log scale; for 20 % within-subject CV and

true ratio 1.00, $n \approx 24$ in a 2×2 crossover. `power_t_TOST()` computes power for a generic two-sample equivalence test with arbitrary symmetric or asymmetric margin: at ± 2 unit margin, true difference 0, and SD 5, $n = 65$ per group achieves 81 % power.

5.18.7 Interpretation

A reporting sentence: "To establish bioequivalence (regulatory 80–125 % equivalence margin on the log scale) with 80 % power at $\alpha = 0.05$, assuming a 20 % within-subject coefficient of variation and a true geometric mean ratio of 1.00, $n = 24$ subjects are required in a 2×2 crossover design. The protocol enrols 28 to allow for 15 % dropout. Sensitivity analyses across CV $\in [15\%, 30\%]$ yield required n from 16 to 38, and across true ratio $\in [0.95, 1.05]$ from 24 to 33." Always state the margin and the assumed true effect.

5.18.8 Practical Tips

- The equivalence margin is a substantive and regulatory choice, not a statistical one; set it based on clinical or regulatory conventions (e.g., the 80–125 % bioequivalence margin) or on a pre-specified clinical-acceptance criterion, and justify the choice in the protocol.
- Power drops sharply as the anticipated true effect approaches the equivalence margin; always include sensitivity analysis over a plausible range of true effects to demonstrate that the design is robust to mild deviations from the central assumption.
- Crossover bioequivalence trials use within-subject variance (CV_w), which is typically smaller than between-subject variance; plan accordingly with the appropriate within-subject CV from pilot or literature.
- Asymmetric equivalence margins are supported by most TOST tools (`low_eqbound` $\neq$ `high_eqbound`) and are useful when only one direction of effect carries a clinical penalty; for example, a drug that is allowed to be more effective but not less effective than the reference.
- Non-inferiority testing is mathematically a one-sided TOST against a single non-inferiority margin; the required sample size is roughly half of a two-sided equivalence test for the same margin and assumptions, reflecting the simpler hypothesis.
- For highly variable drugs (within-subject CV above 30 %), reference-scaled bioequivalence widens the margin proportionally to the reference variability; standard fixed-margin TOST sample-size tools under-estimate the required n for these compounds.

5.18.9 R Packages Used

`PowerTOST` for canonical bioequivalence and equivalence sample-size and power across crossover and parallel-group designs; `TOSTER` for general TOST analysis and power on raw and standardised effect sizes; `MBESS` and `pwrss::pwrss.t.`

`2means()` for related precision and equivalence calculations; `bear` for end-to-end FDA-compliant bioequivalence workflow; `replicateBE` for replicate-design reference-scaled bioequivalence.

5.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.18.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.19 Power for ICC

5.19.1 Introduction

The intraclass correlation coefficient (ICC) is the standard reliability statistic for continuous ratings — agreement between raters on quantitative scales, test-retest stability, instrument repeatability across multiple measurements per subject. Sample-size planning for an ICC study depends on the number of subjects, the number of raters or replicate measurements per subject, the target ICC value to detect (or estimate), and whether the goal is hypothesis testing (rejecting a null ICC value) or precision (achieving a target confidence-interval width). Both questions admit closed-form solutions under standard assumptions, and explicit power planning is now expected for any prospective reliability study aiming for FDA, IVDR, or guideline acceptance.

5.19.2 Prerequisites

A working understanding of ICC variants — one-way random ICC(1,1), two-way random ICC(2,1), two-way mixed ICC(3,1), single-measurement vs average-measurement variants — and the distinction between hypothesis-testing and precision-based sample-size questions.

5.19.3 Theory

Following Walter, Eliasziw, and Donner (1998), the sample size for testing $H_0 : \rho \leq \rho_0$ against $H_1 : \rho \geq \rho_1$ with k raters or measurements per subject and target power $1 - \beta$ has a closed-form expression based on the F-distribution

under the variance-component decomposition. For precision-based planning, the approximate formula

$$n \approx \frac{(1-\rho^2)^2 \cdot 8 \cdot (k-1)}{\epsilon^2 \cdot k}$$

gives the number of subjects needed for a 95 % CI half-width of ϵ around the expected ICC ρ — useful when the goal is a tight reliability estimate rather than rejecting a specific null.

5.19.4 Assumptions

The outcome is continuous and approximately Normal, subjects are independent, and each subject is rated by the same k raters (or measured at k replicate occasions); the ICC variant assumed in the calculation matches the design (one-way random vs two-way random vs two-way mixed).

5.19.5 R Implementation

```r
library(ICC.Sample.Size)

calculateIccSampleSize(p = 0.8, p0 = 0.6, k = 2,
                       alpha = 0.05, tails = 2, power = 0.80)

k <- 2; rho <- 0.75; eps <- 0.1
n_prec <- (1 - rho^2)^2 * 8 * (k - 1) / (eps^2 * k)
ceiling(n_prec)
```

5.19.6 Output & Results

`calculateIccSampleSize()` returns the required number of subjects for the hypothesis test. For $\text{ICC}_1 = 0.8$ vs $\text{ICC}_0 = 0.6$ at $k = 2$ raters, $n \approx 84$. The precision-based calculation gives roughly 19 subjects for a 95 % CI half-width of 0.10 around an expected ICC of 0.75 — a much smaller sample because precision tolerates a wider range than rejecting a specific null.

5.19.7 Interpretation

A reporting sentence: "Each subject will be rated by two independent raters; $n = 84$ subjects are required for 80 % power to detect an ICC of 0.80 against a null hypothesis of ICC = 0.60 at two-sided $\alpha = 0.05$. The protocol enrols 100 subjects to allow for 15 % missing or unevaluable ratings. Sensitivity analyses show $n = 110$ for the more conservative null of 0.65 and $n = 65$ for $k = 3$ raters; the design report includes both alternatives in the supplement." Always state which ICC form and how many raters.

5.19.8 Practical Tips

- More raters per subject reduces the required number of subjects, but at linear cost in rater-hours; the trade-off depends on the relative cost and availability of subjects vs raters and is project-specific.
- Specify which ICC variant matches the design exactly — one-way random (ICC(1,1)) for fully crossed designs without rater identity, two-way random (ICC(2,1)) when raters are sampled from a population, two-way mixed (ICC(3,1)) when raters are fixed and chosen — because the sample-size formula and the analytic ICC differ across variants.
- Small n produces wide ICC confidence intervals; for reliability claims supporting clinical use, aim for a 95 % CI lower bound above 0.75 (good) or 0.90 (excellent), per Koo and Li (2016) recommendations.
- Sensitivity analysis varying the expected ρ across a realistic range is standard practice; protocols with a single point estimate of ρ are increasingly flagged by reviewers, who expect an honest acknowledgement of uncertainty.
- Published reliability studies historically have small samples and produce poorly characterised ICC estimates; planning conservatively (assuming ρ at the lower end of pilot CIs) protects against under-powered ICC validation.
- For agreement on categorical ratings, use Cohen's or Fleiss's kappa rather than ICC; the sample-size frameworks differ but the planning principles are analogous.

5.19.9 R Packages Used

`ICC.Sample.Size::calculateIccSampleSize()` for canonical hypothesis-test sample-size calculation; `irr::icc()` and `psych::ICC()` for ICC analysis after data collection; `ICC` for advanced ICC inference and CI methods; `simr` for simulation-based ICC power across complex designs; `MBESS::ci.reliability()` for confidence-interval-driven sample-size planning across reliability statistics.

5.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.19.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.20 Power for Linear Regression

5.20.1 Introduction

Power analysis for linear regression determines the sample size required to detect either an overall non-zero R^2 (via the omnibus F-test of the regression) or the incremental R^2 contributed by adding one or more predictors to a baseline model. Both questions are central to the design of observational and experimental studies that use multiple regression for inference: the omnibus power calculation tells you whether the model as a whole has a chance to be detected, and the incremental power calculation tells you whether a specific block of predictors of scientific interest will be detectable above and beyond control variables. Cohen's f^2 effect size unifies these calculations into a single non-central F-distribution framework.

5.20.2 Prerequisites

A working understanding of multiple linear regression, the relationship between R^2 and the omnibus F-statistic, and Cohen's f^2 as the standardised effect-size measure for regression.

5.20.3 Theory

Cohen's f^2 for the omnibus regression test is

$$f^2 = \frac{R^2}{1 - R^2},$$

and for the incremental contribution of a new block of predictors,

$$f^2 = \frac{R^2_{\text{full}} - R^2_{\text{reduced}}}{1 - R^2_{\text{full}}}.$$

Conventional benchmarks are 0.02 (small), 0.15 (medium), and 0.35 (large). The omnibus or partial F-test follows a non-central F-distribution under H_1 with non-centrality $\lambda = f^2(u + v + 1)$, where u is the numerator df (number of predictors tested) and v is the residual df. Power is the probability that this non-central F exceeds the critical value at α.

5.20.4 Assumptions

The standard linear-regression assumptions (linearity, homoscedasticity, Normal residuals, independent observations), the number of predictors in the full model and the incremental block are pre-specified, and the effect size f^2 is chosen on the basis of the smallest scientifically meaningful incremental R^2.

5.20.5 R Implementation

```r
library(pwr)

pwr.f2.test(u = 5, v = NULL, f2 = 0.15, sig.level = 0.05, power = 0.80)

pwr.f2.test(u = 2, v = NULL, f2 = 0.02, power = 0.80)
```

5.20.6 Output & Results

`pwr.f2.test()` returns the residual degrees of freedom v required to achieve target power; the total sample size is $n = v + u + 1$. For a 5-predictor model at medium effect $f^2 = 0.15$, $v = 85$ and $n = 91$; for a small incremental effect $f^2 = 0.02$ with 2 added predictors after 5 baseline predictors, $v = 489$ and $n = 497$ — illustrating how dramatically small incremental effects increase sample-size requirements.

5.20.7 Interpretation

A reporting sentence: "With 5 predictors in the regression and an anticipated $f^2 = 0.15$ (medium-effect convention; equivalent to $R^2 = 0.13$), a total sample of $n = 91$ is required to achieve 80 % power for the omnibus F-test at $\alpha = 0.05$. To detect an incremental $f^2 = 0.02$ from 2 additional predictors of primary interest after adjusting for the 5 baseline covariates, the protocol requires $n = 497$. The trial therefore enrols 550 participants to allow for 10 % attrition." Always state u, f^2, and the inflation factor.

5.20.8 Practical Tips

- Convert anticipated R^2 expectations to f^2 using the formulas above; the conversion is non-linear and a "small R^2" of 0.05 corresponds to $f^2 \approx 0.053$, while a "medium R^2" of 0.13 corresponds to $f^2 \approx 0.15$.
- Small incremental effects ($f^2 < 0.05$) require very large samples to detect; weigh the cost-benefit honestly and consider whether a smaller incremental effect is genuinely scientifically interesting.
- Interactions, polynomial terms, and dummy-coded categorical variables each add to the predictor count u and inflate the standard errors of individual coefficients; the sample-size calculation must reflect the actual model structure.
- Pre-specify the minimum detectable incremental effect of interest in the protocol; avoid post-hoc rationalisation of the chosen f^2 to match the achieved power, which is a form of HARKing.
- Include a 15–20 % buffer above the calculated n for residual non-Normality, minor violations of homoscedasticity, and missing data; the calculated n is the minimum, not a target.

- For prediction-focused studies (rather than inferential studies), consider using a learning-curve or repeated-cross-validation framework to determine the sample size required for stable out-of-sample R^2, which is a different question than power for the omnibus test.

5.20.9 R Packages Used

`pwr::pwr.f2.test()` for canonical f^2-based linear-regression power calculation; `WebPower::wp.regression()` for an alternative interface mirroring popular online calculators; `pwrss::pwrss.f.reg()` for extended regression-power tools; `Superpower` for ANOVA and factorial-design power simulation; `simr` for simulation-based power on regression with complex covariance structures.

5.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.20.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.21 Power for Logistic Regression

5.21.1 Introduction

Power analysis for logistic regression is less standardised than for linear models. Three approaches: (1) events-per-variable rule of thumb, (2) closed-form formulas for a single predictor, (3) simulation for complex models.

5.21.2 Prerequisites

Logistic regression, odds ratio, events-per-variable (EPV).

5.21.3 Theory

EPV rule (Peduzzi et al.): at least 10 events per predictor coefficient is a working minimum. Harrell's 20 EPV is safer.

Single-predictor formula (Demidenko 2007) uses $\log(OR)$ and the baseline event rate:

$$n = \frac{(z_{1-\alpha/2} + z_{1-\beta})^2}{p_0(1-p_0)\log^2(OR) \cdot \sigma_X^2/\pi(1-\pi)},$$

with simplifications for dichotomous X.

Simulation: specify the full data-generating process, fit the intended model, compute power across many replicates. More reliable than formulas for multiple predictors.

5.21.4 Assumptions

- Binary outcome.
- Logit-linear predictors.
- Sufficient separation of groups.

5.21.5 R Implementation

```
library(WebPower)

# Single dichotomous predictor: baseline event rate p0 = 0.2, OR = 1.5
wp.logistic(n = NULL, p0 = 0.2, p1 = 0.2 * 1.5 / (1 + 0.2 * (1.5 - 1)),
            alpha = 0.05, power = 0.80,
            family = "Bernoulli", parameter = NULL, alternative =
             ↪  "two.sided")

# Continuous predictor: detect OR = 1.5 per 1-SD increase
wp.logistic(n = NULL, p0 = 0.2, p1 = 0.2,
            alpha = 0.05, power = 0.80,
            family = "normal",
            alternative = "two.sided",
            parameter = log(1.5))

# EPV rule: 15 predictors, min events = 10*15 = 150
# If baseline event rate is 0.20: n = 150 / 0.20 = 750
```

5.21.6 Output & Results

Detecting a modest OR of 1.5 typically requires hundreds of observations; continuous predictors need somewhat less n than dichotomous, given comparable effect magnitude.

5.21.7 Interpretation

"To detect an OR of 1.5 per unit of the standardised predictor at 80 % power and two-sided $\alpha = 0.05$, 250 participants are required (assuming 20 % baseline

event rate)."

5.21.8 Practical Tips

- Prefer simulation over formulas when predictors are correlated or the model is complex.
- Rare-event outcomes dramatically inflate required n; power is approximately symmetric around $p_0 = 0.5$.
- For ordinal or multinomial logistic, scale up by roughly $k - 1$ (number of contrasts).
- Penalised logistic (ridge, Firth) handles small samples with separation.
- Missing data and measurement error further increase required n.

5.21.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.21.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.22 Power for the Log-Rank Test

5.22.1 Introduction

The log-rank test is the standard non-parametric test of equality of two (or more) survival functions and the most widely used inference tool in survival analysis. A central feature of its power calculation distinguishes it from sample-size analyses for t-tests, ANOVAs, or proportion comparisons: power depends on the total number of observed events across the two groups, not on the sample size directly. Event-based sample-size planning therefore separates the recruitment question (how many subjects to enrol) from the inferential question (how many events to observe), and links them through assumptions about event rates, accrual time, and follow-up duration. This decoupling is essential to realistic planning of survival trials.

5.22.2 Prerequisites

A working understanding of the log-rank test, the Kaplan-Meier survival estimator, the proportional-hazards assumption, and the relationship between hazard ratio, event rate, and follow-up time.

5.22.3 Theory

Under the alternative hypothesis HR $= \theta$ with equal allocation, the required total number of events is

$$D = \frac{4(z_{1-\alpha/2} + z_{1-\beta})^2}{\log^2 \theta}.$$

For unequal allocation with arm fractions π_1, π_2 summing to one, the multiplier 4 is replaced by $1/(\pi_1 \pi_2)$. The required sample size n depends on the event probability over the trial duration, which in turn depends on the underlying hazard rates, the planned accrual period, and the post-accrual follow-up period. The relationship is captured in standard formulas (Schoenfeld, Freedman, or Lakatos) and implemented in dedicated power-analysis packages.

5.22.4 Assumptions

The proportional-hazards assumption holds (approximately) over the trial duration, censoring is independent and non-informative, the accrual pattern and follow-up duration are pre-specified, and the underlying hazard rates are reasonably well-estimated from prior data.

5.22.5 R Implementation

```
library(powerSurvEpi); library(gsDesign)

D <- 4 * (qnorm(0.975) + qnorm(0.80))^2 / log(0.65)^2
D

ssizeCT.default(power = 0.80, k = 1,
                pE = 0.40, pC = 0.55,
                RR = 0.65, alpha = 0.05)

gs <- gsDesign(k = 3, test.type = 2, alpha = 0.025, beta = 0.20)
gsBoundSummary(gs)
```

5.22.6 Output & Results

The closed-form calculation gives roughly 88 events required to detect HR $=$ 0.65 with 80 % power at two-sided $\alpha = 0.05$. With assumed event rates of 40 % in the experimental arm and 55 % in the control arm over the planned follow-up window, this corresponds to roughly 185 subjects (just under 100 per arm). The group-sequential design extends this with multiple looks and appropriate alpha-spending.

5.22.7 Interpretation

A reporting sentence: "The trial requires 88 events to detect a hazard ratio of 0.65 with 80 % power at two-sided $\alpha = 0.05$ under the log-rank test. Assuming 24 months of uniform accrual and 12 months of additional follow-up, with expected event rates of 55 % (control) and 40 % (treatment) over the trial duration, the planned enrolment is 180 subjects (90 per arm). The protocol triggers the primary analysis when 88 events have been observed, regardless of calendar time, ensuring the planned power is achieved." Always state the event-driven analysis trigger.

5.22.8 Practical Tips

- Power depends on the total number of events observed, not on the number of subjects enrolled; plan accrual and follow-up to deliver the required event count, and trigger the primary analysis when that count is reached rather than at a fixed calendar time. This is why survival trials report "event-driven" stopping rules.
- Non-proportional hazards (delayed effects, crossing survival curves, time-varying treatment effects) reduce the log-rank test's power; Fleming-Harrington weighted log-rank tests improve detection of early or late differences and can be pre-specified for trials where non-proportionality is expected.
- For interim monitoring, use alpha-spending group-sequential designs (`gsDesign`, `rpact`) with information-fraction-based analysis timing — typically every $D/4$ events for a four-stage design — rather than fixed calendar-time analyses.
- Report the assumed accrual pattern and expected follow-up duration explicitly; they determine the event yield from a given enrolment target, and trial outcomes vary substantially when these assumptions are wrong.
- For competing risks, the log-rank test is not the appropriate inferential tool; use the Fine-Gray subdistribution-hazard test and its corresponding power-analysis tools (`cmprsk`, `crrSC::power.crr()`).
- Sensitivity analysis over a range of plausible hazard ratios is standard; protocols with a single point HR are increasingly flagged by reviewers, who expect a defensible bracket.

5.22.9 R Packages Used

`powerSurvEpi::ssizeCT.default()` and `powerSurvEpi::powerCT.default()` for canonical log-rank sample-size and power calculations; `gsDesign` and `rpact` for group-sequential survival-trial design with alpha-spending; `nph` for non-proportional-hazards weighted log-rank power; `Hmisc::cpower()` for an alternative interface; `survival::survdiff()` for log-rank analysis after data collection.

5.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.22.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.23 Power for McNemar's Test

5.23.1 Introduction

Power for McNemar's test depends on the rate of discordant pairs, not on the overall sample size alone. The concordant pairs contribute nothing to the test, so studies with high agreement need proportionally more pairs.

5.23.2 Prerequisites

McNemar's test, paired binary data.

5.23.3 Theory

Let p_{10} = probability of (+, -) pairs and p_{01} = probability of (-, +). The null hypothesis is $p_{10} = p_{01}$. Under H_1, the total proportion of discordant pairs is $p_{\text{disc}} = p_{10} + p_{01}$; the odds ratio of a "+ on test 1" given discordance is p_{10}/p_{01}.

For a two-sided test at α and power $1 - \beta$:

$$n \approx \frac{(z_{1-\alpha/2} + z_{1-\beta})^2}{p_{\text{disc}} \cdot \left(\frac{p_{10}-p_{01}}{p_{10}+p_{01}}\right)^2}.$$

As discordance drops, n grows rapidly.

5.23.4 Assumptions

- Independent paired observations.
- Pre-specified p_{10}, p_{01} from pilot.

5.23.5 R Implementation

```r
library(pwrss)

# Expected p10 = 0.15, p01 = 0.05, alpha = 0.05, power = 0.80
pwrss.z.mcnemar(p10 = 0.15, p01 = 0.05,
                alpha = 0.05, power = 0.80)

# Manual calculation
p10 <- 0.15; p01 <- 0.05
p_disc <- p10 + p01
OR <- (p10 - p01) / (p10 + p01)
n_manual <- (qnorm(0.975) + qnorm(0.80))^2 / (p_disc * OR^2)
n_manual
```

5.23.6 Output & Results

$n \approx 79$ pairs required. If discordance is lower (say $p_{10} = 0.10$, $p_{01} = 0.05$), required n roughly doubles.

5.23.7 Interpretation

"With an expected proportion of (+, -) pairs of 0.15 and (-, +) pairs of 0.05, McNemar's test requires 79 paired observations for 80 % power at two-sided $\alpha = 0.05$."

5.23.8 Practical Tips

- Plan for the total sample, not just discordant pairs; concordant pairs are expected but non-informative.
- The formula is sensitive to the assumed discordance rates; sensitivity analysis is essential.
- For very high agreement (concordant pairs » discordant), large samples are needed; consider redesigning the comparison.
- Exact McNemar is more conservative in small samples; simulate if exactness matters.
- Extension to Bowker or Stuart-Maxwell for multi-category paired data requires simulation-based power.

5.23.9 Reporting

Always report the assumed discordance rates alongside the resulting sample size. Reviewers and ethics committees expect to see how the number of pairs was derived from the marginal probabilities, because the same total sample can yield very different power depending on the split between p_{10} and p_{01}. Where the pilot estimate of discordance is uncertain, present a small grid of n across plausible

values rather than a single point estimate, and state which value drove the final target. If a sequential design is anticipated, note that the formula above is for a single fixed analysis and that group-sequential or adaptive variants require separate boundary calculations.

5.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.23.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.24 Power for Non-Inferiority Trials

5.24.1 Introduction

Non-inferiority trials aim to establish that a new treatment is "not meaningfully worse" than an active comparator. The non-inferiority margin is a pre-specified loss that defines the boundary. Power calculations use a one-sided test at the margin.

5.24.2 Prerequisites

Equivalence testing, confidence intervals.

5.24.3 Theory

For continuous outcomes with true difference δ (new minus reference), margin $-\Delta$ (loss), and pooled SD σ, the non-inferiority null is $\mu_{\text{new}} - \mu_{\text{ref}} \leq -\Delta$. Power at α one-sided is

$$1 - \beta = \Phi\left(\frac{\delta + \Delta}{\sigma\sqrt{2/n}} - z_{1-\alpha}\right).$$

For binary outcomes (proportions), analogous formulas using Cohen's h or risk-difference variance apply.

Sample size increases with tighter margin, larger allowed effect, or reduced power.

5.24.4 Assumptions

- Pre-specified non-inferiority margin based on regulatory or clinical guidance.
- Assumed true δ (often zero, assuming equivalence).
- Normal or large-sample approximations.

5.24.5 R Implementation

```r
library(gsDesign)

# Continuous outcome: margin -5 units, true delta = 0, sigma = 15
# One-sided alpha = 0.025, power = 0.80
delta  <- 0; margin <- 5; sigma <- 15; alpha <- 0.025; beta <- 0.20
n_per_arm <- 2 * sigma^2 * (qnorm(1 - alpha) + qnorm(1 - beta))^2 /
 ↪  (delta + margin)^2
n_per_arm

# Binary outcome: reference rate 70% cure, margin -10%, true new = 70%
pC <- 0.70; pE <- 0.70; margin_prop <- 0.10
# Farrington-Manning or score-based calculations are standard; simplified
 ↪  Wald here:
p_bar <- (pC + pE) / 2
n_bin <- (qnorm(1 - alpha) + qnorm(1 - beta))^2 *
         (pC*(1-pC) + pE*(1-pE)) / (pE - pC + margin_prop)^2
n_bin

# Group-sequential non-inferiority design
ni <- gsDesign(k = 2, test.type = 1, alpha = 0.025, beta = 0.20,
               sfu = "OF")
gsBoundSummary(ni)
```

5.24.6 Output & Results

Continuous: $n \approx 140$ per arm. Binary: $n \approx 196$ per arm.

5.24.7 Interpretation

"To establish non-inferiority with a 5-unit margin, assuming a true zero difference and SD 15, at 80 % power and one-sided $\alpha = 0.025$, 140 patients per arm are required."

5.24.8 Practical Tips

- Margin selection is the most important decision in non-inferiority design; regulators require explicit justification.

- Use one-sided $\alpha = 0.025$ to keep consistency with two-sided 0.05 superiority conventions.
- Report per-protocol analysis alongside intention-to-treat; dilution effects in ITT make non-inferiority easier to show artificially.
- Include interim futility stops for ethical reasons.
- Conservative margins and high power requirements inflate n quickly.

5.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.24.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.25 Power for One-Proportion Test

5.25.1 Introduction

Power analysis for the one-proportion test computes the sample size required to detect a departure from a null proportion p_0 with adequate probability. The one-proportion test arises constantly in clinical research — testing whether a single-arm response rate exceeds a historical control benchmark, whether a quality-control defect rate departs from a tolerance threshold, whether a survey-derived proportion differs from a reference. Two complementary approaches are widely used: the Normal-approximation power formula, which gives clean closed-form expressions in terms of Cohen's h effect size, and exact-binomial power, which enumerates the binomial distribution and is preferred for small samples or extreme proportions where the Normal approximation is unreliable.

5.25.2 Prerequisites

A working understanding of the one-proportion test, the binomial distribution, the Normal approximation to the binomial, and Cohen's h as the standardised effect-size measure for proportion tests.

5.25.3 Theory

For the test of $H_0 : p = p_0$ versus $H_1 : p = p_1$, Cohen's standardised proportion-difference effect size is

$$h = 2 \arcsin \sqrt{p_1} - 2 \arcsin \sqrt{p_0},$$

which arc-sine transforms each proportion onto a scale where its variance is approximately constant. The Normal-approximation sample-size formula at two-sided α and power $1 - \beta$ is

$$n \approx \left(\frac{z_{1-\alpha/2} + z_{1-\beta}}{h} \right)^2 .$$

Cohen's conventional benchmarks for h are 0.20 (small), 0.50 (medium), and 0.80 (large). Exact binomial power evaluates the binomial CDF under p_1 and sums probabilities in the rejection region defined by the binomial CDF under p_0.

5.25.4 Assumptions

Trials are independent Bernoulli with constant success probability; for the Normal-approximation formula, $np_0 \geq 10$ and $n(1 - p_0) \geq 10$ to ensure the approximation is adequate; for exact binomial power these conditions are not needed.

5.25.5 R Implementation

```r
library(pwr)

h <- ES.h(p1 = 0.40, p2 = 0.25)
pwr.p.test(h = h, sig.level = 0.05, power = 0.80, alternative =
 ↪  "two.sided")

exact_binom_power <- function(n, p0, p1, alpha = 0.05) {
  k_reject <- sum(dbinom(0:n, n, p0) <= alpha)
  crit_lo <- qbinom(alpha/2, n, p0)
  crit_hi <- qbinom(1 - alpha/2, n, p0)
  power_lo <- pbinom(crit_lo - 1, n, p1)
  power_hi <- 1 - pbinom(crit_hi, n, p1)
  power_lo + power_hi
}
exact_binom_power(n = 80, p0 = 0.25, p1 = 0.40)
```

5.25.6 Output & Results

`pwr.p.test()` returns the sample size required under the Normal-approximation formula; the custom exact-binomial function then verifies the calculation by direct enumeration. For the example, the Normal approximation gives $n = 82$

and the exact calculation confirms 80 % power at $n = 80$ — close agreement that is typical when the Normal-approximation conditions are satisfied.

5.25.7 Interpretation

A reporting sentence: "To detect a response rate of 40 % against a historical-control null of 25 % with 80 % power at two-sided $\alpha = 0.05$, $n = 82$ participants are required (Normal-approximation calculation, Cohen's $h = 0.32$). Exact binomial power at $n = 80$ is 80 %, confirming the Normal-approximation result. The protocol enrols 90 to allow for 10 % attrition. Sensitivity analyses across $p_1 \in [0.35, 0.45]$ are reported in the supplement." Always report sensitivity over plausible p_1.

5.25.8 Practical Tips

- For small n (typically < 30) or extreme proportions (very close to 0 or 1), prefer exact binomial power calculations to the Normal approximation; the approximation is unreliable in these regimes and can give misleading sample-size recommendations.
- Cohen's h ranges in $[0, \pi]$ with conventional benchmarks 0.20 (small), 0.50 (medium), and 0.80 (large); these are guidelines for translating effect-size language into sample-size calculations and should be tied to substantive clinical or scientific meaning.
- One-sided tests require less sample size than two-sided tests if the direction of effect is pre-specified and scientifically justified; the trade-off is that a result in the unexpected direction cannot be reported as significant.
- Sensitivity analysis over a range of plausible p_1 values is standard; protocols with a single point estimate of p_1 are increasingly flagged by reviewers, who expect a defensible bracket and a justification of the lower bound used for the sample-size determination.
- For p_0 near 0.5, the test is approximately symmetric and the variance is maximised, so power per unit of n is at its maximum; for p_0 near 0 or 1 the variance is small and the test can be very efficient at detecting small absolute differences.
- Continuity-correction options (Yates correction) marginally affect Normal-approximation power calculations; modern software defaults vary, so check what your sample-size tool is computing.

5.25.9 R Packages Used

`pwr::pwr.p.test()` and `pwr::ES.h()` for canonical Cohen's h Normal-approximation calculations; `binom::power.binom()` for exact binomial sample-size and power; `Hmisc::bpower()` for an alternative implementation; `MESS::power_prop_test()` for fast exact and approximate calculations; `Superpower` and `simr` for simulation-based extensions.

5.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.25.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.26 Power for One-Sample t-Test

5.26.1 Introduction

Power analysis for the one-sample t-test determines the sample size required to detect a specified departure of the population mean from a reference value with a given probability, or equivalently the power achievable at a fixed sample size. The one-sample t-test arises in many practical contexts: validating a quality-control measurement against a target, comparing a single-arm pilot study against a historical benchmark, testing whether a within-subject change differs from zero. In every case, the planning question is the same — given the smallest effect of scientific interest and the assumed within-population standard deviation, how many observations are needed for an adequately-powered test? Pre-specifying the calculation in the protocol is now standard for any prospective study.

5.26.2 Prerequisites

A working understanding of the one-sample t-test, the concept of statistical power and type-II error, and Cohen's d as the standardised effect-size measure for one-mean tests.

5.26.3 Theory

Under the alternative hypothesis $H_1 : \mu = \mu_1 \neq \mu_0$, the t-statistic follows a non-central t-distribution with $n-1$ degrees of freedom and non-centrality parameter

$$\lambda = \frac{\mu_1 - \mu_0}{\sigma/\sqrt{n}} = d\sqrt{n},$$

where $d = (\mu_1 - \mu_0)/\sigma$ is Cohen's standardised effect. Power is the probability $P(|T| > t_{1-\alpha/2,n-1})$ computed under the non-central t-distribution. Closed-form

sample-size solutions invert this expression to produce the smallest n achieving target power against a specified d, α, and one-sided / two-sided test direction.

5.26.4 Assumptions

The population is approximately Normal or the sample size is large enough for the central limit theorem to apply, the within-population standard deviation σ is reasonably known (or has been estimated from pilot data with adequate precision), and the effect size d is pre-specified — usually as the smallest clinically or scientifically meaningful difference rather than the expected value.

5.26.5 R Implementation

```r
library(pwr)

pwr.t.test(d = 0.4, sig.level = 0.05, power = 0.80, type = "one.sample")

pwr.t.test(n = 25, sig.level = 0.05, power = 0.80, type = "one.sample")

n_grid <- seq(10, 100, by = 5)
pw <- sapply(n_grid, function(n)
  pwr.t.test(n = n, d = 0.4, type = "one.sample")$power)
plot(n_grid, pw, type = "l", lwd = 2, col = "#2A9D8F",
     xlab = "n", ylab = "Power", main = "One-sample t-test, d = 0.4")
abline(h = 0.80, lty = 2)
```

5.26.6 Output & Results

`pwr.t.test()` returns the sample size required to achieve target power against a specified d, the achievable power for a given n, or the minimum detectable effect size for a given n and target power. Plotting the power curve against n for the chosen d visualises the trade-off and is a useful supplement to the single-point calculation.

5.26.7 Interpretation

A reporting sentence: "A sample of 51 participants is required to detect a standardised effect of $d = 0.4$ with 80 % power at $\alpha = 0.05$ (two-sided) using a one-sample t-test against the reference value 10. With the planned sample of 25 participants, the minimum detectable effect at 80 % power would be $d = 0.58$, which exceeds the smallest clinically meaningful effect of 0.4 and would leave the study underpowered. The protocol therefore stipulates 51 participants, with 60 enrolled to allow for 15 % attrition." Always state d, α, power, and the attrition margin.

5.26.8 Practical Tips

- The one-sample t-test power formula applies equally to the paired t-test if the effect size d is computed using the within-subject difference SD rather than the between-subject SD; this is the most common practical use of the calculation.
- For very small effect sizes, sample sizes grow rapidly: halving d quadruples the required n because power scales with $d\sqrt{n}$. A clear-headed choice of the smallest clinically meaningful d is therefore essential, and pilot-data d estimates should be treated with appropriate uncertainty.
- One-sided power at α approximately equals two-sided power at 2α when the true effect is in the hypothesised direction; the rule of thumb is useful for quick comparisons but exact non-central t computation is preferred for the protocol calculation.
- Choose d on the basis of the minimum clinically important difference (MCID) or smallest scientifically meaningful effect, not the expected effect from pilot data; powering for the expected effect leads to under-powered trials when the truth is more modest.
- Inflate the calculated sample size for expected attrition, dropout, or non-evaluable observations; a 15–20 % inflation is typical for clinical studies, with field-specific norms varying.
- Distinguish achievable power (post-hoc, computed from observed effect size) from planned power (a priori); only planned power is methodologically valid, and "post-hoc power" calculations are widely discouraged because they are mathematically determined by the observed p-value.

5.26.9 R Packages Used

`pwr::pwr.t.test()` for the canonical one-sample and paired t-test power calculation; `WebPower::wp.t()` for an alternative interface mirroring popular online calculators; `Superpower` for ANOVA and factorial-design power simulation; `simr` for power analysis via simulation when classical formulas do not apply; `MESS::power_t_test()` as a fast alternative.

5.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.26.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.27 Power for Paired t-Test

5.27.1 Introduction

Paired designs achieve their statistical power gain by cancelling between-subject variability from the treatment contrast. Because each subject acts as their own control, the between-subject component of variance — typically the largest source of noise in clinical-pharmacology, behavioural, and clinical-laboratory studies — drops out of the within-subject difference. The result is that paired designs are substantially more powerful than independent-groups designs of equal total sample size, with the benefit growing as the within-subject correlation increases. Power analysis for the paired t-test must therefore explicitly account for this correlation; ignoring it under-uses the design's main analytic advantage and inflates the planned sample size unnecessarily.

5.27.2 Prerequisites

A working understanding of the paired t-test, the relationship between within-subject correlation ρ and difference standard deviation σ_D, and the standardised effect-size measures Cohen's d (between-subject) and d_z (within-subject paired).

5.27.3 Theory

For paired differences $D_i = X_i - Y_i$ with within-subject correlation ρ, the difference standard deviation is

$$\sigma_D = \sigma\sqrt{2(1-\rho)},$$

and the appropriate paired-design effect size is

$$d_z = \frac{\mu_D}{\sigma_D}.$$

Higher within-subject correlation reduces σ_D, inflating d_z for a fixed raw effect, and dramatically reducing the required sample size. With $\rho = 0$ the paired design reduces to the independent-groups case (at half the per-condition sample size); with $\rho \to 1$ the paired design approaches infinite efficiency.

5.27.4 Assumptions

The design genuinely produces paired observations (each subject contributes one value per condition), the differences are approximately Normal (or the sample is large enough for the central limit theorem), and the within-subject correlation is reasonably known from pilot data or literature.

5.27.5 R Implementation

```r
library(pwr)

pwr.t.test(d = 0.4, sig.level = 0.05, power = 0.80, type = "paired")

sigma_diff <- function(sigma, rho) sigma * sqrt(2 * (1 - rho))
d_z_from_d <- function(d, rho) d / sqrt(2 * (1 - rho))

d_z_from_d(d = 0.5, rho = 0.7)

pwr.t.test(d = 0.5, power = 0.80, type = "two.sample")$n
pwr.t.test(d = d_z_from_d(0.5, 0.7), power = 0.80, type = "paired")$n
```

5.27.6 Output & Results

The script reports the required number of pairs at $d_z = 0.4$, the conversion from raw d to paired d_z at correlation 0.7, and the side-by-side sample-size comparison between independent-groups and paired designs for the same raw effect. The paired design needs roughly 21 pairs (42 observations) versus 128 observations for the independent comparison — a six-fold efficiency gain at $\rho = 0.7$.

5.27.7 Interpretation

A reporting sentence: "Assuming a within-subject correlation of 0.7 (estimated from pilot data) and a raw mean difference of 0.5 SD, the paired design requires 21 paired observations for 80 % power at $\alpha = 0.05$ (two-sided), corresponding to a paired effect size $d_z = 0.65$. The same raw effect would require 128 observations (64 per arm) in an independent-groups design — a 6-fold efficiency gain. The protocol therefore enrols 25 subjects to accommodate up to 15 % attrition." Always report the assumed ρ and the efficiency gain.

5.27.8 Practical Tips

- Pre-specify the expected within-subject correlation ρ from pilot data or published literature, and conduct a sensitivity analysis over a plausible range — typically $\rho \in [0.3, 0.8]$. The sample-size calculation is sensitive to this assumption, and protocols should defend the chosen value.
- When $\rho = 0$ the paired design reduces to the independent-groups case at half the per-condition sample size; the paired design is genuinely advantageous only when within-subject correlation is meaningfully positive, which is almost always the case in repeated-measures contexts.
- Report the effect size in paired d_z form (where the denominator is the difference SD) rather than two-sample d form (where the denominator is the within-group SD); the two are commonly confused and lead to apparent disagreements in power calculations across software packages.

- Missing one member of a pair drops the entire pair from the analysis; this attrition pattern is more wasteful than in independent-groups designs and should be reflected in the sample-size inflation factor.
- For more than two repeated measures (three time points, multiple conditions per subject), use repeated-measures ANOVA or mixed-effects power calculations rather than treating the design as a series of pairwise paired tests; the multi-condition analysis is more efficient and avoids multiplicity adjustments.
- Pilot estimates of ρ have substantial uncertainty in small samples; using the lower end of a CI on ρ for power planning is a conservative approach that protects against an over-optimistic point estimate.

5.27.9 R Packages Used

`pwr::pwr.t.test(type="paired")` for the canonical paired-design power calculation in d_z units; `WebPower::wp.t()` with paired specification for an alternative interface; `Superpower` for paired-design ANOVA and factorial extensions; `simr` for simulation-based power analysis when the design extends to mixed-effects models with random slopes; `BUCSS` for bias- and uncertainty-corrected sample-size estimation when pilot data are limited.

5.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.27.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.28 Power for Repeated-Measures ANOVA

5.28.1 Introduction

Repeated-measures designs achieve substantial statistical-power gains over independent-groups designs by exploiting the within-subject correlation across the multiple measurements per subject. Each subject contributes information at every time point or condition, so the within-subject component of variance — typically the smaller of the two variance components — drives the sensitivity of the test. Power calculations for repeated-measures ANOVA must therefore explicitly account for the within-subject correlation; ignoring it leads to

substantially over-estimated sample-size requirements and wastes resources, while assuming an unrealistically high correlation under-powers the trial when reality intervenes.

5.28.2 Prerequisites

A working understanding of repeated-measures ANOVA, the sphericity assumption and its diagnostics, Cohen's f as the standardised effect-size measure for ANOVA, and the role of within-subject correlation in design efficiency.

5.28.3 Theory

For a within-subjects design with k measurements per subject, effect size f, and average within-subject correlation ρ, the non-centrality parameter of the omnibus F-test is approximately

$$\lambda = n f^2 \frac{k}{1-\rho},$$

where n is the number of subjects. Power is the tail probability of the non-central F distribution evaluated at the conventional critical value. Higher ρ inflates λ and reduces required sample size; near-zero ρ gives little efficiency advantage over an equivalent independent-groups ANOVA. The Cohen-conventional benchmarks for f are 0.10 (small), 0.25 (medium), and 0.40 (large).

5.28.4 Assumptions

Sphericity holds (equal pairwise variances of differences across time points), residuals are approximately Normal, and the within-subject correlation is reasonably known from pilot data or literature. Sphericity violations require Greenhouse-Geisser or Huynh-Feldt corrections, which slightly inflate the required sample size.

5.28.5 R Implementation

```
library(WebPower)

wp.rmanova(f = 0.25, ng = 1, nm = 4, nscor = 1,
           alpha = 0.05, power = 0.80, type = 1)

rho_grid <- seq(0.2, 0.9, by = 0.1)
ns <- sapply(rho_grid, function(rho) {
  wp.rmanova(f = 0.25, ng = 1, nm = 4, nscor = rho,
             alpha = 0.05, power = 0.80, type = 1)$n
})
data.frame(rho = rho_grid, n = round(ns))
```

5.28.6 Output & Results

`wp.rmanova()` returns the required sample size at specified f, k, ρ, and target power. The sensitivity table shows required n falling sharply as within-subject correlation increases — from 54 subjects at $\rho = 0.2$ to only 8 at $\rho = 0.9$. This dramatic dependence makes pre-specification of ρ (and a defensible sensitivity range) essential to honest sample-size planning.

5.28.7 Interpretation

A reporting sentence: "For a within-subjects design with four time points, a medium effect ($f = 0.25$), and assumed within-subject correlation 0.5 from pilot data, $n = 34$ subjects provide 80 % power at $\alpha = 0.05$ for the omnibus F-test under the sphericity assumption. Sensitivity analyses across $\rho \in [0.30, 0.70]$ yield required n from 47 to 23 respectively; the protocol enrols 50 to accommodate the most conservative scenario plus 15 % attrition." Always state ρ and report sensitivity.

5.28.8 Practical Tips

- If sphericity is suspected to fail, inflate the calculated sample size by 10–30 % to compensate for the Greenhouse-Geisser or Huynh-Feldt corrections that will be applied at analysis time; the corrections reduce effective degrees of freedom and therefore reduce achievable power.
- For mixed designs combining between-subjects and within-subjects factors, use `wp.rmanova(type=2)` or `type=3` for the appropriate stratum, or use simulation for non-standard designs; classical formulas can be brittle in mixed designs with complex interaction structures.
- Higher numbers of repeated measurements per subject inflate power dramatically because each subject contributes more information; planners often face a trade-off between recruiting more subjects (expensive, slow) and measuring each subject more times (cheap, fast).
- When the primary scientific question is a specific contrast (a linear trend over time, a difference between two conditions), a contrast-specific power calculation typically yields more power than the omnibus F-test and should be used in preference; the omnibus test diffuses power across all interactions.
- For non-Normal outcomes, unbalanced data, or designs with informative dropout, simulation via a linear mixed-effects model (using `simr::powerSim()`) is the gold-standard approach; classical formulas assume a balanced complete-data design and Normality.
- Pilot estimates of ρ have substantial uncertainty in small samples; use the lower end of a reasonable CI on ρ for protective sample-size planning rather than the point estimate.

5.28.9 R Packages Used

`WebPower::wp.rmanova()` for canonical repeated-measures-ANOVA power across one-way and mixed designs; `pwr` family of functions for related ANOVA-power tools; `Superpower::ANOVA_power()` for ANOVA-design power simulation including repeated-measures structures; `simr::powerSim()` for simulation-based power on linear mixed-effects models with arbitrary covariance structure; `afex::aov_car()` for fitting the underlying ANOVA model with sphericity correction.

5.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.28.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.29 Power for Stepped-Wedge Trials

5.29.1 Introduction

Stepped-wedge cluster-randomised trials (SW-CRT) progressively introduce an intervention across clusters over time until every cluster receives it. The design is efficient when the intervention is expected to help and must be rolled out anyway. Power calculations use the Hussey-Hughes (2007) formula.

5.29.2 Prerequisites

Cluster-RCT, ICC, time effects.

5.29.3 Theory

For I clusters and J time periods (with one cluster per step crossing over), the Hussey-Hughes variance of the treatment effect is complex and depends on ICC, cluster-period correlation, cluster size, and total steps. A key insight: adding time periods increases efficiency because each cluster contributes both control and intervention observations.

Software (`swCRTdesign`, `SWSamp`) computes the needed number of clusters given ICC, cluster-autocorrelation, cluster size, and effect.

5.29.4 Assumptions

- Fixed time periods common to all clusters.
- No lag from randomisation to intervention effect (or explicit lag period).
- Underlying secular time trend appropriately modelled.
- ICC and cluster-autocorrelation pre-specified.

5.29.5 R Implementation

```r
library(swCRTdesign)

# Design: 8 clusters, 6 time periods, cluster size 20, ICC = 0.05
swDsn <- swDsn(clusters = c(rep(1, 7), 2))
swDsn

# Power for effect size 0.3, sigma = 1
swPwr(design = swDsn,
      distn = "gaussian",
      n = 20,
      mu0 = 0, mu1 = 0.3,
      tau = 0.05, sigma = 1,
      alpha = 0.05,
      retDATA = FALSE)
```

5.29.6 Output & Results

Power around 0.85 for 8 clusters x 6 periods x 20 subjects/cluster at effect 0.3, ICC 0.05. The exact number depends heavily on the assumed autocorrelation.

5.29.7 Interpretation

"With 8 clusters, 6 time periods, cluster size 20, ICC 0.05, and an assumed intervention effect of 0.3 SD, the stepped-wedge trial achieves 85 % power at $\alpha = 0.05$."

5.29.8 Practical Tips

- Stepped-wedge designs trade statistical efficiency for practical feasibility; a parallel CRT is usually more powerful at equal total observations.
- The cluster-period autocorrelation is a second nuisance parameter often under-specified; include sensitivity analysis.
- Longer follow-up and more time periods dramatically improve efficiency.
- Analyse with mixed models that include cluster, time-period, and (usually) cluster-by-period random effects.
- Published tools (`swCRTdesign`, `SWSamp`, online calculators from Hemming and Girling) differ slightly; cross-check.

5.29.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.29.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.30 Power for Two-Proportion Test

5.30.1 Introduction

Comparing proportions between two independent groups is one of the most common designs in clinical research, public-health epidemiology, and quality-control. Whenever the primary outcome is binary — event vs no event, response vs non-response, success vs failure — the trial's sample-size calculation reduces to a two-proportion power analysis. Several closely-related approaches are widely used: Cohen's arcsine-transformed effect size h that maps cleanly to a Normal-approximation formula, direct sample-size formulas in the original-proportion scale (e.g., Kelsey's formula with continuity correction), and exact Fisher-based methods for rare events. The choice depends on the expected event rates, the desired allocation ratio, and the regulatory or methodological context.

5.30.2 Prerequisites

A working understanding of the two-proportion test, the arcsine variance-stabilising transformation, and Cohen's h as the standardised effect-size measure for proportion-difference designs.

5.30.3 Theory

Cohen's h is the arcsine-transformed proportion difference

$$h = 2 \arcsin \sqrt{p_1} - 2 \arcsin \sqrt{p_2}.$$

For balanced two-arm designs, the sample size per group at two-sided α and power $1 - \beta$ is approximately

$$n \approx 2 \left(\frac{z_{1-\alpha/2} + z_{1-\beta}}{h} \right)^2 \text{ per group.}$$

For unequal allocation with ratio $k = n_2/n_1$, the total sample size grows by a factor $(1+1/k)(1+k)/4$ relative to the balanced minimum. Cohen's benchmarks for h are 0.20 (small), 0.50 (medium), and 0.80 (large).

5.30.4 Assumptions

The two groups are independent, the standard Normal-approximation conditions $np \geq 10$ and $n(1-p) \geq 10$ are satisfied for each group, and the proportions under the alternative hypothesis are pre-specified.

5.30.5 R Implementation

```r
library(pwr); library(Hmisc)

pwr.2p.test(h = ES.h(0.20, 0.10), sig.level = 0.05, power = 0.80)

bsamsize(p1 = 0.20, p2 = 0.10, alpha = 0.05, power = 0.80)

pwr.2p2n.test(h = ES.h(0.20, 0.10), n1 = 100, sig.level = 0.05)
n1_fix <- 100
n2_vals <- seq(40, 300, 10)
pw <- sapply(n2_vals, function(n2)
  pwr.2p2n.test(h = ES.h(0.20, 0.10), n1 = n1_fix, n2 = n2)$power)
n2_vals[which(pw >= 0.80)[1]]
```

5.30.6 Output & Results

`pwr.2p.test()` returns the per-group sample size at specified h, α, and power; `Hmisc::bsamsize()` provides the same calculation directly in the original-proportion scale. For $p_1 = 0.20$ vs $p_2 = 0.10$ at 80 % power, $n \approx 199$ per group (398 total). The unequal-allocation analysis shows that fixing $n_1 = 100$ requires roughly $n_2 = 280$ to maintain 80 % power, illustrating the inefficiency of imbalanced designs.

5.30.7 Interpretation

A reporting sentence: "To detect a difference in event rates from 10 % (historical control) to 20 % (active treatment) at $\alpha = 0.05$ two-sided with 80 % power, 199 patients per arm are required (398 total) under a 1:1 allocation. The protocol enrols 440 patients to allow for 10 % attrition. Sensitivity analyses across $p_2 \in [0.15, 0.25]$ yield required per-arm n from 392 to 105, bracketing the assumed effect; the conservative end of this range was used to justify the operational sample size." Always report the assumed proportions and the allocation ratio.

5.30.8 Practical Tips

- Equal allocation (1:1) is most efficient for detecting a difference between two proportions; unequal allocation (2:1, 3:1) costs 10–30 % more total n for the same power and is justified only when the cost or feasibility of the two arms differs substantially.
- For rare events ($p < 0.01$ in either arm), the Normal-approximation formulas are unreliable; use exact Fisher's-test-based power calculations or simulation, available in `Exact::power.exact.test()` or via Monte Carlo enumeration.
- Continuity correction (Yates) inflates the required sample size slightly and is the historical default for 2×2 chi-squared tests; modern practice often omits it for likelihood-ratio or score tests, so verify what your software is computing.
- For cluster-randomised designs, inflate the calculated n by the design effect $1+(\bar{m}-1)\text{ICC}$, where $\bar{m}$ is the average cluster size and ICC is the intraclass correlation; ignoring clustering grossly under-powers cluster trials.
- Cohen's h benchmarks (0.20 / 0.50 / 0.80) are useful for translating effect-size language into sample-size expectations, but tying the calculation to the actual expected proportions via `ES.h(p1,p2)` is more defensible than invoking the benchmarks directly.
- For non-inferiority or equivalence designs, the calculation differs from a superiority test; use `pwr.2p.test()` with the appropriate margin or specialised tools like `bsamsize()` with non-inferiority arguments.

5.30.9 R Packages Used

`pwr::pwr.2p.test()` and `pwr::pwr.2p2n.test()` for canonical Cohen's h Normal-approximation calculations; `Hmisc::bsamsize()` for direct-proportion-scale calculations with optional continuity correction; `Exact::power.exact.test()` for exact-binomial power across small-sample two-proportion designs; `clusterPower` for cluster-randomised two-proportion power; `Mediana` for trial-design simulation including two-proportion power across complex designs.

5.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.30.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.31 Sample Size for a Two-Sample t-Test

5.31.1 Introduction

Before running a two-group comparison, the investigator needs to know how many observations per group will be required to detect a difference of scientific importance with acceptable probability. Power analysis for the two-sample t-test answers this question by linking four quantities: the true effect size, the significance level, the power, and the sample size. Given any three, R can solve for the fourth.

5.31.2 Prerequisites

Familiarity with the two-sample t-test and with the concepts of type I and type II error is assumed. The reader should know that "power" means $1 - \beta$, the probability of rejecting H_0 when H_1 is true.

5.31.3 Theory

Under H_0, the t statistic follows a central t distribution with $\nu = n_1 + n_2 - 2$ degrees of freedom (Student's test, equal variances). Under H_1 where $\mu_1 - \mu_2 = \delta$, the statistic follows a *non-central* t distribution with non-centrality parameter

$$\lambda = \frac{\delta}{\sigma\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}.$$

Power is the probability that the test statistic exceeds the critical value under this non-central distribution. For Cohen's standardised effect size $d = \delta/\sigma$ and equal group sizes $n_1 = n_2 = n$, this simplifies to $\lambda = d\sqrt{n/2}$.

The four power quantities and their trade-offs:

- **Effect size** d: the minimum standardised difference considered scientifically meaningful. By Cohen's conventions, $d = 0.2$ is small, 0.5 is medium, 0.8 is large – but these conventions are field-dependent and should be used with judgement.
- **Significance level** α: usually 0.05 for two-sided testing.
- **Power**: conventionally 0.80 for exploratory work, 0.90 for confirmatory or registered trials.
- **Sample size per group** n: what we typically solve for.

5.31.4 Assumptions

Power analysis inherits the assumptions of the t-test it plans for: independent groups, approximately normal data (or large enough n for the CLT), and equal variances for Student's test (unequal for Welch's).

5.31.5 R Implementation

```
library(pwr)

pwr.t.test(d = 0.5, sig.level = 0.05, power = 0.80,
           type = "two.sample", alternative = "two.sided")

pwr.t.test(n = 30, sig.level = 0.05, power = 0.80,
           type = "two.sample", alternative = "two.sided")

pwr.t.test(n = 30, d = 0.5, sig.level = 0.05,
           type = "two.sample", alternative = "two.sided")

library(pwrss)
pwrss.t.2means(mu1 = 110, mu2 = 100, sd1 = 18, sd2 = 15,
               power = 0.80, alpha = 0.05)
```

The first `pwr.t.test()` call asks: how many subjects per group do I need to detect a medium-sized effect ($d = 0.5$) with 80% power? The second asks: given $n = 30$ per group, what is the minimum detectable effect at 80% power? The third asks: given $n = 30$ and $d = 0.5$, what is my power? The `pwrss::pwrss.t.2means()` call works directly in raw-mean terms, which is often more natural when planning a clinical trial.

5.31.6 Output & Results

The first call returns $n \approx 64$ per group – 128 subjects in total. The second returns a minimum detectable $d \approx 0.735$, corresponding to a raw difference of about 11 points if $\sigma = 15$. The third returns a power of approximately 0.48, showing that 30 per group is badly underpowered for a medium effect.

```
     Two-sample t test power calculation

              n = 63.76576
              d = 0.5
      sig.level = 0.05
          power = 0.8
    alternative = two.sided
```

5.31.7 Interpretation

The recommended manuscript phrasing is: "Assuming a between-group standardised mean difference of $d = 0.5$ (a medium effect by Cohen's criteria), a two-sided α of 0.05, and power of 0.80, the required sample size is 64 per group. Allowing for 10% loss to follow-up, we aim to enrol 72 per group (144 total)." The power analysis should be reported in the methods section of the grant or protocol, not buried in an appendix.

5.31.8 Practical Tips

- Base your effect size on published data or a pilot, not on a convention like $d = 0.5$. If the published effect is $d = 0.3$, planning for $d = 0.5$ will leave you underpowered.
- Use Welch's sample size for unequal variances (`pwr.t2n.test()` or `pwrss.t.2means()` with `sd1!=sd2`). It is slightly more conservative, which is what you want at the planning stage.
- Plan for loss to follow-up, noncompliance, and data exclusion by inflating the computed n by a realistic percentage (typically 10-25%).
- If the minimum clinically important difference (MCID) is known in raw units, use that directly rather than translating to d. It is more interpretable to clinicians.
- Always produce a sensitivity analysis: a curve of power against n, or a tornado plot of sample size against plausible values of effect size and SD.

5.31.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.31.10 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.32 Sample Size Sensitivity Analysis

5.32.1 Introduction

Every power calculation rests on assumptions that the investigator cannot know exactly in advance: the true effect size, the outcome SD, the dropout rate, the ICC in a cluster design. Sensitivity analysis reports sample size across a realistic range of these assumptions, rather than a single point estimate.

5.32.2 Prerequisites

Power analysis.

5.32.3 Theory

A sensitivity analysis is a two-dimensional (or higher) grid of required sample sizes, varying one or two assumptions while holding others fixed. Tabular or

graphical (contour / heatmap) presentation lets readers see the robustness of the chosen n.

5.32.4 Assumptions

The chosen assumption grid covers the plausible parameter space. Too-narrow grids miss important regions; too-wide grids are unfocused.

5.32.5 R Implementation

```r
library(pwr); library(ggplot2)

# 2D sensitivity: n per group vs d and power
d_grid <- seq(0.2, 0.8, by = 0.05)
power_grid <- c(0.70, 0.80, 0.90)

res <- expand.grid(d = d_grid, power = power_grid)
res$n <- mapply(function(d, p)
  pwr.t.test(d = d, power = p, sig.level = 0.05, type = "two.sample")$n,
  res$d, res$power)

ggplot(res, aes(d, n, colour = factor(power))) +
  geom_line(linewidth = 1) +
  scale_y_log10() +
  scale_colour_manual(values = c("#F4A261", "#6A4C93", "#2A9D8F")) +
  labs(x = "Cohen's d", y = "n per group (log scale)",
       colour = "Power",
       title = "Sample size sensitivity: d x power") +
  theme_minimal()

# Inflation for dropout
n_base <- 64
dropout_rates <- c(0, 0.05, 0.10, 0.15, 0.20)
n_inflated <- round(n_base / (1 - dropout_rates))
data.frame(dropout = dropout_rates, n_enrol = n_inflated)
```

5.32.6 Output & Results

For $d = 0.5$: n ranges from 51 at power 0.70 to 86 at power 0.90. For $d = 0.3$: from 138 to 234. Dropout inflation at 15 % pushes n from 64 to 76 per group.

5.32.7 Interpretation

“Sensitivity analysis shows that for plausible effect sizes (Cohen’s d between 0.35 and 0.55), the required sample size ranges from 72 to 164 per arm at 80

% power. We plan to enrol 180 per arm, accommodating the upper end of this range and allowing for 10 % attrition."

5.32.8 Practical Tips

- Always report sensitivity analysis alongside the headline sample-size calculation in grant proposals.
- Present the calculation that underpins your chosen n in bold; show neighbouring values for context.
- Two-dimensional grids (effect size vs. power, effect size vs. variance) are typical; higher dimensions confuse readers.
- Include a dropout inflation factor explicitly; reviewers frequently check.
- For adaptive or sequential designs, sensitivity to interim analyses and alpha spending deserves separate reporting.

5.32.9 Reporting

A useful sensitivity table is small enough that a reviewer can read every row and large enough that the chosen design point is clearly bracketed by less and more conservative alternatives. State which inputs were varied and which were held fixed, justify the ranges in terms of pilot data or published evidence, and identify the row that drives the final enrolment target so the audit trail from assumption to commitment is explicit. When dropout inflation is applied, write the formula and the assumed completion proportion in the protocol, because reviewers and ethics committees rely on that line to confirm the analysis sample is consistent with the powered effect.

5.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

5.32.11 See also — labs in this chapter

- Sample size, power, and Quarto reporting
- Power: closed-form and simulation

5.33 Week 1, Session 5 — Power: closed-form and simulation

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

5.33.1 Learning objectives

- Run a closed-form power calculation for a two-sample t-test with `pwr`.
- Simulate power for a scenario where closed-form solutions do not exist using `simr`.
- Describe the inputs to any power calculation (effect, variance, alpha, design).

5.33.2 Prerequisites

inference and mixed-model basics (we will fit a small `lmer`).

5.33.3 Background

Power is the probability of rejecting the null hypothesis when a specific alternative is true. Four inputs set it: the effect size you care about, the variability of the outcome, the significance level, and the sample size (with the design). Closed-form formulas exist for standard one- and two-sample tests and for some simple regression settings. For anything more elaborate — clustered data, nonlinear models, interim looks, complex survival designs — you simulate.

Simulation-based power is conceptually straightforward: for a given scenario, generate many datasets under the alternative, run the planned analysis on each, and compute the fraction of times you reject the null. The `simr` package does this for models fit with `lme4`, allowing you to vary the sample size or the effect and trace out a power curve.

A useful rule of thumb: if your closed-form calculation is for a model simpler than the one you will actually fit, the closed-form is giving you a lower bound on the sample size you need, not an answer.

5.33.4 Setup

```
library(tidyverse)
library(pwr)
library(lme4)
library(lmerTest)
library(simr)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

5.33.5 1. Hypothesis

Detect a standardised mean difference of $d = 0.4$ between two groups with 80% power at a 5% two-sided significance level. Then power a mixed model with 20 clusters and 10 observations each to detect a fixed effect of 0.3.

5.33.6 2. Visualise

```r
ns <- seq(20, 400, by = 20)
pw <- map_dbl(ns, \(n) pwr.t.test(
  n = n / 2, d = 0.4, sig.level = 0.05, type = "two.sample"
)$power)

tibble(n = ns, power = pw) |>
  ggplot(aes(n, power)) +
  geom_line() +
  geom_hline(yintercept = 0.8, linetype = "dashed", colour = "grey40") +
  labs(x = "Total sample size", y = "Power")
```

5.33.7 3. Assumptions

Closed-form assumes equal variance across arms, independent observations, and the nominal effect size. The simulation below adds random intercepts at the cluster level and therefore relaxes independence.

5.33.8 4. Conduct

```r
# Closed-form
res <- pwr.t.test(d = 0.4, power = 0.8, sig.level = 0.05, type =
 ↪  "two.sample")
res
```

```r
# Simulation: a small two-arm cluster study
J   <- 20                              # clusters
nj  <- 10                              # obs per cluster
dat <- tibble(
  cluster = factor(rep(seq_len(J), each = nj)),
  arm     = rep(rep(c(0, 1), each = nj), length.out = J * nj),
  y       = rnorm(J * nj)              # placeholder; simr overwrites
)

m0 <- makeLmer(
  y ~ arm + (1 | cluster),
  fixef   = c(0, 0.3),
  VarCorr = 0.5,
  sigma   = 1.0,
  data    = dat
)
m0
```

```
ps <- powerSim(m0, nsim = 50, test = fixed("arm"), progress = FALSE)
ps
```

Fifty simulations is enough for a teaching demo; in practice use at least 1000 for a stable estimate.

5.33.9 5. Concluding statement

> A two-sample t-test targeting $d = 0.4$ at 80% power requires about `ceiling(res$n)*2` participants in total. A mixed model with 20 clusters of size 10 and a fixed effect of 0.3 has estimated power of `round(summary(ps)$mean*100)`% in this simulation (50 replicates).

5.33.10 Common pitfalls

- Using the power formula for independent observations on a clustered design.
- Quoting an effect size taken from a pilot study without acknowledging its noise.
- Computing power from the smallest clinically important effect and the observed pilot effect interchangeably.
- Simulating with too few replicates.

5.33.11 Further reading

- Cohen J. *Statistical Power Analysis for the Behavioral Sciences*.
- Green P, MacLeod CJ (2016), *simr: an R package for power analysis of generalized linear mixed models*.
- Chow SC et al. *Sample Size Calculations in Clinical Research*.

5.33.12 Session info

```
sessionInfo()
```

5.33.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

5.34 Week 4, Session 5 — Sample size, power, and Quarto reporting

Workflow lab: Goal → Approach → Execution → Check → Report.

5.34.1 Learning objectives

- Compute analytic sample size for the core inferential designs.
- Produce the same answer by simulation and understand when each approach is preferable.
- Assemble a short Quarto report whose numbers, tables, and figures all regenerate from a single render.

5.34.2 Prerequisites

Courses 1 Weeks 1–4 up to this lab; `pwr`, `gtsummary`, and `broom` installed.

5.34.3 Background

Sample size is one of the few places where applied statistics is consulted *before* the data exist. A study that is too small cannot distinguish a real effect from noise; a study that is too large wastes participants and money. The four ingredients of a sample-size calculation — effect size, variability, the significance level, and power — translate directly between a clinical protocol and a funding application, and no competent ethics committee will approve a protocol without them.

There are two complementary approaches. Closed-form formulae, packaged in `pwr` and friends, are fast, transparent, and correct for the textbook designs: one- and two-sample t-tests, one- and two-proportion tests, correlation, and ANOVA. Simulation-based power, by contrast, is the honest answer for anything the textbooks skip — mixed models, skewed outcomes, interim analyses — and costs only a few minutes of compute. The habit to build is to compute analytically first, then verify by simulation; if the two disagree, either the formula does not apply to your design or your simulation is wrong, and you need to know which.

Reporting is the other half of this lab. Every number in a well-written report should trace back to the code that produced it. Quarto, with `gtsummary` for tables and `broom::tidy()` for model output, makes this nearly automatic.

5.34.4 Setup

```
library(tidyverse)
library(pwr)
library(gtsummary)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

5.34.5 1. Goal

Plan a hypothetical two-arm trial comparing a new antihypertensive with placebo, choose a sample size with 80% power to detect a 5 mmHg difference at $\alpha = 0.05$, verify by simulation, and produce a reporting table we can paste into the protocol.

5.34.6 2. Approach

Closed-form first. Cohen's d for a 5 mmHg effect on a scale with SD 10 mmHg is $d = 0.5$ — a medium effect by convention.

```
pwr_calc <- pwr.t.test(
  d = 0.5, power = 0.80, sig.level = 0.05,
  type = "two.sample", alternative = "two.sided"
)
pwr_calc
```

The formula calls for about `ceiling(pwr_calc$n)` per arm.

```
ns <- seq(10, 200, by = 5)
powers <- sapply(ns, function(n) {
  pwr.t.test(n = n, d = 0.5, sig.level = 0.05,
             type = "two.sample")$power
})
tibble(n_per_arm = ns, power = powers) |>
  ggplot(aes(n_per_arm, power)) +
  geom_line(linewidth = 0.8) +
  geom_hline(yintercept = 0.8, linetype = 2, colour = "grey50") +
  labs(x = "N per arm", y = "Power",
       title = "Power curve for d = 0.5, α = 0.05")
```

5.34.7 3. Execution

Verify by simulation. A function that runs the trial once and returns the p-value:

```
sim_trial <- function(n_per_arm, delta = 5, sd = 10) {
  placebo <- rnorm(n_per_arm, 0, sd)
  drug    <- rnorm(n_per_arm, -delta, sd)
  t.test(drug, placebo, var.equal = FALSE)$p.value
}

sim_power <- function(n_per_arm, reps = 500, ...) {
  mean(replicate(reps, sim_trial(n_per_arm, ...)) < 0.05)
}

n_try  <- ceiling(pwr_calc$n)
sim_power(n_try, reps = 1000)
```

Within Monte-Carlo error of the analytic 0.80, as expected.

5.34.8 4. Check

Now imagine the data have been collected. We simulate one realisation and fit the planned analysis.

```
n <- ceiling(pwr_calc$n)
trial <- tibble(
  arm = rep(c("placebo", "drug"), each = n),
  sbp_change = c(rnorm(n, 0, 10), rnorm(n, -5, 10))
)

fit <- t.test(sbp_change ~ arm, data = trial, var.equal = FALSE)
broom::tidy(fit)
```

5.34.9 5. Report

```
trial |>
  tbl_summary(
    by = arm,
    statistic = list(all_continuous() ~ "{mean} ({sd})"),
    digits    = all_continuous() ~ 1,
    label     = list(sbp_change = "Change in SBP (mmHg)")
  ) |>
  add_p() |>
  modify_caption("**Table 1. Change in systolic blood pressure, by
   ↪  arm.**")
```

> In a simulated two-arm trial ($n =$ `n` per arm), mean change in systolic blood pressure was lower in the drug arm than in placebo by `round(-diff(fit$estimate),1)` mmHg (95% CI: `round(-fit$conf.int[2],1)` to `round(-fit$conf.int[1],1)`; $p =$ `signif(fit$p.value,2)`).

5.34.10 Common pitfalls

- Plugging a standardised effect size into a sample-size calculator without stopping to think whether it is plausible in your field.
- Quoting "power = 0.8" as a universal target when your decision would justify 0.9 or 0.95.
- Reporting a p-value without the effect size and interval that make it interpretable.
- Pasting numbers into the manuscript by hand — anything that breaks the render trail will eventually break the numbers.

5.34.11 Further reading

- Writing a report.
- Champely S. (2024). *pwr: Basic functions for power analysis.*
- Cohen J. (1988). *Statistical Power Analysis for the Behavioral Sciences.*

5.34.12 Session info

```
sessionInfo()
```

5.34.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 6

Data Visualisation

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

The `ggplot2` grammar, panelled and faceted layouts, statistical graphics for regression and survival, and visual diagnostics. Style choices are biased toward what reproduces in print and what survives a black-and-white photocopy.

This chapter contains **35 method pages** and **1 lab**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

6.1 Method pages

Method	Source slug
Aesthetics and Geoms	`aesthetics-and-geoms`
Annotations and Labels	`annotations-and-labels`
Bar Charts	`bar-charts`
Bland-Altman Plots	`bland-altman-plots`
Boxplots	`boxplots`
Bubble Plots	`bubble-plots`
Colour Palettes	`colour-palettes`
Colour-Blind-Safe Plots	`colour-blind-safe-plots`
Contour Plots	`contour-plots`
Correlation Heatmaps	`correlation-heatmaps`
Density Plots	`density-plots`
Dot Plots	`dot-plots`
Facets and Panels	`facets-and-panels`
Forest Plots (Visualisation)	`forest-plots-viz`
Funnel Plots (Visualisation)	`funnel-plots-viz`
ggplot2 Themes	`ggplot-themes`

Method	Source slug
Heatmaps	`heatmaps`
Hexbin Plots	`hexbin-plots`
Histograms	`histograms`
Interactive Plots with ggiraph	`interactive-ggiraph`
Interactive Plots with plotly	`interactive-plotly`
Line Plots	`line-plots`
Pairs Plots	`pairs-plots`
Patchwork: Multi-Plot Composition	`patchwork-composition`
Raincloud Plots	`raincloud-plots`
Ridge Plots	`ridge-plots`
ROC Curves	`roc-curves-plot`
Saving and Exporting Figures	`ggsave-and-export`
Scales and Coordinates	`scales-and-coordinates`
Scatter Plots	`scatter-plots`
Stacked and Dodged Bars	`stacked-dodged-bars`
Survival Curves	`survival-curves-plot`
The Grammar of Graphics	`grammar-of-graphics`
Time Series Plots	`time-series-plots`
Violin Plots	`violin-plots`

6.2 Labs

Lab
ggplot2 grammar and multi-panel layouts

6.3 Aesthetics and Geoms

6.3.1 Introduction

`ggplot2` is built on Wilkinson's grammar of graphics, which separates two concerns that ad-hoc plotting languages tend to conflate. An **aesthetic** is a mapping from a column of data to a visual property: x-position, y-position, colour, fill, shape, size. A **geom** is the geometric object drawn using those aesthetics: a point, a line, a bar, a histogram, a smoother. Once you internalise the split — "what variable becomes which visual property" versus "what shape do I want drawn" — `ggplot2` becomes a Lego kit of swappable parts rather than a dialect to memorise.

6.3.2 Prerequisites

A working knowledge of basic `ggplot2` syntax (`ggplot()`, `aes()`, `+geom_*`), and tidy data conventions where each row is an observation and each column a

variable.

6.3.3 Theory

An **aesthetic mapping** lives inside `aes()` and binds a column to a visual property:

```
aes(x = flipper_length_mm, y = body_mass_g, colour = species)
```

The mapping triggers automatic scale construction, legend generation, and axis labelling — `ggplot2` knows that a continuous numeric column needs a continuous scale and a categorical column needs a discrete one.

A **geom** consumes the aesthetics and draws the corresponding marks. Common geoms include `geom_point()` for scatter, `geom_line()` for lines, `geom_bar()` and `geom_col()` for bars, `geom_histogram()` and `geom_density()` for distributions, `geom_smooth()` for trend lines.

Mapping vs setting is the most important distinction in `ggplot2`. Inside `aes()` binds a visual property to a variable, producing one value per row and a legend. Outside `aes()` sets a constant for all rows with no legend. `geom_point(colour="blue")` makes every point blue; `geom_point(aes(colour=species))` colours points by species.

Multiple geoms in the same plot share base aesthetics declared in `ggplot(aes(...))` and may add their own. This layering is the heart of the grammar: a complex figure is just a stack of compatible geoms reading the same data.

6.3.4 Assumptions

Tidy data: one observation per row, one variable per column. Long-format frames are the standard input for `ggplot2`; wide-format data should be reshaped via `tidyr::pivot_longer()` before plotting.

6.3.5 R Implementation

```
library(ggplot2); library(palmerpenguins)
data(penguins, package = "palmerpenguins")

# Basic mapping: x, y, colour
ggplot(penguins, aes(flipper_length_mm, body_mass_g, colour = species)) +
  geom_point(alpha = 0.7) +
  theme_minimal()

# Adding a smoother geom on top
ggplot(penguins, aes(flipper_length_mm, body_mass_g, colour = species)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", se = FALSE) +
```

```
  theme_minimal()

# Aesthetics differ between geoms: shape is mapped only for points
ggplot(penguins, aes(flipper_length_mm, body_mass_g)) +
  geom_point(aes(colour = species, shape = sex), alpha = 0.7) +
  theme_minimal()

# Constant aesthetics outside aes()
ggplot(penguins, aes(flipper_length_mm, body_mass_g)) +
  geom_point(colour = "steelblue", size = 3, alpha = 0.6) +
  theme_minimal()
```

6.3.6 Output & Results

Four versions of the same scatter: a basic colour-coded plot, the same plot with per-species linear smoothers added, a four-aesthetic plot mapping species to colour and sex to shape simultaneously, and a single-colour version where every aesthetic except position is held constant. Each illustrates a different combination of mapping and setting.

6.3.7 Interpretation

A reporting sentence (figure caption): "Body mass against flipper length (n = 333 penguins, three species); colour encodes species and shape encodes sex. Per-species linear smoothers are overlaid in the right panel." Be explicit about which aesthetics carry information and which are constant; readers should not have to infer the mapping.

6.3.8 Practical Tips

- Put aesthetics shared across geoms in the base `ggplot()` call; put geom-specific aesthetics inside the geom. Repetition is fine for clarity but is rarely necessary.
- Reach for `alpha` to mitigate overplotting in dense scatter plots; values between 0.2 and 0.5 typically reveal density structure.
- `fill` controls the interior of geoms with area (bars, boxplots, polygons); `colour` controls edges and points. Mixing them up is one of the most common `ggplot2` errors.
- For categorical colour scales, prefer `viridis::scale_colour_viridis_d()` or `RColorBrewer::scale_colour_brewer()`; avoid the default rainbow.
- Use `shape` only for two to six groups; beyond that, shapes become indistinguishable and a facet or text label communicates better.
- When you want a constant value but `ggplot2` keeps drawing a legend, you forgot to move the assignment outside `aes()`; this is the single most common debugging step.

6.3.9 R Packages Used

`ggplot2` for the core grammar of graphics; `palmerpenguins` for a clean teaching dataset rich in aesthetic-mapping opportunities; `viridis` and `RColorBrewer` for accessible categorical and continuous scales; `ggrepel` for non-overlapping point labels that supplement aesthetic mappings with text.

6.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.3.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.4 Annotations and Labels

6.4.1 Introduction

A figure that shows the data without any annotation forces the reader to do the interpretive work alone. A figure with too many annotations buries the data under labels, arrows, and ornamental boxes. The middle ground — a few well-placed labels, an arrow that points at the one observation worth singling out, a rich-text title that names the comparison — is what separates publication-quality plots from informative-but-amateurish ones. The `ggplot2` ecosystem provides three families of annotation tools that cover virtually every requirement: `annotate()` for single hand-placed items, `geom_text` / `geom_label` (and the repelling `ggrepel` variants) for data-driven labels, and `ggtext` for rich Markdown and HTML inside titles, axes, and strip text.

6.4.2 Prerequisites

A working knowledge of `ggplot2`'s grammar of graphics, layers, and aesthetic mappings.

6.4.3 Theory

- `annotate()` adds a single geom (text, segment, rectangle, point) at hard-coded coordinates with no aesthetic mapping from the data; this is the right tool for a one-off label or arrow that does not depend on a row of the data frame.

- `geom_text()` and `geom_label()` map labels to rows of the data, drawing one label per row at the corresponding (x, y) coordinates.
- `ggrepel::geom_text_repel()` and `geom_label_repel()` solve the universal overlap problem by iteratively pushing labels away from each other and from the data points, with adjustable force, segment lines, and exclusion regions.
- `ggtext::element_markdown()` parses Markdown and a subset of HTML in axes, legends, titles, and strip text, enabling inline italics, bold, colour, and superscripts without dropping into expression syntax.

6.4.4 Assumptions

No statistical assumptions; the only requirement is that text strings be on the same scale as the surrounding data so labels appear in the right region of the plot.

6.4.5 R Implementation

```r
library(ggplot2); library(ggrepel); library(ggtext)

p <- ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  theme_minimal()

# Single annotation
p + annotate("text", x = 3, y = 30, label = "Lightweight, efficient",
 ↪  size = 5)

# Labelled points with repel
p + geom_text_repel(aes(label = rownames(mtcars)), size = 3)

# Arrow with annotation
p + annotate("segment", x = 5, xend = 4, y = 25, yend = 20,
             arrow = arrow(length = unit(0.3, "cm"))) +
  annotate("text", x = 5.1, y = 25.5, label = "Outlier")

# Markdown in titles
p + labs(title = "Weight vs <span style='color:#2A9D8F'>*mpg*</span>") +
  theme(plot.title = element_markdown())
```

6.4.6 Output & Results

Four variations on the same scatterplot: one with a hand-placed text annotation, one with each car labelled by name and laid out by `ggrepel` so labels do not collide, one with an arrow drawing attention to a single point, and one with

a Markdown-formatted title. The point geom remains identical across all four; only the annotation layer changes.

6.4.7 Interpretation

A reporting sentence: "We labelled the four cars with the highest fuel efficiency using `ggrepel::geom_text_repel()` (force = 1, max.overlaps = 10), and emphasised the outlying Maserati Bora with a hand-placed arrow." When annotations carry interpretive content, describe them in the figure caption so readers do not have to guess what is being highlighted.

6.4.8 Practical Tips

- Prefer `ggrepel::geom_text_repel()` over `geom_text()` whenever labels can overlap; the repelling variants always look more polished and are robust to data updates.
- `annotate()` is the right tool for a single fixed label; using `geom_text()` with a one-row tibble is unnecessarily verbose.
- For a small set of labels with the same style, build a small data frame and map all aesthetics through `geom_text()` rather than stacking many `annotate()` calls.
- Keep annotations away from data marks, legends, and figure margins; if you cannot, increase the plot size before re-styling the labels.
- Use `ggtext::element_markdown()` for inline italics in scientific names, bold for statistical thresholds, and colour for paired emphasis between text and data; it is far cleaner than `expression()`-based plotmath.
- Save the labelled and unlabelled versions of any figure separately; reviewers occasionally request a "clean" version for the supplement.

6.4.9 R Packages Used

`ggplot2` for the base plot system, `ggrepel` for collision-free labels, and `ggtext` for Markdown/HTML rendering inside themes; `cowplot` and `patchwork` complement the workflow when you need to compose annotated figures into multi-panel layouts.

6.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.4.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.5 Bar Charts

6.5.1 Introduction

Bar charts are the workhorse for categorical data and the geom most frequently misused. The confusion almost always stems from a single distinction: a bar can show how many observations fall in a category (a count) or a summary statistic computed for that category (a mean, a percentage, a proportion). `ggplot2` separates the two cases through `geom_bar()` for counts and `geom_col()` for pre-computed values, and getting the right geom for the data on hand is the first step to a sensible figure.

6.5.2 Prerequisites

A working knowledge of categorical variables, basic summary statistics (mean, standard error), and `ggplot2`'s aesthetic mappings.

6.5.3 Theory

`geom_bar()` defaults to `stat="count"`: it tabulates rows by the x-aesthetic and draws bars whose heights equal those counts. No y-aesthetic is required; supplying one along with `stat="identity"` is equivalent to using `geom_col()` directly.

`geom_col()` plots an explicit y-aesthetic as the bar height — useful when the data are already summarised. For group means with uncertainty, pair `geom_col()` with `geom_errorbar()` whose `ymin` and `ymax` come from a confidence interval or standard error column.

The cognitive risk with bar-plus-error-bar plots is well documented: they hide the within-group distribution and overemphasise the mean. For showing distributional information, raincloud plots, dot plots, or boxplots are more transparent. Bars remain the right choice for genuine counts and for cumulative quantities (proportions, percentages, totals).

6.5.4 Assumptions

For count bars: the x-axis is categorical or discrete. For summary bars: the y-aesthetic is a meaningful summary of the data, ideally one that respects the bar's "starts at zero" implication.

6.5.5 R Implementation

```r
library(ggplot2); library(dplyr)

# Counts
ggplot(mpg, aes(class)) +
  geom_bar(fill = "#2A9D8F") +
  theme_minimal()

# Ordered bars by count
ggplot(mpg, aes(forcats::fct_infreq(class))) +
  geom_bar(fill = "#2A9D8F") +
  labs(x = "Class") +
  theme_minimal()

# Summarised values (e.g., mean mpg by class)
mpg_summary <- mpg |> group_by(class) |>
  summarise(mean_hwy = mean(hwy), se = sd(hwy)/sqrt(n()))
ggplot(mpg_summary, aes(class, mean_hwy)) +
  geom_col(fill = "#2A9D8F") +
  geom_errorbar(aes(ymin = mean_hwy - se, ymax = mean_hwy + se),
                width = 0.2) +
  theme_minimal()
```

6.5.6 Output & Results

Three bar charts: a count of cars per class in alphabetical order, the same data with categories ordered by descending frequency, and a summary chart of mean highway mileage per class with ± 1 standard-error bars. Reordering and the explicit summary radically improve readability over the default.

6.5.7 Interpretation

A reporting sentence (figure caption): "Bar heights show the mean highway fuel efficiency per class (n varies, see Table 1); error bars are ± 1 standard error of the mean. Classes are ordered alphabetically." Always state the summary statistic and the uncertainty measure in the caption — bars are not self-explanatory.

6.5.8 Practical Tips

- Order categories by value (`forcats::fct_reorder()` or `fct_infreq()`) rather than alphabetically; sorted bars communicate ranking instantly.
- Use `geom_col()` when you already have summary statistics and `geom_bar()` (default `stat="count"`) when you want raw frequencies.
- For two-category comparisons, a simple sentence often beats a bar chart; reserve bars for three or more groups.

- Add the underlying observations as jittered points (`geom_jitter(width =0.2,alpha=0.4)`) on top of summary bars to recover the distributional information bars hide.
- Always label the y-axis with units when bars summarise quantities ("Mean fuel economy (mpg)") rather than counts.
- Avoid 3D bars, gradient fills along the bar length, and other ornamental flourishes; they distort the perceptual mapping between bar length and value.

6.5.9 R Packages Used

`ggplot2` for `geom_bar()`, `geom_col()`, and `geom_errorbar()`; `dplyr` and `forcats` for the summary and category-ordering pipeline; `ggdist` if you want to overlay distributional summaries (slabintervals) on top of bar charts to communicate spread alongside the mean.

6.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.5.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.6 Bland-Altman Plots

6.6.1 Introduction

The Bland-Altman plot is the standard graphical tool for assessing agreement between two measurement methods, two raters, or two devices. Unlike a correlation coefficient or a 45-degree scatter, which conflate agreement with linear association, a Bland-Altman plot displays the *difference* between paired measurements against their *mean*, exposing systematic bias, proportional bias, and the limits within which 95 % of differences are expected to fall. Since Bland and Altman's 1986 *Lancet* paper, it has become the canonical method-comparison figure in clinical chemistry and laboratory medicine.

6.6.2 Prerequisites

A working understanding of paired measurements, mean and standard deviation, and the distinction between agreement (do two methods give the same value?) and correlation (do two methods rank observations the same way?).

6.6.3 Theory

For paired measurements (X_i, Y_i), compute the mean $M_i = (X_i + Y_i)/2$ and the difference $D_i = X_i - Y_i$. Plot M on the x-axis and D on the y-axis, with three horizontal reference lines:

- **Bias**: $\bar{D}$, the mean of the differences. Non-zero bias indicates systematic disagreement.
- **Upper limit of agreement (LoA)**: $\bar{D} + 1.96 \cdot \mathrm{SD}(D)$.
- **Lower LoA**: $\bar{D} - 1.96 \cdot \mathrm{SD}(D)$.

If the differences are approximately Normal, about 95 % of points fall within the LoA. Trends in D as a function of M — typically tested by regressing D on M — indicate proportional bias, where the disagreement scales with magnitude. Heteroscedasticity (a fan-shaped pattern) suggests the LoA should be expressed as a percentage of the mean rather than as an absolute interval.

6.6.4 Assumptions

Differences approximately Normal, no proportional bias (constant spread across the measurement range), and independent paired observations. For repeated measures within subjects, the modified Bland-Altman 1999 method partitions within- and between-subject variance.

6.6.5 R Implementation

```r
library(ggplot2); library(blandr)

set.seed(2026)
m1 <- rnorm(80, 100, 15)
m2 <- m1 + rnorm(80, 2, 5)

d <- data.frame(M = (m1 + m2)/2, D = m1 - m2)
bias <- mean(d$D)
loa <- bias + c(-1, 1) * 1.96 * sd(d$D)

ggplot(d, aes(M, D)) +
  geom_point(alpha = 0.6) +
  geom_hline(yintercept = bias, colour = "#2A9D8F", linewidth = 1) +
  geom_hline(yintercept = loa, colour = "#F4A261", linetype = "dashed") +
  labs(x = "Mean of methods", y = "Difference (M1 - M2)") +
  theme_minimal()

# With blandr
blandr.draw(m1, m2, plotter = "ggplot")
```

6.6.6 Output & Results

A scatter of differences vs means with three reference lines: a solid line at the bias of approximately 2 units and two dashed lines at the upper and lower limits of agreement of approximately -7.8 and 11.8 units. About 4 of the 80 points (5 %) should fall outside the LoA under the Normality assumption.

6.6.7 Interpretation

A reporting sentence: "Bland-Altman analysis showed a bias of +2.0 units (95 % CI 0.9 to 3.1) and 95 % limits of agreement of -7.8 to 11.8 units; clinical equivalence requires the LoA to fall within ± 5 units, so the methods are not interchangeable as currently calibrated." Always report the LoA with their 95 % confidence intervals — the LoA endpoints are themselves uncertain, especially for $n < 100$.

6.6.8 Practical Tips

- Check for proportional bias by regressing D on M; a significant slope indicates the disagreement scales with magnitude and the constant LoA misrepresents the data.
- Report the LoA with confidence intervals; the standard formulae give intervals of $\pm 1.71 \cdot \text{SE}$ around each LoA.
- For repeated measures within subjects, use Bland-Altman's 1999 modified method that splits the variance into within- and between-subject components; ignoring the structure underestimates the LoA.
- A 45-degree scatter complements (but does not replace) the Bland-Altman plot; it reveals correlation but hides the level-dependent bias the Bland-Altman exposes.
- For heteroscedastic differences, log-transform both measurements first; the resulting Bland-Altman is on the log scale and the LoA become percentages.
- `blandr::blandr.statistics()` returns the full statistical summary including bias, LoA, their CIs, and a Shapiro-Wilk test for the Normality of differences.

6.6.9 R Packages Used

`blandr` for `blandr.statistics()` and `blandr.draw()`; `BlandAltmanLeh` for an alternative interface; `ggplot2` for hand-built versions; `epiR::epi.ccc()` for the concordance correlation coefficient as a complementary single-number agreement summary.

6.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.6.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.7 Boxplots

6.7.1 Introduction

A boxplot — also called a box-and-whisker plot — compresses a continuous distribution into a five-number summary: minimum, first quartile, median, third quartile, maximum, with Tukey's fences at $\pm 1.5 \times \text{IQR}$ defining the whiskers and flagging anything beyond as an outlier. Boxplots are the workhorse for compact group-by-group comparisons because they put many groups side by side without ink overload. They are also the geom most often misused: they hide bimodality, hide sample size, and hide the raw observations behind a tidy rectangle, all of which can mislead readers who do not realise what the box actually summarises.

6.7.2 Prerequisites

A working understanding of quartiles, the interquartile range, and the standard $\pm 1.5 \cdot \text{IQR}$ outlier convention.

6.7.3 Theory

The standard Tukey boxplot has the following components per group:

- A box spanning the first and third quartiles (the IQR).
- A horizontal line at the median inside the box.
- Whiskers extending to the most extreme observation within $\pm 1.5 \times \text{IQR}$ of the box edges.
- Individual points beyond the whiskers drawn as outliers.

Variants:

- **Notched boxplot**: notches at $\pm 1.57 \cdot \text{IQR}/\sqrt{n}$ give an approximate 95 % confidence interval for the median; non-overlapping notches between groups suggest a real difference in medians.
- **Variable-width boxplot**: box width proportional to $\sqrt{n}$, encoding group size visually.
- **Letter-value plot**: replaces the single box with nested boxes of decreasing depth, suitable for very large n where a single Tukey box hides tail structure.

6.7.4 Assumptions

For descriptive use, none. For the notched-boxplot CI interpretation, the approximation assumes approximately symmetric within-group distributions; under heavy skew the notches mislead.

6.7.5 R Implementation

```r
library(ggplot2)

# Standard
ggplot(mtcars, aes(factor(cyl), mpg, fill = factor(cyl))) +
  geom_boxplot() +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()

# Notched
ggplot(iris, aes(Species, Sepal.Length, fill = Species)) +
  geom_boxplot(notch = TRUE) +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()

# Boxplot + jittered points (show data and summary)
ggplot(mtcars, aes(factor(cyl), mpg)) +
  geom_boxplot(outlier.shape = NA, fill = "#2A9D8F", alpha = 0.5) +
  geom_jitter(width = 0.2, alpha = 0.5) +
  theme_minimal()
```

6.7.6 Output & Results

Three views: a standard side-by-side boxplot of mpg by cylinder count, a notched boxplot of iris sepal length per species (notches show approximate median CIs), and a hybrid plot that hides the default outlier dots and overlays jittered raw points so the boxplot's summary and the underlying data are both visible.

6.7.7 Interpretation

A reporting sentence (figure caption): "Boxplots of fuel efficiency by cylinder count (n = 11, 7, 14 for 4, 6, 8 cylinders); boxes span Q1 to Q3, the line marks the median, whiskers extend to the most extreme observation within $1.5 \cdot \text{IQR}$, and dots are outliers." Always describe the whisker definition; some software uses 5 %/95 % quantiles or min/max instead of the Tukey rule.

6.7.8 Practical Tips

- Hide default outliers (`outlier.shape=NA`) whenever you overlay `geom_jitter()`; otherwise extreme points appear twice.
- Notches can overshoot the box for very small samples; R prints a warning when this happens, and the plot becomes uninterpretable.
- Use `varwidth=TRUE` to encode group sizes via box width when groups are unbalanced; the visual cue is subtle but informative.
- For very small samples ($n < 10$), prefer a dot plot or strip plot over a boxplot; quartile-based summaries are unstable for tiny groups.
- Boxplots hide multi-modality; always inspect a density or histogram alongside before concluding that a group is unimodal.
- For very large samples, a letter-value plot (`lvplot::geom_lv()`) preserves more of the tail structure than a single Tukey box.

6.7.9 R Packages Used

`ggplot2` for `geom_boxplot()` and the `varwidth` / `notch` parameters; `lvplot` for letter-value plots at very large n; `ggbeeswarm` and `ggdist` for richer alternatives that combine boxplot summaries with raw points and distributional shape.

6.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.7.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.8 Bubble Plots

6.8.1 Introduction

A bubble plot is a scatter plot enriched with a third continuous variable encoded by point size. It is a standard idiom in business analytics ("revenue × growth × market share") and in international development reporting (Hans Rosling's life-expectancy × income × population). The third dimension comes at a real perceptual cost — humans judge area less accurately than position — so a bubble plot's size aesthetic should be reserved for variables where rank order and rough magnitude matter more than precise comparison.

6.8.2 Prerequisites

A working knowledge of scatter plots, `ggplot2`'s aesthetic mappings, and the perceptual principles of position vs area encoding.

6.8.3 Theory

A bubble plot maps three continuous variables: x, y, and `size`. The size mapping must use an *area* scale — `scale_size_area()` — so that perceived magnitude (area) is proportional to the data value. The default `scale_size()` maps the data to point radius, which means a doubled data value occupies four times the visual area; this is a long-standing bug-as-feature in many bubble-chart implementations and produces misleading impressions of magnitude. Colour can encode a fourth variable, but four-variable bubble plots tend to overwhelm readers; alternatives include a facet on the categorical fourth variable or an animated time slider for longitudinal data.

6.8.4 Assumptions

All three (or four) variables are continuous (or treated as such). The size variable is non-negative and on a scale where zero means "absent"; mapping size to a centred or signed quantity is undefined.

6.8.5 R Implementation

```
library(ggplot2)

ggplot(mtcars, aes(wt, mpg, size = hp, colour = factor(cyl))) +
  geom_point(alpha = 0.7) +
  scale_size_area(max_size = 12, name = "Horsepower") +
  scale_colour_brewer(palette = "Set2", name = "Cylinders") +
  theme_minimal()
```

6.8.6 Output & Results

A scatter of weight vs. miles-per-gallon with bubble area proportional to horsepower and colour encoding cylinder count. Heavy, high-horsepower cars cluster in the lower-right corner; light four-cylinder cars cluster in the upper-left. The combination of size and colour makes structural patterns visible that would require a four-panel facet to communicate without bubbles.

6.8.7 Interpretation

A reporting sentence (figure caption): "Bubble plot of vehicle weight vs fuel economy (n = 32); bubble area is proportional to horsepower (`scale_size_area`, max area 12), and colour encodes cylinder count." Always specify that area

(not radius) is the encoding; readers familiar with poorly designed bubble charts will otherwise assume the wrong mapping.

6.8.8 Practical Tips

- Always use `scale_size_area()` so the visual area is proportional to the data; the default radius mapping double-counts magnitude.
- Set `max_size` deliberately (typically 8–15) to avoid enormous bubbles dominating the panel and obscuring smaller observations.
- For many overlapping bubbles, lower `alpha` to 0.4–0.6 and prefer a sequential fill so density still reads through the transparency.
- Do not map size to negative or signed values; area is undefined for negatives, and the visual mapping is ambiguous.
- For a fourth dimension, prefer a facet over a colour aesthetic on top of size; combining both fights for the reader's attention.
- For time series of bubbles ("Rosling-style"), use `gganimate::transition_time(year)` rather than a static panel; the temporal dimension is the variable that makes bubble plots compelling.

6.8.9 R Packages Used

`ggplot2` for the underlying scatter, `scale_size_area()` for correct area encoding, `gganimate` for time-animated bubble charts, and `ggrepel` for non-overlapping bubble labels when individual observations need to be named.

6.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.8.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.9 Colour-Blind-Safe Plots

6.9.1 Introduction

Roughly 8 % of men of European descent and about 0.5 % of women have some form of colour-vision deficiency. Red-green deficiency (deuteranopia and protanopia) is by far the most common; blue-yellow (tritanopia) is rarer but still present in any large readership. A figure that relies on red-versus-green as its

only contrast is unreadable for a meaningful fraction of every audience, including journal reviewers. Designing for colour-blind safety is therefore not optional politeness — it is a basic requirement of accessible scientific communication, increasingly mandated by journal style guides and by web-accessibility standards (WCAG 2.x AA).

6.9.2 Prerequisites

A working knowledge of `ggplot2`'s colour and fill scales, basic familiarity with palette functions, and an understanding that "colour" includes both qualitative palettes (categorical groups) and sequential / diverging palettes (ordered or signed values).

6.9.3 Theory

Safe design strategies fall into three categories:

- **Perceptually uniform palettes**: `viridis`, `cividis`, `magma`, and `inferno` are designed so that equal steps in the colour space correspond to equal steps in perceived brightness, even under simulated colour-blindness. They handle the continuous case well.
- **Hand-curated qualitative palettes**: `RColorBrewer`'s `Set2`, `Dark2`, and `Paired` are explicitly tested for colour-blind safety up to a certain number of categories (six to eight); beyond that, distinguishability degrades.
- **Redundant encoding**: pair colour with shape, linetype, line weight, or a direct text label so a reader who cannot distinguish the colours can still tell the groups apart from a second visual channel.

Simulation tools such as `colorBlindness::cvdPlot()` and `dichromat` re-render any `ggplot` under deuteranopia, protanopia, and tritanopia simulations; treat them as the colour equivalent of a spell-checker run before submission.

6.9.4 Assumptions

No statistical assumptions; the only constraint is that the figure's information content survives loss of one colour channel.

6.9.5 R Implementation

```
library(ggplot2); library(viridis); library(colorBlindness)

# Unsafe: default red-green
p_bad <- ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) +
  geom_point(size = 3) +
  scale_colour_manual(values = c("red", "orange", "green")) +
  theme_minimal()
```

```
# Safe: viridis + shape redundancy
p_good <- ggplot(mtcars, aes(wt, mpg, colour = factor(cyl), shape =
 ↪  factor(cyl))) +
  geom_point(size = 3) +
  scale_colour_viridis_d(option = "D", end = 0.85) +
  theme_minimal()

cvdPlot(p_good)   # simulates deuteranopia, protanopia, tritanopia
```

6.9.6 Output & Results

`cvdPlot()` returns a four-panel composite showing the figure under normal vision, deuteranopia, protanopia, and tritanopia. A safely designed plot remains interpretable in all four panels; an unsafe one collapses to indistinguishable groups in at least one.

6.9.7 Interpretation

A reporting sentence (methods or figure-preparation note): "All figures use the `viridis` palette for ordered fills and `RColorBrewer::Set2` for qualitative colours; categorical groups are encoded redundantly with shape so colour-blind readers can distinguish them. Plots were simulated under deuteranopia and protanopia (`colorBlindness::cvdPlot`) before submission." Mentioning the accessibility check pre-empts reviewer requests to redo the figures.

6.9.8 Practical Tips

- Default to `viridis` or `cividis` for continuous scales; both are perceptually uniform and remain ordered under all common colour-blindness simulations.
- For categorical groups, pair colour with shape (`shape=group`) or with linetype (`linetype=group`); colour alone is insufficient.
- Avoid red-on-green for success/failure annotations; use blue-on-orange or a categorical palette like `Okabe-Ito`.
- Use `colorBlindness::cvdPlot()` (or the `ColorOracle` desktop tool) on every figure before submission; simulating colour-vision deficiencies takes seconds and catches problems that the human eye cannot.
- Beyond colour, ensure adequate contrast (WCAG AA minimum 4.5:1 for body text), avoid pure red or pure green for thin lines, and keep font sizes large enough to remain legible in greyscale photocopies.
- For network or flow diagrams, prefer position and topology to colour as the primary information channel; colour should be reinforcement, not the only signal.

6.9.9 R Packages Used

`viridis` for perceptually uniform sequential palettes, `RColorBrewer` for qualitative and diverging palettes (`brewer.pal`, `display.brewer.all(colorblindFriendly=TRUE)`), `colorBlindness` for `cvdPlot()` simulations, and `paletteer` for unified access to dozens of curated palettes from across the R ecosystem.

6.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.9.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.10 Colour Palettes

6.10.1 Introduction

Colour in a data visualisation is not decoration — it encodes information. The right palette makes ordered values read as ordered, signed values read as signed, and categorical values read as distinct without implying a ranking. The wrong palette (the infamous rainbow) creates false visual hotspots where the hue cycle bends and obscures structure that a perceptually uniform palette would reveal. Mastering the three palette families and choosing among them deliberately is one of the highest-payoff skills in figure design.

6.10.2 Prerequisites

A working knowledge of `ggplot2`'s colour and fill scales, and an awareness of colour-vision deficiencies that affect roughly 8 % of male readers.

6.10.3 Theory

Three palette families serve three different data types:

- **Sequential**: monotone hue or saturation gradient from low to high (`viridis`, `plasma`, `magma`, `YlOrRd`, `Blues`); for ordered data without a meaningful zero or centre.
- **Diverging**: two hues meeting at a neutral midpoint (`RdBu`, `PuOr`, `BrBG`); for signed data where positive and negative deviations from a centre are both meaningful.

- **Qualitative**: distinct, equally salient hues (`Set1`, `Set2`, `Dark2`, `Okabe-Ito`); for unordered categorical groups, ideally limited to eight or fewer.

`viridis` is the modern default for continuous scales because it is perceptually uniform (equal data steps look like equal visual steps), colour-blind safe under deuteranopia and protanopia, and degrades gracefully to greyscale. ColorBrewer palettes label colour-blind-friendly variants explicitly (`display.brewer.all(colorblindFriendly=TRUE)`), and the Okabe-Ito eight-colour palette was designed specifically for accessibility.

6.10.4 Assumptions

The data type matches the palette family — sequential for ordered values, diverging for signed, qualitative for nominal.

6.10.5 R Implementation

```
library(ggplot2); library(viridis); library(RColorBrewer)

# Continuous: viridis
ggplot(faithfuld, aes(waiting, eruptions, fill = density)) +
  geom_tile() +
  scale_fill_viridis_c(option = "plasma") +
  theme_minimal()

# Continuous: diverging
ggplot(mtcars, aes(wt, mpg, colour = hp - mean(hp))) +
  geom_point(size = 3) +
  scale_colour_gradient2(low = "blue", mid = "grey80", high = "red",
                         midpoint = 0) +
  theme_minimal()

# Qualitative: Brewer Set2
ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) +
  geom_point(size = 3) +
  scale_colour_brewer(palette = "Set2") +
  theme_minimal()

# Show colour-blind-safe palettes
display.brewer.all(colorblindFriendly = TRUE)
```

6.10.6 Output & Results

Four illustrative figures: a 2D density tile plot using `plasma` for sequential encoding, a centred-at-zero scatter using `gradient2` for signed deviations, a categorical

scatter using Brewer `Set2` for cylinder counts, and a panel of all colour-blind-friendly Brewer palettes for reference.

6.10.7 Interpretation

A reporting sentence (figure caption): "Cell colour encodes 2D kernel density on the `viridis::plasma` perceptually uniform palette; positive and negative horsepower deviations from the mean use a red-blue diverging scale centred at zero." Always state the palette by name when it is non-default; readers should not have to guess.

6.10.8 Practical Tips

- Use `viridis` (or its variants `magma`, `plasma`, `inferno`, `cividis`) as the default for continuous scales; all are perceptually uniform and colour-blind safe.
- Limit qualitative palettes to 8 colours or fewer; beyond that, switch to shapes, linetypes, or facets to disambiguate groups.
- Use `scale_colour_gradient2()` (or `scale_fill_gradient2()`) for any signed quantity; the centred palette communicates symmetry that one-sided sequentials cannot.
- For greyscale fallback (photocopying, monochrome printing), `viridis` translates almost losslessly; rainbow hues collapse into similar greys and become unreadable.
- Verify colour-blind accessibility on every figure with `colorBlindness::cvdPlot()`; simulating deuteranopia and protanopia takes seconds and prevents costly resubmissions.
- For heatmaps, perceptual uniformity is non-negotiable — a non-uniform palette creates apparent hotspots wherever the hue cycle bends.

6.10.9 R Packages Used

`viridis` for `viridis`, `magma`, `plasma`, `inferno`, `cividis` continuous scales; `RColorBrewer` for the Brewer family of qualitative, sequential, and diverging palettes; `colorspace` for HCL-based custom palettes; `paletteer` for unified access to dozens of curated palettes from across the R ecosystem.

6.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.10.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.11 Contour Plots

6.11.1 Introduction

A contour plot draws isolines — curves along which a scalar field is constant — over a 2D domain. The same geom serves two related purposes: visualising the joint density of paired observations, where the contours are level sets of a kernel-density estimate, and visualising a deterministic function $z = f(x, y)$, where the contours are mathematical isolines. Both uses share a common visual grammar — peaks, ridges, saddles — and contour plots remain one of the most compact ways to communicate where probability mass concentrates or where a function attains specific values.

6.11.2 Prerequisites

A working knowledge of `ggplot2`, basic kernel-density estimation in two dimensions, and the convention that closer contour lines indicate steeper gradients.

6.11.3 Theory

Two distinct workflows feed contour layers in `ggplot2`:

- **Density contours** are computed from raw paired observations via `geom_density_2d()` or `stat_density_2d()`. Internally, a 2D kernel density estimate produces a smoothed surface and `MASS::kde2d()` or `ggplot2`'s built-in estimator returns the level sets.
- **Functional contours** require a complete grid of (x, y, z) values supplied by the user; `geom_contour()` then computes isolines of `z` exactly. The grid resolution determines smoothness; coarse grids produce jagged contours.

Filled variants — `geom_density_2d_filled()` and `geom_contour_filled()` — render closed bands rather than lines and demand a sequential or diverging colour palette so that ordering is preserved.

6.11.4 Assumptions

For density contours, the input is a sample large enough that a kernel density is meaningful (at least a few dozen points; ideally hundreds). For functional contours, the grid covers the domain of interest at adequate resolution.

6.11.5 R Implementation

```r
library(ggplot2); library(metR)

# Contour of 2D density
ggplot(faithful, aes(eruptions, waiting)) +
  geom_point(alpha = 0.3) +
  geom_density_2d(colour = "#F4A261") +
  theme_minimal()

# Filled contour
ggplot(faithful, aes(eruptions, waiting)) +
  geom_density_2d_filled(alpha = 0.7) +
  geom_point(alpha = 0.3, size = 0.5, colour = "white") +
  theme_minimal()

# Function surface: z = sin(x) * cos(y)
grid <- expand.grid(x = seq(-3, 3, 0.1), y = seq(-3, 3, 0.1))
grid$z <- sin(grid$x) * cos(grid$y)
ggplot(grid, aes(x, y, z = z)) +
  geom_contour_filled() +
  theme_minimal()
```

6.11.6 Output & Results

Three figures: density contours of the Old Faithful eruption / waiting joint distribution showing the two well-known modes, the same density rendered as filled bands with the underlying points overlaid, and a filled-contour view of $z = \sin(x)\cos(y)$ on a 61×61 grid showing alternating positive and negative regions.

6.11.7 Interpretation

A reporting sentence (figure caption): "Top: 2D kernel density of eruption duration vs waiting time for Old Faithful (n = 272), with raw observations as background points; the bimodality reflects two distinct eruption regimes. Bottom: deterministic surface $z = \sin(x)\cos(y)$ with filled contours at default levels." Mention the kernel and bandwidth choice when density contours are used, since they shape the visible structure.

6.11.8 Practical Tips

- For sparse data, density contours can be misleading — the estimator extrapolates into low-mass regions and produces artefactual closed loops. Use `bins=5` or `bins=8` for sparse samples; let the default handle dense ones.

- Always overlay raw points (with low alpha) on density contour plots; the points anchor the contours to actual observations and let readers spot extrapolation.
- For functional surfaces, increase grid resolution until the contours are visibly smooth; under-resolved grids produce stair-stepped lines.
- `metR::geom_contour_fill()` and `metR::geom_text_contour()` add labelled contours that read more clearly on filled backgrounds.
- Use sequential palettes (`viridis`, `magma`) for ordered density and diverging palettes (`RdBu`, `BrBG`) for surfaces that cross zero; qualitative palettes destroy the level ordering and are wrong for filled contours.
- For published figures with multiple contour layers, label a few key isolines with their level values so readers can read off heights without consulting a separate legend.

6.11.9 R Packages Used

`ggplot2` for `geom_density_2d()`, `geom_density_2d_filled()`, `geom_contour()`, and `geom_contour_filled()`; `metR` for `geom_contour_fill()` and labelled contours; `MASS::kde2d()` as the underlying kernel density estimator when finer control is needed.

6.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.11.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.12 Correlation Heatmaps

6.12.1 Introduction

A correlation heatmap renders a correlation matrix as a grid of colour-encoded cells, with the cell colour mapping the correlation coefficient on a diverging palette centred at zero. It is the natural overview figure for a multivariate dataset: dozens of pairwise correlations that would be impossible to read as a numeric table become a single image where blocks of similarly correlated variables stand out by colour. Hierarchical reordering of rows and columns brings related variables together and reveals cluster structure, often suggesting candidate factors for downstream analysis.

6.12.2 Prerequisites

A working understanding of correlation coefficients (Pearson, Spearman, Kendall), basic familiarity with hierarchical clustering, and `ggplot2` colour scales.

6.12.3 Theory

A correlation heatmap displays r_{ij} in cell (i, j). Standard variants:

- Symmetric with numbers shown — useful when the matrix is small (≤ 8 variables).
- Upper or lower triangle only — avoids redundancy and saves visual space.
- Significance overlay — marks non-significant correlations as blank or with a cross, so the eye is not drawn to noise.
- Hierarchical reordering — uses correlation distance $(1 - |r|)$ to cluster related variables and reveal block structure.

The colour scale must be diverging (typically red–white–blue or the Brewer `RdBu`) and centred at zero so positive and negative correlations are visually symmetric. Sequential or qualitative palettes destroy the sign information and are wrong for correlations.

6.12.4 Assumptions

A valid correlation matrix (any kind: Pearson for linear, Spearman for monotone, polychoric for ordinal). The matrix should be positive-semi-definite if it will feed downstream methods; this is automatic when computed from raw data with `cor()`.

6.12.5 R Implementation

```
library(corrplot); library(ggcorrplot)

corr_mat <- cor(mtcars)

# corrplot
corrplot(corr_mat, method = "color", order = "hclust",
         addCoef.col = "black", tl.col = "black")

# ggcorrplot with p-value overlay
p_mat <- cor_pmat(mtcars)
ggcorrplot(corr_mat, hc.order = TRUE, type = "lower",
           p.mat = p_mat, insig = "blank",
           lab = TRUE, lab_size = 3)
```

6.12.6 Output & Results

Two correlation heatmaps for the 11-variable `mtcars` dataset: a full symmetric heatmap with hierarchical reordering and numeric cell labels, and a lower-triangle version with non-significant cells masked. The first emphasises the full structure; the second is a cleaner publication figure.

6.12.7 Interpretation

A reporting sentence (figure caption): "Pearson correlation matrix for 11 vehicle attributes (n = 32), reordered hierarchically using $1 - |r|$ as the distance; cells with $p > 0.05$ are blanked. The two visible blocks correspond to size-related variables (weight, displacement, horsepower) and efficiency-related variables (mpg, quarter-mile time)." Always state the correlation type and the reordering method.

6.12.8 Practical Tips

- Use a diverging palette centred at zero; the Brewer `RdBu` or `viridis::cividis` (with the centre stretched) work well in print.
- Hierarchical reordering (`hc.order=TRUE` in `ggcorrplot`, `order="hclust"` in `corrplot`) brings related variables together and is almost always preferable to alphabetical order.
- Mask non-significant cells (`insig="blank"`) to direct the reader's attention to robust signals; alternatively, mark with crosses to show the test was performed.
- For more than 30 variables, the figure becomes a coloured pixel grid without readable labels; consider summarising via principal components or factor analysis instead.
- State the correlation type (Pearson, Spearman, Kendall) and the test correction (if any) in the caption; readers should not have to guess.
- For partial correlations, use `ppcor::pcor()` and feed the resulting matrix into the same heatmap geom; partial correlations often reveal structure that pairwise correlations obscure.

6.12.9 R Packages Used

`corrplot` for the canonical correlation heatmap with reordering and significance overlays; `ggcorrplot` for a `ggplot2`-native interface; `Hmisc::rcorr()` for correlation matrices with associated p-values; `psych::corPlot()` for an alternative with rich cell formatting.

6.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.12.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.13 Density Plots

6.13.1 Introduction

A density plot estimates the underlying probability density of a continuous variable via kernel smoothing, producing a smooth curve instead of the binned bars of a histogram. Densities are easier to compare across groups (overlaid translucent fills are more legible than dodged or transparent bars) and they sidestep the histogram's bin-boundary arbitrariness — at the cost of introducing a different arbitrary parameter, the bandwidth. Choosing a sensible bandwidth and being explicit about it is the main craft of density-plot design.

6.13.2 Prerequisites

A working understanding of histograms, kernel density estimation basics, and the role of bandwidth as the smoothness parameter.

6.13.3 Theory

A kernel density estimate (KDE) is

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\Big(\frac{x - x_i}{h}\Big),$$

where K is a kernel (Gaussian by default) and h is the bandwidth. The bandwidth controls smoothness: small h undersmooths and produces a noisy curve that follows individual observations, large h oversmooths and hides modes or skew. R's default uses Silverman's rule of thumb (`bw="nrd0"`), which works well for unimodal Normal-like data but can oversmooth multi-modal distributions; Sheather–Jones plug-in (`bw="SJ"`) is a more flexible default that adapts to local variability.

The choice of kernel (K) matters far less than the bandwidth in practice — Gaussian, Epanechnikov, and rectangular kernels produce visually indistinguishable density curves at the same h.

6.13.4 Assumptions

Observations are independent and identically distributed; the underlying distribution has a density (not a discrete spike). For bounded variables (proportions, durations), the kernel density spills past the boundary and a reflection or truncation method is required to keep the estimate inside the support.

6.13.5 R Implementation

```r
library(ggplot2)

# Basic density
ggplot(mtcars, aes(mpg)) +
  geom_density(fill = "#2A9D8F", alpha = 0.5) +
  theme_minimal()

# Density by group
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.5) +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()

# Bandwidth control
ggplot(faithful, aes(eruptions)) +
  geom_density(bw = 0.05, colour = "red") +      # undersmoothed
  geom_density(bw = 0.5,  colour = "black") +    # default-ish
  geom_density(bw = 2.0,  colour = "blue") +     # oversmoothed
  theme_minimal()

# Histogram + density overlay
ggplot(mtcars, aes(mpg)) +
  geom_histogram(aes(y = after_stat(density)), bins = 15,
                 fill = "#2A9D8F", alpha = 0.4) +
  geom_density(linewidth = 1) +
  theme_minimal()
```

6.13.6 Output & Results

Four illustrative views: a single-group density, a three-group overlay for the iris dataset, a bandwidth comparison on the bimodal Old Faithful data showing how bandwidth choices alternately reveal or hide the second mode, and a histogram with a density overlay aligned on a common density y-axis.

6.13.7 Interpretation

A reporting sentence (figure caption): "Kernel density estimate of sepal length per species (n = 50 each), Gaussian kernel with Silverman bandwidth; setosa is well-separated from the other two species, which overlap considerably." Always state the bandwidth method (or value) and the kernel; both shape the visible structure.

6.13.8 Practical Tips

- Show a histogram alongside the density during exploratory work; the density alone can hide binning artefacts that the histogram makes visible.
- For bounded data (proportions on $[0, 1]$, response times bounded below by zero), use `density(...,from=0,to=1)` or a reflection-based estimator; default kernels spill past the boundary and produce artificially low density at the edges.
- For small samples ($n < 30$), the density estimate is noisy and misleading; show a strip plot, dot plot, or histogram instead.
- Compare densities across groups via `fill` (with `alpha=0.4`) for translucent overlays or `colour` for line-only renderings; the latter scales better to many groups.
- For 2D joint densities, `geom_density_2d()` draws contours and `stat_density_2d(aes(fill=after_stat(level)),geom="polygon")` renders filled level sets.
- When in doubt, plot the same data with three bandwidths (Silverman, Sheather-Jones, and one tighter) and pick whichever respects the structure you already know exists; do not let bandwidth invent features.

6.13.9 R Packages Used

`ggplot2` for `geom_density()` and `stat_density()`; `KernSmooth` for advanced bandwidth selectors; `ks` for multivariate kernel densities; `ggdist` for slabinterval geoms that combine density and credible-interval annotation.

6.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.13.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.14 Dot Plots

6.14.1 Introduction

"Dot plot" refers to two distinct chart types that share a name and a geometry but answer different questions. **Cleveland dot plots** show one value per category as a single point, replacing bar charts for ranked comparisons; they were popularised by William Cleveland's 1985 *Elements of Graphing Data* on the grounds that perceiving position is more accurate than perceiving length. **Wilkinson dot plots** show one point per observation, stacked at the relevant x-value, and act as a hand-drawn histogram that exposes individual observations rather than binned counts.

6.14.2 Prerequisites

A working knowledge of `ggplot2`, an idea of when to prefer position to length encoding, and basic familiarity with histograms and bar charts.

6.14.3 Theory

Cleveland dot plots (`geom_point()` with category on y and value on x) eliminate the visual noise of bar lengths and make small differences across many categories easy to compare. A faint horizontal segment from zero to the point — `geom_segment()` — preserves the bar-like sense of "distance from origin" without committing the visual heaviness of a filled bar.

Wilkinson dot plots (`geom_dotplot()`) bin observations along one axis and stack equal-sized dots in each bin. The `method="histodot"` argument anchors bins to the data range; `method="dotdensity"` adapts bin centres to local density. They are most useful for small samples ($n < 100$) where every observation should be visible; for larger samples, jittered strip plots or swarm plots scale better.

6.14.4 Assumptions

For Cleveland dot plots: one value per category (or a clearly defined summary). For Wilkinson dot plots: the binwidth makes sense for the data range — too small and dots stack into spikes; too large and they merge into bars.

6.14.5 R Implementation

```r
library(ggplot2)

# Cleveland dot plot: mean mpg by car class
mpg_summary <- aggregate(hwy ~ class, mpg, mean)
ggplot(mpg_summary, aes(hwy, reorder(class, hwy))) +
  geom_segment(aes(x = 0, xend = hwy, yend = class), colour = "grey80") +
```

```
  geom_point(size = 3, colour = "#2A9D8F") +
  labs(y = NULL) +
  theme_minimal()

# Wilkinson dot plot: distribution of a small sample
ggplot(mtcars, aes(mpg)) +
  geom_dotplot(method = "histodot", binwidth = 1, fill = "#2A9D8F") +
  theme_minimal()

# Grouped dot plot
ggplot(iris, aes(Species, Sepal.Length)) +
  geom_dotplot(binaxis = "y", stackdir = "center", binwidth = 0.1,
               fill = "#2A9D8F") +
  theme_minimal()
```

6.14.6 Output & Results

Three views: a Cleveland dot plot ranking car classes by mean highway mpg with faint baseline segments, a Wilkinson dot plot of the 32 `mtcars` mpg values stacked as a histogram, and a grouped Wilkinson plot showing sepal-length distributions for three iris species side by side.

6.14.7 Interpretation

A reporting sentence (figure caption): "Cleveland dot plot of mean highway fuel economy (mpg) by class for 234 vehicles; categories are ordered by descending mean. Wilkinson dot plot of mpg values for 32 vehicles, with each dot representing a single car." When showing summary points, always describe which summary (mean, median, count) the dot represents.

6.14.8 Practical Tips

- Order Cleveland-dot categories by value with `forcats::fct_reorder()` or `reorder()`; alphabetic orderings rarely communicate ranking.
- Combine Cleveland dot plots with horizontal error bars (`geom_errorbarh()` or `geom_segment()` for confidence intervals) when uncertainty is important.
- Choose Wilkinson dot binwidth to match the data resolution; for integer data, set binwidth equal to the unit step.
- For grouped Wilkinson plots, set `stackdir="center"` for symmetric stacks or `"up"` for one-sided; `binaxis="y"` rotates the dotplot for vertical y-axis layouts.
- Dot plots in presentation slides work better than bar charts at small sizes; the bar's length advantage becomes a length-perception headache when the bar is only a few pixels tall.

- Above a few hundred observations per category, switch from Wilkinson dot plots to strip plots, swarm plots (`ggbeeswarm`), or violin plots; stacked dots no longer scale.

6.14.9 R Packages Used

`ggplot2` for `geom_dotplot()`, `geom_point()`, and `geom_segment()`; `forcats` for category reordering; `ggbeeswarm` for swarm-style dot plots that scale to larger samples; `ggdist` for slabinterval geoms that combine the dot-plot and density idioms.

6.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.14.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.15 Facets and Panels

6.15.1 Introduction

Small multiples — a grid of identical plots, one per subgroup — are one of the most powerful visualisation idioms in modern data analysis. They let the reader compare a relationship across many conditions without overlaying everything on a single set of axes, which becomes unreadable beyond three or four series. `ggplot2` calls this faceting and provides two complementary functions: `facet_wrap()` for a one-dimensional set of panels arranged into rows and columns, and `facet_grid()` for a strict row-by-column matrix indexed by two grouping variables.

6.15.2 Prerequisites

A working knowledge of `ggplot2`'s grammar of graphics, layered geoms, and the distinction between aesthetic mappings and faceting variables.

6.15.3 Theory

`facet_wrap(~var)` partitions the data by `var`, creates one panel per level, and arranges them into a single block whose dimensions can be specified with `nrow` or `ncol`. `facet_grid(rows~cols)` produces a matrix of panels indexed by

two variables, leaving an empty cell for combinations that have no data. Both functions share a `scales` argument controlling axis behaviour:

- `"fixed"` (default): all panels share x and y axes — the right choice when the comparative question is the headline.
- `"free_x"`, `"free_y"`, `"free"`: each panel sets its own axis range — useful when scales differ enough that fixed axes hide structure within panels but at the cost of cross-panel comparability.

Strip labels (the panel headers) are themed via `labeller`; `label_both` combines variable name and level, `label_parsed` interprets the level as a math expression.

6.15.4 Assumptions

No statistical assumptions; faceting is a display convention. The only practical constraint is that the number of panels remains small enough to be readable at the final print size — usually no more than 9–12 panels per page.

6.15.5 R Implementation

```
library(ggplot2); library(palmerpenguins)
data(penguins)

# facet_wrap: one factor
ggplot(penguins, aes(flipper_length_mm, body_mass_g)) +
  geom_point(aes(colour = species), alpha = 0.7) +
  facet_wrap(~ sex) +
  theme_minimal()

# facet_grid: two factors (rows = island, cols = sex)
ggplot(penguins, aes(flipper_length_mm, body_mass_g)) +
  geom_point(aes(colour = species), alpha = 0.7) +
  facet_grid(island ~ sex) +
  theme_minimal()

# Free scales
ggplot(penguins, aes(flipper_length_mm, body_mass_g)) +
  geom_point(aes(colour = species), alpha = 0.7) +
  facet_wrap(~ species, scales = "free") +
  theme_minimal()
```

6.15.6 Output & Results

Three faceted views: a two-panel split by sex, a 3-by-3 grid by island and sex, and a three-panel split by species with free axes per panel. Each version answers a different question — “how does the relationship differ between sexes”, “across

both island and sex", or "what does it look like inside each species ignoring scale".

6.15.7 Interpretation

A reporting sentence (figure caption): "Body mass against flipper length faceted by sex (`facet_wrap(~sex)`); colour distinguishes the three penguin species." Always describe the faceting variable in the caption — readers should not have to infer the grid axis from strip labels alone.

6.15.8 Practical Tips

- Reach for `facet_wrap()` when there is one grouping variable; use `facet_grid()` only when the row/column structure is meaningful and balanced.
- Keep `scales="fixed"` whenever the cross-panel comparison is the inferential focus; switch to free scales only when within-panel magnitudes differ enough to obscure shape.
- Limit the panel count: 9–12 is the practical upper bound on a journal-sized figure; beyond that, faceting fragments rather than illuminates.
- Use `labeller=labeller(var=c("M"="Male","F"="Female"))` to spell out short factor levels for the reader.
- For aligned axes across `facet_wrap()` panels with very different ranges, fix one axis (`scales="free_y"`) rather than freeing both — this preserves comparability where it matters.
- `coord_cartesian(xlim=...,ylim=...)` zooms while leaving the underlying data unfiltered; combine with `facet_grid()` when you want fixed-scale comparisons but with a tighter view than the natural data range.

6.15.9 R Packages Used

`ggplot2` for `facet_wrap()` and `facet_grid()`; `ggh4x` for nested facets and per-panel scale control; `palmerpenguins` as a small dataset rich in faceting structure for examples.

6.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.15.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.16 Forest Plots (Visualisation)

6.16.1 Introduction

The forest plot is the signature visualisation of meta-analysis and of subgroup analyses in clinical trials. Each row shows one study (or one subgroup) as a point estimate and a horizontal confidence interval, with study name and sample size in a margin column and a pooled summary as a diamond at the bottom. The compact layout invites direct visual comparison: heterogeneity reads as horizontal scatter across rows, while consistency reads as alignment near the same vertical position. Done well, a forest plot communicates more about a meta-analysis than the accompanying numerical table.

6.16.2 Prerequisites

A working knowledge of point estimates, confidence intervals, the difference between fixed-effect and random-effects pooling, and basic `ggplot2`.

6.16.3 Theory

A forest plot encodes three quantities per study: the point estimate (a marker, with size optionally proportional to inverse-variance weight), the lower and upper confidence-interval limits (a horizontal segment), and a study label. Conventional additions:

- A vertical reference line at the null value (1 for ratios, 0 for differences).
- A diamond rather than a circle for pooled estimates, with the diamond width equal to the CI.
- Effect-size and CI text in a right-margin column for readers who need numbers as well as positions.

Ratios (HR, OR, RR) belong on a log axis so that the visual width of a CI corresponds to its multiplicative width and so that intervals symmetric on the log scale appear symmetric.

6.16.4 Assumptions

Effect estimates and confidence intervals are available and on a comparable scale across studies; weighting (if any) is computed appropriately for the meta-analytic model.

6.16.5 R Implementation

```
library(ggplot2)

studies <- data.frame(
  study = c("Trial A", "Trial B", "Trial C", "Trial D", "Pooled"),
```

```
  estimate = c(0.85, 0.70, 0.95, 0.88, 0.85),
  lo = c(0.65, 0.50, 0.75, 0.66, 0.75),
  hi = c(1.10, 0.98, 1.20, 1.16, 0.96),
  is_pooled = c(FALSE, FALSE, FALSE, FALSE, TRUE)
)

ggplot(studies, aes(x = estimate, y = factor(study, levels =
↪  rev(study)))) +
  geom_vline(xintercept = 1, linetype = "dashed", colour = "grey60") +
  geom_errorbarh(aes(xmin = lo, xmax = hi), height = 0.2) +
  geom_point(aes(size = is_pooled, shape = is_pooled), colour =
  ↪  "#2A9D8F") +
  scale_size_manual(values = c(3, 5), guide = "none") +
  scale_shape_manual(values = c(16, 18), guide = "none") +
  scale_x_log10() +
  labs(x = "Hazard ratio (log scale)", y = NULL) +
  theme_minimal()
```

6.16.6 Output & Results

A five-row forest plot: four trials displayed as circles with horizontal error bars, a pooled summary as a larger diamond at the bottom, and a vertical null line at HR = 1 on a log-x axis. The dispersion of the four trial estimates around the pooled diamond communicates between-study heterogeneity at a glance.

6.16.7 Interpretation

A reporting sentence (figure caption): “Forest plot of hazard ratios from four trials and the random-effects pooled estimate (REML, $I^2 = 23\%$); horizontal error bars show 95 % confidence intervals on a log scale, the diamond marks the pooled estimate, and the dashed vertical line corresponds to no effect (HR = 1).” Always state the pooling model and a heterogeneity statistic.

6.16.8 Practical Tips

- Log-transform ratios on the x-axis so confidence-interval widths correspond to multiplicative width; otherwise small effects look disproportionately large.
- Make marker size proportional to inverse-variance weight in meta-analysis (`size=1/SE^2`); this is the convention readers expect.
- Use a diamond for pooled estimates with width equal to the CI; `geom_polygon()` produces these cleanly.
- Order studies meaningfully — chronologically, by size, by point estimate, or by subgroup — and disclose the ordering choice in the caption.
- Add a right-margin column with numerical effect estimates and CIs (`geo`

`m_text()` at fixed x, mapped y); readers expect both visual and numeric information.
- For automated meta-analysis output, `metafor::forest.rma()` and `forestplot` produce publication-ready figures with weighted markers, columns, and heterogeneity annotations without hand-coding the `ggplot`.

6.16.9 R Packages Used

`ggplot2` for hand-built forest plots; `forestplot` for two-column publication-style forests; `metafor::forest.rma()` for meta-analysis output forests; `meta::forest()` for random-effects forests with prediction intervals; `ggforestplot` for tidy hazard-ratio displays from regression models.

6.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.16.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.17 Funnel Plots (Visualisation)

6.17.1 Introduction

A funnel plot is the canonical visualisation for diagnosing publication bias in meta-analysis. It plots each study's effect size against its precision (or its standard error), forming an inverted-funnel shape under the null of no bias: large precise studies cluster near the pooled estimate at the narrow top of the funnel, smaller imprecise studies scatter wider at the base. Asymmetry of the funnel — typically a missing wedge of small null or negative studies — suggests that studies have been suppressed for non-statistical reasons, often because the field's preferred direction is positive and significant.

6.17.2 Prerequisites

A working understanding of meta-analysis, the relationship between sample size and standard error, and the difference between heterogeneity and publication bias as competing explanations for asymmetry.

6.17.3 Theory

A funnel plot has axes:

- **x**: effect size (log odds ratio, log hazard ratio, standardised mean difference, etc.).
- **y**: standard error (with smaller SEs at the top, so the funnel narrows upward) or precision 1/SE.

Under the null of no publication bias and no small-study effects, studies fall symmetrically inside a triangular pseudo-confidence funnel centred on the pooled estimate. Asymmetry — most commonly an absence of small studies on one side — is the visual cue for bias. Formal tests quantify it: Egger's regression of the standardised effect on its precision is the most widely used; Begg–Mazumdar's rank correlation is more robust to sparse data. Trim-and-fill (`metafor::trimfill()`) imputes missing studies to estimate a bias-adjusted effect.

6.17.4 Assumptions

A meta-analysis with at least 10 studies (fewer makes asymmetry impossible to judge), study-level effects and standard errors on a comparable scale, and the working assumption that no other sources (e.g. true heterogeneity correlated with study size) explain the asymmetry.

6.17.5 R Implementation

```r
library(metafor)

# Meta-analysis of 10 hypothetical studies
dat <- data.frame(
  yi = c(-0.1, -0.3, -0.5, -0.4, -0.2, -0.6, -0.7, -0.3, -0.9, -1.2),
  vi = c(0.04, 0.03, 0.08, 0.05, 0.02, 0.09, 0.07, 0.06, 0.10, 0.12)
)
res <- rma(yi = yi, vi = vi, data = dat, method = "REML")
funnel(res, main = "Funnel plot of 10 studies")

# Egger test
regtest(res, model = "lm")
```

6.17.6 Output & Results

A funnel plot with each study marked at its (y_i, SE_i) position, a vertical line at the pooled estimate, and pseudo-95 % confidence funnel boundaries that widen as standard error increases. `regtest()` returns Egger's test slope and p-value; a significant slope is conventional evidence of asymmetry.

6.17.7 Interpretation

A reporting sentence: "The funnel plot of 10 studies showed visible asymmetry (Egger's regression slope -2.1, $p = 0.04$); trim-and-fill imputed three additional studies on the right and shifted the pooled estimate from -0.55 (95 % CI -0.78 to -0.32) to -0.42 (95 % CI -0.66 to -0.17), suggesting that publication bias inflates the apparent effect by roughly 25 %." Always pair the visual with a formal test; a single test rarely provides strong evidence on its own.

6.17.8 Practical Tips

- Require at least 10 studies before drawing conclusions from a funnel plot; fewer studies cannot support a meaningful asymmetry test.
- True between-study heterogeneity correlated with study size (smaller studies in different populations) can mimic publication-bias asymmetry; consider both explanations.
- `metafor::trimfill()` adjusts for detected asymmetry but is not a cure — it is one of several sensitivity analyses to report.
- Contour-enhanced funnel plots (`funnel(res,level=c(90,95,99))`) overlay significance contours, helping to distinguish bias from heterogeneity.
- For binary outcomes, plot on the log-OR or log-RR scale; raw OR axes distort the symmetry under the null.
- Custom `ggplot2` funnel plots are easy to build (`geom_point()` plus `geom_segment()` for the funnel boundaries) when journal style requires consistent typography across figures.

6.17.9 R Packages Used

`metafor` for `rma()`, `funnel()`, `regtest()`, and `trimfill()`; `meta` for `metabias()` and alternative funnel implementations; `ggplot2` for hand-built funnel plots when the default base-graphics output does not match a publication's style; `dmetar` for additional bias diagnostics.

6.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.17.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.18 ggplot2 Themes

6.18.1 Introduction

A `ggplot2` theme controls everything in a plot that is not the data itself — panel background, axis ticks, gridlines, font sizes, legend placement, plot margins. Two layers of control coexist: a **complete theme** (`theme_minimal()`, `theme_bw()`, `theme_classic()`, etc.) sets every non-data element at once and is usually the right starting point, while individual `theme()` calls override specific elements for publication polish. Mastering both layers is what turns a default plot into one that looks at home in a journal article or a polished report.

6.18.2 Prerequisites

A working knowledge of `ggplot2`'s grammar of graphics, layered geoms, and aesthetic mappings.

6.18.3 Theory

Complete themes are functions that return a fully populated `theme` object. Built-in choices include `theme_minimal()` (white background, light gridlines, the de facto modern default), `theme_bw()` (white panel with thin black border), `theme_classic()` (axes only, no gridlines, mimicking print journals), and `theme_void()` (no axes or gridlines, useful for maps and decorative composites). The `ggthemes` package adds journal-flavoured options (`theme_tufte()`, `theme_economist()`, `theme_few()`).

Element overrides come from `theme()`, which accepts named arguments for every theme element. Each element is built from one of four constructors: `element_text()` for text (size, family, face, colour, hjust, vjust, margin), `element_line()` for lines, `element_rect()` for rectangles, and `element_blank()` to hide an element entirely. Common overrides include `legend.position`, `axis.title`, `panel.grid.minor`, and `plot.title`.

`theme_set()` applies a theme globally to every subsequent plot in the session, useful for enforcing house style across a multi-figure report.

6.18.4 Assumptions

No statistical assumptions; the only practical constraint is that font sizes, line widths, and margins must remain legible at the final print size, which is best verified by saving the plot at the target dimensions before judging it on screen.

6.18.5 R Implementation

```
library(ggplot2); library(ggthemes)
```

```r
p <- ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) +
  geom_point(size = 3)

# Complete themes
p + theme_minimal()
p + theme_bw()
p + theme_classic()
p + theme_tufte()

# Custom theme
p +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 12),
    legend.position = "bottom",
    panel.grid.minor = element_blank()
  ) +
  labs(title = "MPG vs weight", x = "Weight (1000 lbs)", y = "MPG",
       colour = "Cylinders")

# Set a global theme
theme_set(theme_minimal(base_size = 12))
```

6.18.6 Output & Results

Side-by-side comparisons of the same scatterplot under four complete themes plus a fully customised version. The custom variant combines `theme_minimal()` as the base with overrides for title weight, axis title size, legend position, and gridline visibility — a typical recipe for a publication-ready figure.

6.18.7 Interpretation

A reporting sentence (figure caption): "All figures use `theme_minimal(base_size=12)` with custom title and axis label sizes; legend position was moved to the bottom for horizontal compositions." Mention the theme only when it affects interpretation (e.g. removing gridlines deliberately to emphasise a contrast); otherwise it can be left to a methods section "figure preparation" line.

6.18.8 Practical Tips

- Set a global theme once at the top of your analysis script via `theme_set()`; the rest of the figures inherit it automatically and stay consistent.
- For biomedical publications, `theme_minimal(base_size=12)` with `panel.grid.minor=element_blank()` is a sensible default; it reads well in print and on screen.

- Use `element_blank()` to hide an element entirely; setting `colour=NA` only makes it invisible while still occupying its space.
- Keep font choices simple: `theme(text=element_text(family="Helvetica"))` (or the corresponding sans-serif default) avoids the rendering surprises that come with fancier typefaces.
- Always export with `ggsave()` at the final intended `width`, `height`, and `dpi` (300 for raster, 600+ for line art); judging a plot on a screen at the wrong dimensions hides legibility problems.
- For bespoke journal styles, build a wrapper function that returns a fully configured theme and use it instead of repeating the same `theme()` overrides in every script.

6.18.9 R Packages Used

`ggplot2` for the base theme system, `ggthemes` for journal- and Tufte-flavoured presets, `cowplot` for `theme_cowplot()` (a publication-oriented variant), and `extrafont` if you need access to non-default system fonts in PDF output.

6.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.18.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.19 Saving and Exporting Figures

6.19.1 Introduction

The final step of any figure pipeline is exporting the plot at the right dimensions and resolution. It is also the step where most figures go wrong: a plot designed and previewed at 1024×768 pixels on screen looks fine until it is dropped into a manuscript and reprinted at journal column width, where the fonts have shrunk to illegibility and the raster edges have blurred. `ggsave()` is `ggplot2`'s export function, and the discipline to invoke it with explicit dimensions and an explicit DPI early in the workflow is what separates publication-ready figures from ones that have to be re-made the night before submission.

6.19.2 Prerequisites

A working knowledge of `ggplot2`, an idea of what the final figure will be used for (print, web, slide), and basic familiarity with vector vs raster graphics.

6.19.3 Theory

File-format choice depends on the destination:

- **PDF** is vector and infinite-resolution; it is the standard for journal submissions and the safest default unless told otherwise.
- **SVG** is also vector and ideal when the figure will be edited in Illustrator or Inkscape before final use.
- **PNG** is raster with lossless compression; use 300 dpi or higher for print, 96 dpi for web.
- **TIFF** is raster and required by several biomedical journals; use lossless `compression="lzw"` to avoid quality loss.
- **EPS** is vector and required by a shrinking minority of journals; PDF is generally a better choice when allowed.

Dimensions are specified through `width` and `height` with units (`"in"`, `"cm"`, `"mm"`); DPI applies only to raster formats. Font sizes scale with `base_size` in the theme; setting `base_size=12` against a 6×4 inch figure produces fonts that print legibly at journal column width.

6.19.4 Assumptions

The graphics device for the chosen format is available (Cairo for PNG with anti-aliasing, `ragg` for high-quality raster output, `svglite` for compact SVG); modern R installations include all of these but verify if a specific format fails.

6.19.5 R Implementation

```
library(ggplot2)

p <- ggplot(mtcars, aes(wt, mpg)) +
  geom_point(size = 3) +
  theme_minimal(base_size = 12)

# Standard PDF for journal submission
ggsave("figure1.pdf", p, width = 6, height = 4, units = "in")

# High-resolution PNG
ggsave("figure1.png", p, width = 6, height = 4, units = "in", dpi = 300)

# Journal-standard column width (typically 85 mm for single column)
ggsave("figure1.tiff", p, width = 85, height = 60, units = "mm",
```

```
       dpi = 300, compression = "lzw")

# Vector SVG for editing
ggsave("figure1.svg", p, width = 6, height = 4)
```

6.19.6 Output & Results

Four export variants of the same `ggplot` object: a PDF for journal submission, a 300-dpi PNG for web embedding, an LZW-compressed TIFF at 85 mm single-column width, and an SVG for downstream editing. Each file is identical in content but differs in format-specific properties.

6.19.7 Interpretation

A reporting sentence (methods or figure-preparation note): "All figures were rendered with `ggplot2` 3.5 (`base_size=12`) and exported to PDF (vector) and TIFF (LZW-compressed, 300 dpi) at single-column width (85 mm); raster previews use the `ragg` device for anti-aliased text." Documenting the export pipeline aids reproducibility and pre-empts reviewer questions about figure resolution.

6.19.8 Practical Tips

- Always export at the final publication size; resizing a saved figure in a word processor distorts fonts and produces blurry raster edges.
- For multi-panel figures composed with `patchwork`, export the composed object directly; do not save panels separately and stitch them in an editor.
- Use `ragg::agg_png()` (or set `options(ragg.background="white")`) for anti-aliased PNG output that handles UTF-8 and emoji correctly.
- For TIFF submissions, `compression="lzw"` produces lossless files at typically half the size of uncompressed; some journals also accept `"zip"`.
- Save a small reproducible script next to every published figure (`figure 1.R`); regenerating a figure six months later is otherwise an exercise in archaeology.
- For vector figures with thousands of overlapping elements (large hexbins, dense scatter), file sizes balloon; rasterise the busy layer with `ggrastr::rasterise()` while keeping axes and text vector for a hybrid output that prints cleanly.

6.19.9 R Packages Used

`ggplot2` for `ggsave()`; `ragg` for fast, high-quality raster devices; `svglite` for compact SVG; `Cairo` for legacy anti-aliased rendering; `ggrastr` for selectively rasterising vector layers in otherwise vector exports.

6.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.19.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.20 The Grammar of Graphics

6.20.1 Introduction

`ggplot2` is the dominant plotting system in R because it is built on a principled grammar rather than an ad-hoc collection of plot types. Every plot is constructed from the same seven building blocks: data, aesthetic mappings, geometric objects (geoms), statistical transformations, scales, coordinate systems, and faceting. Once the grammar is internalised, the vast library of plot types reduces to a small set of decisions in a common vocabulary.

6.20.2 Prerequisites

The reader should know R vectors, data frames, and the `%>%` or `|>` pipe. No prior `ggplot2` knowledge is assumed.

6.20.3 Theory

The layered grammar expresses every plot as a stack of layers added to a base specification. Each layer may override or inherit data and mappings from the base. The seven components:

- **Data**: a data frame whose rows are observations and whose columns are variables.
- **Mappings**: assignments from variables in the data to visual aesthetics (x-position, y-position, colour, size, shape, alpha, etc.).
- **Geoms**: the geometric objects drawn on the plot – points, lines, bars, polygons, text.
- **Stats**: statistical transformations applied to the data before drawing (identity, binning for histograms, smoothing for trend lines, summary for means with error bars).
- **Scales**: the functions that translate data values into aesthetic values – a continuous scale for a numeric x-axis, a discrete ordinal scale for a factor colour mapping.

- **Coord**: the coordinate system, usually Cartesian but occasionally polar, map projections, or fixed aspect ratios.
- **Facets**: subplots defined by one or more categorical variables, laid out in rows, columns, or a grid.

A plot is rendered by evaluating the stats on the data, mapping the results through the scales to aesthetic values, drawing the geoms in the chosen coordinate system, and replicating the whole thing across facets.

6.20.4 R Implementation

The `palmerpenguins` package provides a tidy biological dataset for demonstration:

```
library(ggplot2)
library(palmerpenguins)

ggplot(data = penguins,
       mapping = aes(x = flipper_length_mm,
                     y = body_mass_g,
                     colour = species)) +
  geom_point(alpha = 0.7, size = 2) +
  geom_smooth(method = "lm", se = TRUE) +
  scale_colour_brewer(palette = "Set2") +
  facet_wrap(~ sex, nrow = 1) +
  labs(x = "Flipper length (mm)",
       y = "Body mass (g)",
       colour = "Species",
       title = "Body mass vs. flipper length by species and sex") +
  theme_minimal(base_size = 12) +
  theme(legend.position = "bottom")
```

This single plot uses all seven grammar components: the data is `penguins`, the mappings send `flipper_length_mm` to x, `body_mass_g` to y, and `species` to colour; two geoms (point and smooth) are layered; the `lm` stat fits a linear trend within each species; the Brewer scale converts the species factor to colours; the Cartesian coordinate system is implicit; faceting by sex produces a panel per level.

6.20.5 Output & Results

The plot shows three colour-coded point clouds in each of two panels (one per sex). Linear trends are drawn within each species, and the Brewer "Set2" palette is colour-blind-safe. Axis labels and a title provide context; the legend is placed at the bottom to give maximum plot area to the data.

6.20.6 Interpretation

The layered-grammar view turns plot design into a checklist:

1. What data?
2. Which variables go to which aesthetics?
3. What geoms best display the relationship?
4. Are any statistical transformations needed?
5. Which scales (colour palette, axis transformation) best carry the intended comparison?
6. Coordinate system – usually Cartesian, but check.
7. Faceting – does the story separate across a grouping variable?

Going through these questions in order produces plots that are intentional rather than stumbled into.

6.20.7 Practical Tips

- Start with `geom_point()` and add layers; do not try to build the final plot in one shot.
- Map aesthetics inside `aes()` when they depend on data; set them outside `aes()` (e.g. `colour="steelblue"`) when they are constant.
- Use colour-blind-safe palettes by default: `scale_colour_viridis_d()`, `scale_colour_brewer()`, or the `colorblindr` package.
- Export figures with `ggsave(filename,width=7,height=5,dpi=300)` to get publication-grade output. Most journals accept PDF or TIFF at 300+ DPI.
- Avoid dual-y-axis plots unless the two scales are genuinely comparable; they are almost always misleading.

6.20.8 For Reviewers

Use the shared reviewer checklist, together with this method's theory and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.20.9 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.21 Heatmaps

6.21.1 Introduction

A heatmap displays a matrix as a grid of coloured cells, with numerical values encoded by colour. The geom is a workhorse across genomics (gene-expression matrices), clinical research (cross-tabulations of conditions and measurements), classifier evaluation (confusion matrices), and exploratory data analysis (correlation matrices). Two design choices dominate readability: the colour palette (sequential, diverging, or qualitative) and the row / column order, which often comes from hierarchical clustering on rows, columns, or both.

6.21.2 Prerequisites

A working understanding of matrices, hierarchical clustering, and colour palette selection (sequential vs diverging).

6.21.3 Theory

Two ecosystems handle heatmaps in R:

- **`ggplot2` + `geom_tile()`**: flexible but requires long-format data and manual handling of clustering. Best when the heatmap is one panel in a larger composition.
- **`pheatmap` and `ComplexHeatmap`**: dedicated packages that handle clustering, dendrograms, row/column annotations, and z-scoring natively. Best when the heatmap is the figure and customisation focus is on annotation tracks.

The colour scale must match the data semantics: sequential (`viridis`, `Blues`) for ordered values without a meaningful zero, diverging (`RdBu`, `BrBG`) for z-scored data or any quantity centred on zero. Heatmaps with a one-sided palette applied to centred data are misleading because the visual midpoint does not match the numerical midpoint.

6.21.4 Assumptions

A matrix or matrix-like structure with rows and columns that admit a sensible reordering. For clustering, rows and columns must be on comparable scales (or pre-scaled per row).

6.21.5 R Implementation

```
library(ggplot2)

# ggplot tile heatmap
d <- as.data.frame(scale(as.matrix(mtcars)))
d$car <- rownames(d)
```

```
d_long <- reshape2::melt(d, id.vars = "car")
ggplot(d_long, aes(variable, car, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red",
                       midpoint = 0) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# pheatmap with hierarchical clustering
library(pheatmap)
pheatmap(scale(mtcars), clustering_method = "ward.D2",
         color = colorRampPalette(c("blue", "white", "red"))(100))
```

6.21.6 Output & Results

Two heatmaps of the z-scored `mtcars` matrix: a hand-built `ggplot2` tile heatmap (no clustering, alphabetical row order) and a `pheatmap` version with Ward's hierarchical clustering on both rows and columns plus accompanying dendrograms. The clustered version reveals two clear groups of cars distinguished by their efficiency-versus-power profile; the unclustered version requires the reader to find that structure unaided.

6.21.7 Interpretation

A reporting sentence (figure caption): "Heatmap of z-scored vehicle attributes ($n = 32$ cars $\times$ 11 variables); colour encodes z-score on a blue-white-red diverging scale centred at zero, and rows and columns are clustered with Ward's method on Euclidean distance." Always state the scaling, the distance metric, and the linkage method; results depend on all three.

6.21.8 Practical Tips

- Centre and scale rows (z-score per row) before clustering whenever variables are on different scales; otherwise, large-magnitude variables dominate the distance metric.
- Use a diverging palette with an explicit midpoint for z-scored data; one-sided sequential palettes collapse the visual midpoint and obscure structure.
- For large matrices (thousands of rows or columns), `ComplexHeatmap` scales better than `pheatmap` and supports complex annotation tracks (categorical, continuous, mark-driven).
- Add side annotations (group, condition, batch) to help readers spot whether clustering tracks biology or technical artefacts.
- Cluster only the dimensions that should be ordered; force the other dimension's order when biological or temporal meaning matters.

- Use `cluster_rows=FALSE` (or `cluster_cols=FALSE`) when an existing ordering carries information you want preserved (e.g. chromosome position).

6.21.9 R Packages Used

`ggplot2` for `geom_tile()` and flexible composition; `pheatmap` for one-call heatmaps with clustering; `ComplexHeatmap` for large-scale heatmaps with rich annotations; `superheat` for an alternative ggplot-flavoured heatmap with linked summary plots; `viridis` and `RColorBrewer` for palette selection.

6.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.21.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.22 Hexbin Plots

6.22.1 Introduction

A scatterplot of a few hundred points is informative; a scatterplot of a few hundred thousand points is mostly black ink. Once n exceeds a few thousand, individual points overplot to the point of hiding density structure: ridges, modes, and non-linear boundaries vanish under the cloud. Hexbin plots solve this by dividing the plane into hexagonal cells, counting points per cell, and colour-encoding the count. The result preserves the joint distribution at any sample size and renders quickly even for millions of observations.

6.22.2 Prerequisites

A working knowledge of scatterplots, basic 2D density estimation, and `ggplot2`'s aesthetic mappings.

6.22.3 Theory

Hexagonal bins tile the plane more efficiently than squares because every hexagon has six equidistant neighbours; this isotropy reduces the visual aliasing that affects rectangular bins along diagonal density boundaries. `ggplot2::geom_hex()` calls the `hexbin` package internally to construct the bins and aggregate

counts. The `bins` argument sets the resolution along each axis; `binwidth` is an alternative that fixes the cell size in data units. The colour scale should respect the count distribution: linear scales for roughly uniform counts, logarithmic scales (`scale_fill_viridis_c(trans="log")`) for highly skewed counts that span orders of magnitude.

6.22.4 Assumptions

Both x and y are continuous; categorical variables need a different geom (heatmap, mosaic, or jitter). The plot assumes the joint distribution is the inferential target, not individual points — for one-off outliers, overlay a labelled `geom_point()`.

6.22.5 R Implementation

```r
library(ggplot2); library(hexbin)

# Diamonds dataset: 54 000 observations
ggplot(diamonds, aes(carat, price)) +
  geom_hex(bins = 40) +
  scale_fill_viridis_c() +
  theme_minimal()

# Log-scale for skewed data
ggplot(diamonds, aes(carat, price)) +
  geom_hex(bins = 40) +
  scale_x_log10() +
  scale_y_log10() +
  scale_fill_viridis_c(trans = "log") +
  theme_minimal()
```

6.22.6 Output & Results

A 40-by-40 hexbin grid showing the joint distribution of carat and price across 54,000 diamonds. The linear-scale version emphasises the dense middle of the distribution; the log-log version with log-scaled fill reveals the sparse tails — small low-priced diamonds and large high-priced ones — that are invisible at the original scale.

6.22.7 Interpretation

A reporting sentence (figure caption): "Hexbin density of diamond carat versus price (n = 53,940), 40 bins per axis, fill on a log-count scale; both axes are log-transformed to expose the price-vs-carat relationship across the full range." Always state the bin count and any transformations, since both shape the visual conclusion.

6.22.8 Practical Tips

- `bins=30` to `bins=60` is a sensible default for tens of thousands of observations; finer grids reveal more detail at the cost of noisier counts per cell.
- Log-transform the fill scale when counts span orders of magnitude; otherwise the dense centre saturates the palette and the tails vanish.
- For square bins, use `geom_bin2d()`; the choice between hex and square is largely aesthetic, but hex bins reduce diagonal-aliasing artefacts.
- Overlay a smoother (`geom_smooth(method="gam")`) on top of the hexbin for a model-based summary; the bin colour shows where the smoother is well-supported.
- For extremely large n (millions of points), hexbin remains fast and effective where individual scatter would crash the renderer; the underlying counts are stored as a small array regardless of input size.
- Below a few thousand points, prefer alpha-blended scatter or 2D density contours; hexbin is overkill for small samples and reads as cluttered.

6.22.9 R Packages Used

`ggplot2` for `geom_hex()` and `geom_bin2d()`, `hexbin` for the underlying hexagonal binning, `viridis` for perceptually uniform fill scales, and `ggdensity` if you want to combine binning with model-based density contours.

6.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.22.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.23 Histograms

6.23.1 Introduction

A histogram is the simplest distributional display: bin the range of a continuous variable, count the observations in each bin, and draw bars proportional to those counts. It is the canonical first plot of any exploratory data analysis. It is also the geom most often misled by its own settings — the impression of "skewness" or "bimodality" can be conjured or hidden entirely by the choice of bin width, so

a good histogram is one inspected at several resolutions before being committed to print.

6.23.2 Prerequisites

A working knowledge of continuous variables, the difference between counts and densities, and `ggplot2` aesthetic mappings.

6.23.3 Theory

Three choices shape every histogram:

- **Bin count** (or `binwidth`). Too few bins hide modes and skew; too many exaggerate sampling noise. Common rules of thumb: Sturges ($\log_2 n + 1$, optimal for Normal data of moderate size), Freedman–Diaconis (binwidth $= 2 \cdot \mathrm{IQR}/n^{1/3}$, robust to outliers), Scott ($3.5\sigma/n^{1/3}$, optimised for Normal data).
- **Frequency vs density y-axis**. Frequency (raw count) is the default; density (count divided by binwidth, integrating to 1) is essential for comparing groups of different sample sizes or overlaying with a density curve.
- **Bin boundaries**. Where the first bin starts affects the appearance, especially for small samples; this is the so-called "boundary effect", invisible in dense data and obvious in sparse.

The shape impression depends on all three; a histogram should always be reproduced with at least two bin counts before any conclusion is drawn.

6.23.4 Assumptions

A continuous (or discrete-but-numerous) variable. Categorical data should go to bar charts; small ordinal scales (Likert) often look better as bar charts even if technically histograms apply.

6.23.5 R Implementation

```
library(ggplot2)

# Basic histogram
ggplot(mtcars, aes(mpg)) +
  geom_histogram(bins = 15, fill = "#2A9D8F", colour = "white") +
  theme_minimal()

# Density y-axis for group comparison
ggplot(iris, aes(Sepal.Length, fill = Species)) +
  geom_histogram(aes(y = after_stat(density)),
                 position = "identity", alpha = 0.5, bins = 20) +
  scale_fill_brewer(palette = "Set2") +
```

```r
  theme_minimal()

# Automatic binwidth via Freedman-Diaconis
ggplot(diamonds, aes(price)) +
  geom_histogram(binwidth = 2 * IQR(diamonds$price) /
   ↪  nrow(diamonds)^(1/3),
                 fill = "#2A9D8F", colour = "white") +
  theme_minimal()
```

6.23.6 Output & Results

Three histograms: the 32-row `mtcars` mileage with 15 bins, a three-species iris overlay on a density y-axis (so the groups are comparable despite identical sample sizes), and a Freedman–Diaconis-binwidth view of diamond prices that handles the long right tail more gracefully than a default 30-bin choice.

6.23.7 Interpretation

A reporting sentence (figure caption): "Distribution of fuel efficiency (mpg) across 32 cars, 15 equal-width bins covering the range 10–35 mpg." Always state the bin count or binwidth in the caption; a histogram without that information is impossible to reproduce.

6.23.8 Practical Tips

- Always inspect with several bin counts before committing to one; modes that survive a doubling of bins are real, modes that disappear were artefacts.
- Use `after_stat(density)` (formerly `..density..`) when overlaying multiple groups of different sizes; raw counts conflate distribution shape with sample-size differences.
- For long right tails (waiting times, prices, gene expression), log-transform the x-axis (`scale_x_log10()`) before binning; the resulting histogram reveals structure invisible on the linear scale.
- For small samples ($n < 30$) prefer a dot plot or strip plot; a histogram of 30 points binned into 10 bars is mostly empty.
- Overlay `geom_density()` (with `aes(y=after_stat(density))`) for a smooth summary that complements the bins; the two together communicate both the discrete bins and the smooth shape.
- Avoid mixing colour (group) with fill aesthetics when the same column drives both; pick one and let the legend speak for itself.

6.23.9 R Packages Used

`ggplot2` for `geom_histogram()`; `scales` for x-axis label formatters when the data are on currency or large-number scales; `ggdensity` for combined histogram-and-density displays; `ggdist` for modern slabinterval geoms that often replace histograms entirely in posterior visualisations.

6.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.23.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.24 Interactive Plots with ggiraph

6.24.1 Introduction

Where `plotly` translates `ggplot2` figures into a separate JavaScript engine, `ggiraph` keeps the original `ggplot2` rendering and adds SVG event handlers to individual elements — points, bars, polygons. The result is interactivity that preserves `ggplot2`'s native typography, themes, and palettes, with sharp vector output that scales cleanly to print and screen alike. Tooltips, click events, hover highlighting, and Shiny linkage come from a small set of `*_interactive` geoms that mirror their `ggplot2` counterparts.

6.24.2 Prerequisites

A working knowledge of `ggplot2` and an output environment that supports HTML — Quarto, RMarkdown, Shiny, or any browser-based viewer.

6.24.3 Theory

Each interactive geom (`geom_point_interactive()`, `geom_col_interactive()`, `geom_polygon_interactive()`, etc.) accepts additional aesthetics on top of its `ggplot2` base:

- `tooltip`: text (or HTML) shown on hover.
- `data_id`: identifier used for cross-element highlighting and selection.
- `onclick`: arbitrary JavaScript invoked when the element is clicked.

The `girafe()` function wraps the resulting plot into an HTML widget; `opts_*` helpers tune hover CSS, selection behaviour, zoom, toolbar visibility, and download options. Because the output is plain SVG, every interactive plot remains a standalone vector graphic that can be exported, embedded, or re-styled with external CSS.

6.24.4 Assumptions

The output target supports HTML and SVG rendering. For PDF or PNG output, `ggiraph` falls back to a static SVG snapshot — interactivity is lost but typography is preserved, unlike `plotly`'s rasterised exports.

6.24.5 R Implementation

```
library(ggplot2); library(ggiraph)

p <- ggplot(mtcars, aes(wt, mpg,
                        tooltip = rownames(mtcars),
                        data_id = rownames(mtcars))) +
  geom_point_interactive(size = 3, colour = "#2A9D8F") +
  theme_minimal()

girafe(ggobj = p,
       options = list(opts_hover(css = "fill:#F4A261;stroke:black"),
                      opts_selection(type = "multiple")))
```

6.24.6 Output & Results

An interactive SVG scatterplot in which hovering a point highlights it with the configured CSS rule, clicking selects it, and shift-clicking adds to the selection. Inside Shiny, the selection propagates to the `input$<plot_id>_selected` reactive, enabling brushed cross-filtering between charts and tables.

6.24.7 Interpretation

A reporting sentence: "Figure 2 (rendered with `ggiraph`): each point shows a vehicle; hover reveals the model name and selected points propagate to the linked summary table below." Mention the technology and any selection semantics in the caption — readers should know whether interactivity is decorative or part of the analytical workflow.

6.24.8 Practical Tips

- Each `geom_*_interactive()` accepts the same arguments as its non-interactive parent plus the new aesthetics, so refactoring an existing plot is a one-word swap per layer.

- Use `data_id` to group elements that should highlight together (all points from one country, all bars in one facet); the SVG event system propagates by `data_id` automatically.
- Inside Shiny, swap `plotOutput()` and `renderPlot()` for `girafeOutput()` and `renderGirafe()`; selection and hover events become reactive inputs without additional plumbing.
- Customise hover and selection CSS with `opts_hover(css=...)` and `opts_selection(css=...)`; the same plot can present very different interactive feedback depending on context.
- Avoid `ggiraph` for plots with many tens of thousands of elements; SVG renders every element as a DOM node, and browsers slow down beyond a few thousand. For very large data, hexbin or canvas-rendered alternatives (`scattermore`, `plotly` with `scattergl`) scale better.
- For static publication, the same `ggplot` object renders without modification through `ggsave()` — write your interactive code over a normal `ggplot` and you get both versions for free.

6.24.9 R Packages Used

`ggiraph` for the interactive geoms, `girafe()`, and the `opts_*` helpers; `ggplot2` for the underlying grammar; `htmlwidgets` for embedding standalone widgets; `shiny` when reactive selection events drive downstream computation.

6.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.24.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.25 Interactive Plots with plotly

6.25.1 Introduction

A static figure is what a journal prints; an interactive figure is what a colleague needs when exploring a result on a screen. `plotly::ggplotly()` converts an existing `ggplot` object into an HTML widget that supports hover tooltips, drag-to-zoom, panning, and selection brushing — without requiring any JavaScript. The result fits naturally into Quarto and RMarkdown reports, Shiny dashboards,

and standalone HTML pages, and gives reviewers and collaborators a way to interrogate the data without exporting CSVs.

6.25.2 Prerequisites

A working knowledge of `ggplot2` (geoms, aesthetics, themes) and an output context that supports HTML rendering — Quarto, RMarkdown, Shiny, or RStudio's Viewer pane.

6.25.3 Theory

`ggplotly()` walks the structure of a `ggplot` object — its layers, scales, and aesthetic mappings — and translates each component into the corresponding `plotly.js` primitives. Hover tooltips default to whatever aesthetic mappings are in scope; custom tooltips come from mapping `aes(text=...)` and passing `tooltip="text"` to `ggplotly()`. The translation is faithful for most common geoms (point, line, bar, ribbon, density), partial for some statistical layers (`geom_smooth`, faceting with free scales), and weakest for non-standard extensions where you may need to drop down to native `plot_ly()` syntax.

6.25.4 Assumptions

The output environment renders HTML. Static targets (PDF, PNG, EPS) lose all interactivity, so for journal submission you should retain the original `ggplot` and re-render it with `ggsave()`; `plotly` is for screens, not paper.

6.25.5 R Implementation

```
library(ggplot2); library(plotly)

p <- ggplot(mtcars, aes(wt, mpg, colour = factor(cyl),
                        text = rownames(mtcars))) +
  geom_point(size = 3) +
  theme_minimal()

ggplotly(p, tooltip = "text")

# Link two plots
p2 <- ggplot(mtcars, aes(hp, mpg)) + geom_point()
subplot(ggplotly(p), ggplotly(p2))
```

6.25.6 Output & Results

An interactive HTML scatter where each marker shows the car name on hover, the legend is clickable to toggle cylinder groups, and drag-zoom narrows the axes.

`subplot()` arranges multiple `ggplotly` objects side by side with synchronised axes — useful for cross-filtering between related views.

6.25.7 Interpretation

A reporting sentence: "Figure 2 is an interactive scatterplot rendered with `plotly::ggplotly()`; hovering over a marker shows the car name and cylinder count, and clicking a legend entry toggles the corresponding group." Mention the technology in the figure caption when interactivity carries part of the figure's meaning so static fallbacks (a screenshot for the print version) are not mistaken for the whole story.

6.25.8 Practical Tips

- Render the static `ggplot` and the `ggplotly` version from the same source; that way the printed and online versions stay in sync.
- For custom tooltips, include only the variables that matter: too many columns under the cursor become a wall of text. `aes(text=paste("Name:",rownames(data),"<br>HP:",hp))` is a common pattern.
- `subplot()` combines multiple `ggplotly` objects with optional shared axes; for many panels, build them via `lapply()` and combine with `do.call(subplot,...)`.
- Performance degrades beyond a few thousand points; switch to `plot_ly(...,type="scattergl")` for WebGL rendering when working with tens of thousands.
- Inside Shiny, swap `plotOutput()` / `renderPlot()` for `plotlyOutput()` / `renderPlotly()` and the rest of the app remains unchanged.
- Use `config(displayModeBar=FALSE)` to suppress the toolbar for embedded reports where the toolbar is more clutter than feature.

6.25.9 R Packages Used

`plotly` for `ggplotly()`, `plot_ly()`, `subplot()`, and the JavaScript bindings; `ggplot2` for the static plot underlying the conversion; `htmlwidgets` for embedding the result in standalone HTML pages.

6.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.25.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.26 Line Plots

6.26.1 Introduction

Line plots connect ordered observations and exploit a powerful perceptual cue: the human eye reads slope as rate of change effortlessly. They are the default geom for time-series, dose-response curves, individual-trajectory ("spaghetti") displays, cumulative distributions, and any context where the x-axis carries a meaningful sequence. The same simplicity that makes them effective also makes them dangerous when the x-axis lacks order — connecting unordered categories with a line implies a trajectory that does not exist and misleads readers into seeing trends in noise.

6.26.2 Prerequisites

A working knowledge of `ggplot2`'s aesthetic mappings, a clear distinction between continuous, ordered, and unordered variables, and an understanding of how `group` aesthetics control which points get joined.

6.26.3 Theory

`geom_line()` joins points in the order they appear within each group defined by the `group` aesthetic; default groups come from any discrete colour, fill, or linetype mapping. For paired or repeated-measures data, the `group` aesthetic must be set explicitly to the subject identifier so that individual trajectories are connected, not all observations across subjects. Variants include:

- `geom_step()` for cumulative quantities such as ECDFs or Kaplan–Meier survival curves, where the underlying process changes only at observed events.
- `geom_path()` for trajectories ordered by row position rather than the x-aesthetic, useful for state-space plots.
- `geom_ribbon()` paired with a line for confidence or prediction bands.

For many overlapping lines, alpha-blending or faceting prevents the "spaghetti pile" that hides individual trajectories.

6.26.4 Assumptions

The x-axis values are ordered meaningfully — by time, dose, position, or rank. Connecting nominal categories with a line is a presentation error, not a stylistic choice.

6.26.5 R Implementation

```r
library(ggplot2)

# Single time series
ggplot(economics, aes(date, unemploy)) +
  geom_line(colour = "#2A9D8F", linewidth = 1) +
  theme_minimal()

# Multiple groups
ggplot(airquality, aes(Day, Ozone, group = Month, colour =
 ↪  factor(Month))) +
  geom_line(linewidth = 1) +
  scale_colour_brewer(palette = "Set2") +
  theme_minimal()

# Paired trajectories (pre/post style)
df <- data.frame(
  subject = rep(1:20, each = 2),
  time    = rep(c("pre", "post"), 20),
  value   = c(rnorm(20, 50, 10), rnorm(20, 55, 10))
)
ggplot(df, aes(time, value, group = subject)) +
  geom_line(colour = "grey70") +
  geom_point(size = 2) +
  theme_minimal()

# Step line (cumulative)
ggplot(mtcars, aes(wt)) +
  stat_ecdf(geom = "step", colour = "#2A9D8F", linewidth = 1) +
  theme_minimal()
```

6.26.6 Output & Results

Four illustrative line plots: a long-running unemployment series, monthly ozone levels coloured by month, individual subject trajectories from pre to post, and the empirical cumulative distribution function of car weights as a step line. Each shows a different idiomatic use of the line geom.

6.26.7 Interpretation

A reporting sentence: “Individual subject responses from pre to post are connected by grey lines, with overlaid points indicating measurement times; group-mean changes (not shown) are reported in Table 1.” Always describe the unit of analysis a line represents (subject, country, replicate); a line plot without that context is ambiguous.

6.26.8 Practical Tips

- Set the `group` aesthetic explicitly for paired or repeated-measures data; the default group inferred from colour or linetype rarely matches the intended unit.
- Add `geom_point()` on top of `geom_line()` when x-values are sparse so individual observations remain visible; pure lines hide irregular sampling.
- For uncertainty bands, layer `geom_ribbon(aes(ymin=lo,ymax=hi),alpha=0.3)` *before* the line so the line draws on top.
- Use `geom_step()` for ECDFs and survival curves — they are step functions, not smooth ones, and a smooth interpolation misrepresents the underlying mathematics.
- For dozens of overlapping trajectories, lower `alpha` (0.2–0.4) and add a single bold line for the group mean or median; pure spaghetti loses readability above 30–40 trajectories.
- When the y-axis spans orders of magnitude with multiplicative growth, use `scale_y_log10()` so constant-rate growth reads as a straight line.

6.26.9 R Packages Used

`ggplot2` for `geom_line()`, `geom_step()`, `geom_path()`, `geom_ribbon()`, and `stat_ecdf()`; `ggrepel` for non-overlapping line-end labels; `directlabels` for legend-free per-series labelling that scales better than colour-and-legend in print.

6.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.26.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.27 Pairs Plots

6.27.1 Introduction

A pairs plot — also called a scatterplot matrix — shows every pairwise scatter for a set of continuous variables in a single grid figure. Modern variants enrich the matrix with marginal histograms or densities along the diagonal and correlation coefficients (or other pairwise summaries) in the upper triangle. The result is the single most efficient exploratory diagnostic for a small-to-moderate

set of variables: it surfaces non-linearity, outliers, subgroup separation, and multicollinearity in one pass.

6.27.2 Prerequisites

A working knowledge of scatter plots, correlation, and group-coloured aesthetics; familiarity with the `GGally` package's extensions to `ggplot2`.

6.27.3 Theory

For p variables the matrix has p^2 cells. The diagonal shows the marginal distribution of each variable (histogram, density, or a labelled bar for factors). The lower triangle shows pairwise scatter plots; the upper triangle shows summary statistics — most commonly Pearson, Spearman, or Kendall correlations, optionally annotated by significance. `GGally::ggpairs()` is the canonical implementation, with arguments `lower`, `upper`, and `diag` each accepting a list of cell-type specifications. Group structure is added via `aes(colour=group)` which propagates through every panel and provides immediate insight into between-group differences.

6.27.4 Assumptions

Each pairwise cell inherits the assumptions of the relevant geom: scatter plots assume continuous data; correlation cells assume meaningful linear (or rank) association. Categorical variables are displayed with bar charts and contingency tables in the off-diagonal cells.

6.27.5 R Implementation

```
library(GGally)

# Basic pairs plot
ggpairs(iris, columns = 1:4, aes(colour = Species, alpha = 0.5)) +
  theme_minimal()

# Custom lower-triangle geoms
ggpairs(mtcars, columns = c("mpg", "wt", "hp", "qsec"),
        lower = list(continuous = "smooth"))
```

6.27.6 Output & Results

Two pairs plots: a four-variable matrix of the iris dataset coloured by species, and a four-variable matrix of `mtcars` numerical columns with LOESS smoothers in the lower triangle. The first reveals the species-driven separation across nearly every pairwise plot; the second emphasises the curvilinear weight–mpg and horsepower–quarter-mile relationships that a single scatter would obscure.

6.27.7 Interpretation

A reporting sentence (figure caption): "Pairs plot of four iris measurements (n = 150), coloured by species; diagonal shows kernel densities, lower triangle shows pairwise scatter, upper triangle shows Pearson correlations within each species (`GGally::ggpairs`)." Always state the correlation type used in the upper triangle and the geom chosen for the lower; defaults differ between versions.

6.27.8 Practical Tips

- Keep the matrix to 3–6 variables; cells shrink quadratically and become unreadable beyond that. For more variables, switch to a correlation heatmap or a focused PCA.
- Add `aes(colour=group)` to surface between-group structure; the same colour propagates through diagonal densities and lower-triangle scatters automatically.
- For large n, replace scatter cells with hexbin via `lower=list(continuous="hex")`; a 50-by-50 grid scales to millions of points where individual scatters fail.
- `diag=list(continuous="barDiag")` shows histograms instead of densities; useful for discrete-looking continuous variables (e.g. integer scores).
- Save pairs plots at large dimensions (`ggsave(...,width=10,height=10,dpi=300)`); shrinking them to a single column makes the cells unreadable.
- For posterior or simulation diagnostics, `bayesplot::mcmc_pairs()` is the Bayesian-flavoured analogue and overlays divergent transitions on the off-diagonal cells.

6.27.9 R Packages Used

`GGally` for `ggpairs()`, `ggcorr()`, and the `wrap()` helper that lets you customise individual cell types; `ggplot2` for the underlying grammar; `bayesplot::mcmc_pairs()` for posterior-draw matrices in MCMC diagnostics.

6.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.27.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.28 Patchwork: Multi-Plot Composition

6.28.1 Introduction

Almost every journal-worthy figure is a composition: two or more panels arranged with shared legends, alphabetical tags, a unifying title, and aligned axes. Building such figures by hand with `gridExtra` or `cowplot` is verbose and brittle to layout changes. The `patchwork` package replaces that workflow with arithmetic-like operators that mirror the way the figure is read: `+` for "next to", `/` for "below", `|` for "in a row". Combined with `plot_layout()` for relative sizes and `plot_annotation()` for shared titles and tags, it is now the standard tool for assembling multi-panel `ggplot2` figures.

6.28.2 Prerequisites

A working knowledge of `ggplot2` — geoms, scales, themes — and an idea of which combinations of plots actually communicate (rather than merely fill space).

6.28.3 Theory

`patchwork` treats each `ggplot` as an object that supports a small algebra:

- `p1+p2` places two plots side by side at equal width.
- `p1/p2` stacks them vertically.
- Parentheses control nesting: `(p1|p2)/p3` puts `p1` and `p2` horizontally with `p3` spanning below.
- `plot_layout(guides="collect")` deduplicates shared legends, moving a single copy to the figure margin.
- `plot_annotation(title=...,tag_levels="A")` adds a global title and panel tags (A, B, C, …) automatically.
- `&theme_minimal()` applies a theme to every panel in the composition simultaneously, useful when individual panels were built without a consistent theme.

The package handles axis alignment automatically when plots share axis ranges, which avoids the tiny horizontal offsets that bedevil hand-aligned multi-panel figures.

6.28.4 Assumptions

No statistical assumptions; the only practical constraint is that combined panels must remain legible at the final printed size, which usually means three to four panels per row at most.

6.28.5 R Implementation

```r
library(ggplot2); library(patchwork)

p1 <- ggplot(mtcars, aes(wt, mpg))       + geom_point() + theme_minimal()
p2 <- ggplot(mtcars, aes(factor(cyl)))   + geom_bar()   + theme_minimal()
p3 <- ggplot(mtcars, aes(hp))             + geom_histogram(bins = 15) +
 ↪  theme_minimal()

# Two side by side
p1 + p2

# Stacked
p1 / p2

# Complex layout
(p1 | p2) / p3 +
  plot_annotation(title = "mtcars overview", tag_levels = "A")

# Collect shared legends
p1 + p2 + plot_layout(guides = "collect")
```

6.28.6 Output & Results

Four progressively more complex layouts: two horizontal panels, two vertical panels, a 2-up-1-down arrangement with global title and panel tags A, B, C, and a side-by-side composition with one shared legend collected into the right margin. Saving any of these with `ggsave()` produces a single file at the specified dimensions, ready for journal submission.

6.28.7 Interpretation

A reporting sentence (figure caption): "Figure 2 (assembled with `patchwork`): A) weight vs miles-per-gallon, B) cylinder distribution, C) horsepower distribution; panels share a common minimal theme and a single legend." Caption-level credit to the composition tool is unnecessary, but tags and a one-line description of each panel are essential for any multi-panel figure.

6.28.8 Practical Tips

- Use `plot_layout(widths=c(2,1))` to give one panel twice the horizontal space of another; `heights` does the same vertically.
- `tag_levels="A"` produces A, B, C tags; `"1"` produces 1, 2, 3; `"i"` produces lowercase Roman numerals.
- `&theme_minimal()` (note the `&`) applies a theme globally to every panel; `+` only applies to the most recently added plot.

- For many small panels with the same structure, prefer `facet_wrap()` inside a single `ggplot` rather than composing dozens of plots; faceting is more efficient and produces aligned axes by default.
- Always save the composed plot as a single file via `ggsave()`; assembling separate exports in a vector editor produces fragile artwork.
- When a shared legend has many entries, place it on the right via `plot_layout(guides="collect")&theme(legend.position="right")` rather than letting it stack vertically inside one panel.

6.28.9 R Packages Used

`patchwork` for composition, `ggplot2` for the underlying plots, and `cowplot` for the alternative `plot_grid()` API and the `draw_image()` helper when embedding bitmap insets is required.

6.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.28.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.29 Raincloud Plots

6.29.1 Introduction

A raincloud plot is the modern alternative to the bar-with-error-bar chart for comparing continuous distributions across groups. Proposed by Allen and colleagues in 2019, it combines three pieces of information into one panel: a half-violin showing the kernel density (the "cloud"), jittered raw observations (the "rain"), and a small boxplot encoding the median and quartiles. The result preserves all the distributional information that bars throw away — modes, skew, outliers, sample sizes — while still presenting the central tendency that drives most reporting decisions.

6.29.2 Prerequisites

A working knowledge of boxplots, violin plots, kernel density estimation, and `ggplot2` aesthetic mappings.

6.29.3 Theory

A raincloud plot stacks three layers per group:

1. **Half-violin** drawn to one side (`stat_halfeye()` from `ggdist` or `geom_violinh()` from `gghalves`); displays the density shape.
2. **Jittered raw points** (`geom_point()` with `position_jitter()` or `ggbeeswarm::geom_beeswarm()`); shows individual observations.
3. **Narrow boxplot** with hidden outliers (`geom_boxplot(width=0.12,outlier.shape=NA)`); communicates the median, IQR, and whisker range.

`coord_flip()` rotates the layout to horizontal, which makes long category names easier to read. The half-violin opens away from the data column so the cloud, the rain, and the boxplot sit side by side rather than stacked.

6.29.4 Assumptions

A continuous outcome and a categorical (or ordered) grouping variable. Each group should have enough observations (at least 20–30) for the kernel density to be meaningful; below that, drop the half-violin and show the points alone.

6.29.5 R Implementation

```
library(ggplot2); library(ggdist)

ggplot(iris, aes(Species, Sepal.Length, fill = Species)) +
  stat_halfeye(adjust = 0.5, justification = -0.2, .width = 0,
   ↪  point_colour = NA) +
  geom_boxplot(width = 0.12, outlier.shape = NA, alpha = 0.5) +
  geom_point(position = position_jitter(width = 0.05, seed = 1),
             size = 1.3, alpha = 0.5) +
  scale_fill_brewer(palette = "Set2") +
  coord_flip() +
  theme_minimal()
```

6.29.6 Output & Results

A horizontal raincloud plot for the iris dataset: each species occupies one row with its density rendered as a half-violin to the right, a small boxplot in the middle, and jittered raw points to the left. The full distribution, summary statistics, and individual observations are visible simultaneously.

6.29.7 Interpretation

A reporting sentence (figure caption): "Raincloud plots of sepal length by species (n = 50 per group): the half-violin shows kernel-density estimates (`adjust=0.5`), the jittered points show individual observations, and the boxplot summarises

the median and IQR. Setosa is well-separated from the other two species, which overlap considerably." Mention the bandwidth (or `adjust`) used; it shapes the visible structure of the cloud.

6.29.8 Practical Tips

- Use `coord_flip()` for long group names; vertical raincloud plots crowd labels and waste space.
- Drop the boxplot layer when group sizes are small (< 30); it adds visual clutter without summarising much.
- For very large per-group n (thousands), replace jittered points with a single hexbin column or an alpha-blended strip; thousands of dots stack into noise.
- Set `outlier.shape=NA` in the boxplot so individual outliers do not appear twice (once in the rain, once flagged by the box whiskers).
- Use `ggbeeswarm::geom_beeswarm()` instead of `position_jitter()` when collisions matter; it produces a tidy spread along the category axis.
- `ggdist` is the modern implementation and integrates with Bayesian posterior summaries; `gghalves` is an older alternative still in active use.

6.29.9 R Packages Used

`ggdist` for `stat_halfeye()`, `stat_dots()`, and other slabinterval geoms; `ggplot2` for the underlying grammar; `gghalves` as an older alternative for half-violins; `ggbeeswarm` for collision-free jittered rain layers.

6.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.29.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.30 Ridge Plots

6.30.1 Introduction

A ridge plot stacks kernel-density curves for multiple groups vertically with a small overlap, evoking the cover of a 1979 Joy Division album (hence the alternative name "joy plot"). The idiom is particularly effective for comparing

the shape of distributions across many ordered groups — months of the year, dose levels, time periods, ranked categories — where a panel of separate density plots would be too sparse and a violin plot would lose detail in the overlap.

6.30.2 Prerequisites

A working knowledge of kernel density estimation, grouped distributional comparisons, and `ggplot2`'s aesthetic mappings.

6.30.3 Theory

Each group's density curve is computed independently and then offset vertically by the group's position on the y-axis. The `scale` argument controls how much the curves overlap: a `scale` of 1 means the tallest curve in each group exactly reaches the next baseline, while values above 1 produce overlap and values below 1 produce gaps. The fill aesthetic can encode the group itself (categorical fill) or be mapped to the x-value via `after_stat(x)` to produce gradient fills that highlight quantiles or thresholds. The horizontal axis carries the variable of interest; the vertical axis carries group identity, ideally ordered.

6.30.4 Assumptions

A continuous outcome variable, a categorical (or ordered) grouping variable, and at least a few dozen observations per group so that the kernel density estimate is meaningful. With fewer than 20–30 observations per group, ridge plots can produce noisy, misleading curves.

6.30.5 R Implementation

```
library(ggplot2); library(ggridges)

# Classic ridge plot
ggplot(iris, aes(Sepal.Length, Species, fill = Species)) +
  geom_density_ridges(alpha = 0.6) +
  scale_fill_brewer(palette = "Set2") +
  theme_ridges()

# With gradient fill encoding quantiles
ggplot(diamonds, aes(price, cut, fill = after_stat(x))) +
  stat_density_ridges(geom = "density_ridges_gradient", scale = 2) +
  scale_fill_viridis_c() +
  scale_x_continuous(trans = "log10") +
  theme_ridges()
```

6.30.6 Output & Results

Two ridge plots: the iris data showing three species along the sepal-length axis with translucent fills, and the diamonds data showing the price distribution per cut on a log scale with a gradient fill that encodes price itself. The second example is a particularly good use of ridges — five ordered groups, log-transformed x, and a gradient fill that emphasises the right tail.

6.30.7 Interpretation

A reporting sentence (figure caption): "Density of log-price per cut for 53,940 diamonds, computed via kernel density estimation with default bandwidth; fill encodes log price to highlight quantile structure across cuts." Mention bandwidth and any transformations because both shape the visible structure.

6.30.8 Practical Tips

- Choose `scale` deliberately: 1.5–2 is the typical sweet spot for 6–10 groups; tighter ranges work for fewer groups, wider ranges for many.
- Order groups meaningfully (`forcats::fct_reorder()`, `fct_relevel()`); alphabetical ordering rarely communicates the structure you want to show.
- For highly skewed distributions, log-transform the x-axis (`scale_x_continuous(trans="log10")`); the ridges then capture multiplicative differences across groups.
- `theme_ridges()` removes axis lines and adjusts margins to suit stacked densities; pair with a sequential or qualitative palette.
- Ridge plots struggle with very few groups (two or three) where violin plots or paired density plots communicate better; reach for ridges when six or more groups need to be compared simultaneously.
- For groups with very different sample sizes, consider showing density on a comparable scale (`scale=0.95`) so a small group does not dominate the figure visually.

6.30.9 R Packages Used

`ggridges` for `geom_density_ridges()`, `stat_density_ridges()`, and `theme_ridges()`; `ggplot2` for the underlying grammar; `viridis` and `RColorBrewer` for sequential and qualitative fill scales appropriate to ordered group comparisons.

6.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.30.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.31 ROC Curves

6.31.1 Introduction

A receiver operating characteristic (ROC) curve traces a binary classifier's sensitivity against 1− specificity as the decision threshold sweeps from one end of the score range to the other. It is the canonical diagnostic-performance figure in medicine and machine learning, summarising the trade-off between true-positive and false-positive rates without committing to a single threshold. The area under the curve (AUC) condenses the curve into one number, equal to the probability that the classifier ranks a randomly chosen positive case higher than a randomly chosen negative one.

6.31.2 Prerequisites

A working understanding of sensitivity, specificity, decision thresholds, and the difference between calibration and discrimination in classifier evaluation.

6.31.3 Theory

For a continuous score s and a binary outcome y, the ROC curve plots

$$(\mathrm{FPR}(t), \mathrm{TPR}(t)) = (P(s > t \mid y = 0), P(s > t \mid y = 1))$$

for thresholds t. A perfect classifier passes through $(0, 1)$; a random one follows the diagonal. The AUC equals the Mann-Whitney U statistic divided by the sample sizes; it lies in $[0, 1]$ with 0.5 representing chance and 1 a perfect rank-ordering. Conventional benchmarks: AUC > 0.7 acceptable, > 0.8 good, > 0.9 excellent. DeLong's test compares two AUCs with correlated samples.

6.31.4 Assumptions

A binary outcome and a continuous (or ordinal) score. The ROC curve is invariant to monotone transformations of the score, so calibration is irrelevant to the curve itself; thresholding decisions, however, do depend on calibration.

6.31.5 R Implementation

```
library(pROC); library(plotROC); library(ggplot2)

set.seed(2026)
```

```r
n <- 200
disease <- rbinom(n, 1, 0.5)
score   <- rnorm(n, mean = disease, sd = 1)

# pROC
roc_obj <- roc(disease, score, quiet = TRUE)
auc(roc_obj); ci.auc(roc_obj)
plot(roc_obj, legacy.axes = TRUE, col = "#2A9D8F", lwd = 2)

# ggplot via plotROC
df <- data.frame(D = disease, score = score)
ggplot(df, aes(d = D, m = score)) +
  geom_roc(labelsize = 3, labelround = 2, colour = "#2A9D8F", size = 1) +
  style_roc() +
  labs(title = paste0("AUC = ", round(auc(roc_obj), 3)))
```

6.31.6 Output & Results

A monotone curve rising from $(0, 0)$ toward $(1, 1)$, with optional threshold labels at selected points and the AUC reported either in the title or via `ci.auc()` for its 95 % confidence interval. The `plotROC` package adds `ggplot2`-native styling; `pROC::plot()` produces a fast base-R version.

6.31.7 Interpretation

A reporting sentence: "The classifier achieved an AUC of 0.85 (95 % CI 0.79 to 0.91), indicating good discrimination; at the Youden-optimal threshold of 0.42, sensitivity was 0.78 and specificity was 0.81." Always pair the AUC with its confidence interval and at least one operating point so readers can judge clinical utility.

6.31.8 Practical Tips

- Report the AUC with its 95 % CI, not just a point estimate; the bootstrap interval (`ci.auc(roc_obj)`) handles the typical correlated-sample structure.
- Compare two ROC curves with DeLong's test (`pROC::roc.test`); pairwise AUC differences with p-values are the standard way to claim one classifier outperforms another.
- Avoid smoothed ROC curves (`smooth=TRUE`) for primary reporting; smoothing hides finite-sample irregularities. Show smoothed and raw side by side when smoothing aids interpretation.
- Under severe class imbalance, precision-recall curves often communicate operating-point trade-offs better than ROC; report both when positives are rare.

- Always draw the diagonal reference line ($y = x$); without it, readers cannot judge how far above chance the classifier is.
- For threshold selection, mark the Youden index, the equal-error-rate point, or a clinically motivated sensitivity / specificity target on the curve and quote its threshold.

6.31.9 R Packages Used

`pROC` for ROC objects, AUC, confidence intervals, and DeLong's test; `plotROC` for `ggplot2`-native ROC layers (`geom_roc()`, `style_roc()`); `ROCit` for an alternative API with bootstrap CIs; `yardstick` for tidy classifier metrics within the `tidymodels` ecosystem.

6.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.31.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.32 Scales and Coordinates

6.32.1 Introduction

Scales and coordinates are the two layers of `ggplot2` that turn raw data into visual marks on a page. **Scales** map data values to visual properties — numbers to positions, factor levels to colours, ordered values to sizes — while **coordinates** describe the geometry of the plot area itself — Cartesian, polar, log, fixed aspect ratio. Most figure customisation happens at the scale level; coordinates change less often, but understanding the difference between them is what lets you switch between zooming, transforming, and projecting without breaking the rest of the plot.

6.32.2 Prerequisites

A working knowledge of `ggplot2` aesthetics and geoms, and basic familiarity with how `aes()` mappings flow through to scale choices.

6.32.3 Theory

Every aesthetic has a family of scales:

- `scale_x_continuous`, `scale_x_discrete`, `scale_x_log10`, `scale_x_date`, `scale_x_datetime` for x; `scale_y_*` for y.
- `scale_colour_*` and `scale_fill_*` for colour and fill, with continuous, discrete, viridis, brewer, gradient, and manual variants.
- `scale_size_*`, `scale_alpha_*`, `scale_shape_*`, `scale_linetype_*` for the remaining aesthetics.

Each scale exposes `breaks`, `labels`, `limits`, `trans`, and (for continuous) `expand` arguments that control tick placement, label formatting, axis range, monotone transformations, and padding.

Coordinates change the geometry of the panel:

- `coord_cartesian()` is the default; supplying `xlim` / `ylim` zooms without dropping data.
- `coord_flip()` swaps x and y axes after scales are applied.
- `coord_polar()` produces pie charts and radial layouts.
- `coord_fixed(ratio=1)` enforces equal aspect ratio, essential for maps and structural diagrams.
- `coord_sf()` and `coord_map()` apply geographic projections.

The crucial distinction: `scale_x_continuous(limits=c(a,b))` *filters* observations outside `[a,b]` and recomputes statistics on the remaining data; `coord_cartesian(xlim=c(a,b))` *zooms* without filtering, leaving statistics untouched.

6.32.4 Assumptions

The data supplied are valid for the chosen scale (no negative values for log scales, no fractional dates for date scales).

6.32.5 R Implementation

```
library(ggplot2); library(scales)

# Log-scale y with sensible breaks
ggplot(diamonds, aes(carat, price)) +
  geom_point(alpha = 0.1) +
  scale_y_log10(labels = scales::dollar_format()) +
  theme_minimal()

# Manual colour and custom axis formatting
ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) +
  geom_point(size = 3) +
  scale_colour_manual(values = c("4" = "#2A9D8F",
                                 "6" = "#F4A261",
                                 "8" = "#6A4C93")) +
  scale_x_continuous(breaks = 1:5, labels = paste0(1:5, "K lbs")) +
```

```
  theme_minimal()

# Coord flip: horizontal bar chart
ggplot(mpg, aes(class)) +
  geom_bar(fill = "#2A9D8F") +
  coord_flip() +
  theme_minimal()

# Zoom without filtering
ggplot(diamonds, aes(carat, price)) +
  geom_point(alpha = 0.1) +
  coord_cartesian(xlim = c(0, 2)) +
  theme_minimal()
```

6.32.6 Output & Results

Four illustrative plots: a log-scale price axis with currency formatting, a manual colour scale with custom x-axis labels, a horizontal bar chart via `coord_flip()`, and a zoomed-in scatter that preserves the full underlying dataset for accurate statistical layers.

6.32.7 Interpretation

A reporting sentence: "Price is shown on a base-10 log scale (`scale_y_log10`); the linear relationship between log-price and carat reflects an approximately power-law pricing structure." Note any non-default scale or coordinate in the figure caption — a log-scale axis is easy to overlook and can mislead readers expecting linear interpretation.

6.32.8 Practical Tips

- Reach for `scale_y_log10()` whenever the y-axis spans orders of magnitude; constant-rate growth then reads as a straight line.
- Use `coord_cartesian()` to zoom — `scale_x_continuous(limits=...)` *removes* the filtered data and recomputes any smoother or summary, which can mislead.
- Always label axis units; "Weight" alone is ambiguous, "Weight (1000 lbs)" is informative.
- For date axes, use `scale_x_date(date_breaks="1year",date_labels="%Y")` and pick break frequencies that match the publication width.
- Reverse a scale with `scale_x_reverse()` (continuous) or `scale_x_discrete(limits=rev)` (categorical); avoid hand-flipping data frames.
- For diverging numeric scales centred at zero, combine `scale_fill_gradient2()` with explicit `low`, `mid`, `high` colours; avoid one-sided palettes that misrepresent symmetry.

6.32.9 R Packages Used

`ggplot2` for the scale and coordinate functions; `scales` for label formatters (`dollar_format`, `percent_format`, `comma_format`, `label_log`); `ggh4x` for nested axis labels and per-panel scale control; `cowplot` for `theme_cowplot()` paired with these scales for a publication look.

6.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.32.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.33 Scatter Plots

6.33.1 Introduction

A scatter plot is the default visualisation for the relationship between two continuous variables: each observation becomes a point, position encodes the value pair, and the eye reads correlation, clustering, and outliers directly. It is the first graphic to make whenever a regression or correlation is contemplated. The simplicity is also the trap: at modest sample sizes a scatter is informative, but at large n overplotting hides density, and at small n a few points are easy to misread as a strong relationship that disappears under any reasonable significance test.

6.33.2 Prerequisites

A working knowledge of `ggplot2`'s aesthetic mappings and an idea of how the bivariate distribution differs from marginal univariate summaries.

6.33.3 Theory

The core geom is `geom_point()`, with point size, shape, alpha, and colour as the controllable aesthetics. Overplotting — multiple observations stacking on the same pixel — is the universal challenge once n exceeds a few hundred. Standard remedies:

- **Alpha blending**: lower the per-point opacity so density translates into colour intensity.

- **Jittering**: `position_jitter()` adds a small random offset; useful when one or both axes are discrete.
- **Hexagonal binning**: `geom_hex()` divides the plane into hexagons coloured by count; appropriate for $n \geq 5{,}000$.
- **Density colouring**: `ggpointdensity::geom_pointdensity()` keeps individual points but colours each by local density.
- **2D density contours**: `geom_density_2d()` and `stat_density_2d_filled()` overlay smooth level sets on top of a sparse scatter.

Trend layers come from `geom_smooth()`, with LOESS as the default for exploratory work and `method="lm"` (or `"gam"`, `"glm"`) for parametric fits.

6.33.4 Assumptions

Both x and y are continuous; one categorical variable should be encoded with colour or shape rather than placed on an axis. No statistical assumptions until a smoother or fitted line is added.

6.33.5 R Implementation

```
library(ggplot2); library(ggpointdensity)

# Simple scatter
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(size = 3, colour = "#2A9D8F") +
  theme_minimal()

# Overplotted data with alpha
ggplot(diamonds, aes(carat, price)) +
  geom_point(alpha = 0.05) +
  theme_minimal()

# Point density colouring
ggplot(diamonds[sample(nrow(diamonds), 2000), ], aes(carat, price)) +
  geom_pointdensity() +
  scale_colour_viridis_c() +
  theme_minimal()

# Scatter with trend line
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(size = 3) +
  geom_smooth(method = "lm", colour = "#F4A261") +
  theme_minimal()
```

6.33.6 Output & Results

Four scatter plots showing the same idea at different scales: a small clean scatter for the 32-row `mtcars`, a density-revealing alpha-blended scatter for the 53,940-row `diamonds`, a density-coloured point-by-point view for an intermediate sample, and a small scatter with an OLS smoother and confidence band overlay.

6.33.7 Interpretation

A reporting sentence: "Higher carat values are associated with higher price (Spearman $\rho = 0.93$, $n = 53{,}940$); the relationship is approximately linear on a log-log scale and the alpha-blended scatter reveals heteroscedasticity in the right tail." Always inspect a scatter before computing a correlation: non-linearity, subgroup structure, and outliers are visible here but hidden in a single summary number.

6.33.8 Practical Tips

- For $n > 1{,}000$ use alpha (0.1–0.3 typical); for $n > 10{,}000$ switch to hexbin or density colouring outright.
- Always label axes with units; "Weight" is ambiguous, "Weight (1000 lbs)" is informative.
- Add a smoother only when a relationship is plausible; over-eager LOESS curves through random scatter create the illusion of structure.
- For paired samples, connect points with `geom_line(aes(group=subject))` so within-subject change is visible alongside the cross-sectional spread.
- Use `stat_ellipse()` to draw 95 % data ellipses for group boundaries when colour-coded clusters need an extra visual hint.
- Avoid 3D scatter plots in print; rotation kills depth perception and the third dimension is better encoded as colour, size, or a faceting variable.

6.33.9 R Packages Used

`ggplot2` for `geom_point()`, `geom_smooth()`, `stat_ellipse()`; `ggpointdensity` for density-coloured scatter; `hexbin` for `geom_hex()` with large n; `ggrepel` for labelling individual points without overlap.

6.33.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.33.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.34 Stacked and Dodged Bars

6.34.1 Introduction

When a bar chart needs to display two categorical variables at once, three layout options compete: stacking the sub-bars within each main bar (showing totals and composition), placing them side by side (showing within-category comparison), or normalising to 100 % (showing proportions independent of total). Each layout answers a different question, and choosing the right one is more important than tweaking colours or fonts. The `position` argument of `geom_bar()` and `geom_col()` selects between the three.

6.34.2 Prerequisites

A working knowledge of bar charts, the distinction between `geom_bar()` (counts) and `geom_col()` (pre-computed values), and `ggplot2` aesthetic mappings.

6.34.3 Theory

`position="stack"` is the default for bar geoms with a fill aesthetic: sub-bars stack vertically inside each main bar, the bar's total height equals the sum of the sub-counts, and segments encode composition.

`position="dodge"` (or `position_dodge(width=...)` for finer control) places sub-bars side by side, one per fill level, at the same x-tick. Each sub-bar's height equals its own count, making within-x comparison direct.

`position="fill"` stacks but normalises every main bar to height 1 (or 100 %), hiding the absolute totals to expose proportional structure across the x-axis.

The choice maps directly onto the analytical question: "How big is each group?" → stack; "How does subgroup A compare to subgroup B within each main category?" → dodge; "How does the proportion of subgroup A shift across main categories?" → fill.

6.34.4 Assumptions

Two categorical variables (or one categorical and one ordered) and a count or summary that makes sense to add across sub-bars.

6.34.5 R Implementation

```r
library(ggplot2)

d <- data.frame(
  stage = factor(rep(c("I", "II", "III"), 2)),
  sex   = factor(rep(c("F", "M"), each = 3)),
  count = c(30, 45, 25, 25, 50, 30)
)

# Stacked
ggplot(d, aes(stage, count, fill = sex)) +
  geom_col(position = "stack") +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()

# Dodged
ggplot(d, aes(stage, count, fill = sex)) +
  geom_col(position = position_dodge(width = 0.8)) +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()

# Filled (proportions)
ggplot(d, aes(stage, count, fill = sex)) +
  geom_col(position = "fill") +
  scale_y_continuous(labels = scales::percent_format()) +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()
```

6.34.6 Output & Results

Three views of the same disease-stage-by-sex data: a stacked bar showing the total cases per stage decomposed by sex, a dodged bar enabling direct sex-vs-sex comparison within stage, and a filled bar showing the female-to-male proportion balance across stages with the y-axis as a percentage.

6.34.7 Interpretation

A reporting sentence (figure caption): "Stacked bars show total case counts per stage decomposed by sex; dodged bars display male and female counts side by side for direct comparison within each stage; filled bars normalise to 100 % to expose the sex proportion across stages." Always describe which layout was chosen and why; it is the most common source of bar-chart misreading.

6.34.8 Practical Tips

- Stacked bars communicate totals cleanly but obscure within-stage comparisons because the segments above the first sub-bar do not start at zero

— readers cannot judge their lengths accurately.
- Dodged bars need adequate bar width and stage spacing; with three or more fill levels, the figure widens quickly and benefits from `coord_flip()` for horizontal layouts.
- Filled bars hide absolute counts entirely; pair them with a text or table summary giving the per-stage n.
- For more than four fill levels, stacking and dodging both become hard to read; consider a heatmap, a faceted small-multiple, or a Sankey diagram instead.
- Order fill levels deliberately (`forcats::fct_relevel()`); the legend reads top-to-bottom and the bar reads bottom-to-top, so the visual flow depends on factor order.
- Use `geom_text(aes(label=count),position=position_stack(vjust=0.5))` to label segments with their counts when the total alone is not enough.

6.34.9 R Packages Used

`ggplot2` for `geom_bar()`, `geom_col()`, and the `position_*` family; `forcats` for factor reordering of the fill levels; `scales` for percent and comma label formatters; `ggalluvial` for flow-style alternatives when stacked bars at three or more levels become confusing.

6.34.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.34.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.35 Survival Curves

6.35.1 Introduction

A Kaplan-Meier survival curve is the most recognisable figure in clinical biostatistics. It estimates the probability of surviving (or remaining event-free) past each time point and displays the result as a step function dropping at every observed event. A publication-grade KM plot is rarely just the curve, however — it carries a risk table beneath the panel showing how many subjects remain at each landmark time, a log-rank p-value annotating the between-group comparison, and 95 % confidence bands shaded behind the curve. The `survm`

iner package combines all of these into a single ggsurvplot() call that works directly on survfit objects from the survival package.

6.35.2 Prerequisites

A working understanding of survival analysis, the Kaplan-Meier estimator, censoring, and the log-rank test. Familiarity with ggplot2 helps when customising the resulting object.

6.35.3 Theory

The KM estimator is

$$\hat{S}(t) = \prod_{t_i \le t} \left(1 - \frac{d_i}{n_i}\right),$$

where d_i is the number of events and n_i the number at risk just before time t_i. The plot displays $\hat{S}(t)$ on the y-axis, time on the x-axis, with steps at each event and tick-marks (or shorter steps) at censoring times. Multiple groups are overlaid in different colours, and the risk-table below the panel shows n_i at chosen landmark times for each group. The log-rank p-value summarises the between-group difference under proportional hazards.

6.35.4 Assumptions

Right-censored survival data, non-informative censoring (the censoring mechanism does not depend on prognosis), and — for the log-rank annotation — approximately proportional hazards across groups. Severely non-proportional hazards make the log-rank test underpowered and a restricted-mean comparison may be more appropriate.

6.35.5 R Implementation

```
library(survival); library(survminer)

fit <- survfit(Surv(time, status) ~ sex, data = lung)

ggsurvplot(fit,
  data = lung,
  conf.int = TRUE,
  pval = TRUE,
  risk.table = TRUE,
  legend.labs = c("Male", "Female"),
  palette = c("#2A9D8F", "#F4A261"),
  xlab = "Days",
```

```
  ylab = "Survival probability",
  risk.table.y.text.col = TRUE
)

# Cumulative event curve (1 - S)
ggsurvplot(fit, data = lung, fun = "event")
```

6.35.6 Output & Results

A two-panel figure: the upper panel shows two survival curves (male and female) with shaded 95 % confidence bands and a log-rank p-value in the corner; the lower panel is a risk table tabulating the number at risk in each group at six evenly spaced time points. The cumulative-event variant (`fun="event"`) inverts the y-axis to show $1 - \hat{S}(t)$, useful when the event rate is the more natural reporting scale.

6.35.7 Interpretation

A reporting sentence (figure caption): "Kaplan–Meier overall-survival curves by sex (`survival::survfit`); shaded bands are 95 % point-wise confidence intervals, the log-rank p-value compares the two groups, and the risk table reports the number of subjects at risk at each labelled time." Always state the time-zero anchor (date of randomisation, surgery, diagnosis) in the methods so the x-axis is unambiguous.

6.35.8 Practical Tips

- Always include a risk table; without it, the curve at late time points is misleading because the estimate is based on a tiny remaining cohort.
- Use `fun="event"` for cumulative incidence framing when communicating to clinical audiences who think in event rates rather than survival probabilities.
- For competing risks, switch to a cumulative incidence function (CIF) plot via `cmprsk::cuminc()` and `ggcompetingrisks()`; KM curves overestimate the event of interest when competing events are non-trivial.
- Pair the log-rank p-value with a hazard-ratio estimate (e.g. from a univariate Cox model); the test alone gives no effect size.
- For multi-arm trials, lay out subgroups in panels (`facet.by="subgroup"`) rather than crowding many curves into one panel.
- Export at 300 dpi (or higher for line art) and at the journal's column width so the risk-table text remains legible.

6.35.9 R Packages Used

`survival` for `survfit()` and `Surv()`; `survminer` for `ggsurvplot()`, `ggsurvtable()`, and risk-table support; `ggplot2` for downstream customisation of the returned object; `cmprsk` for competing-risks extensions when KM is not appropriate.

6.35.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.35.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.36 Time Series Plots

6.36.1 Introduction

Time-series plots are deceptively easy to make and surprisingly easy to ruin. The data are ordered, the x-axis is a date, and a single line is often the right geom — but the choice of tick spacing, transformation, and accompanying decomposition or forecast layer determines whether the plot communicates the underlying dynamics or hides them. Trend, seasonality, and forecast uncertainty are three distinct stories the same series can tell, and a good time-series figure usually shows at least two of them.

6.36.2 Prerequisites

A working knowledge of `ggplot2`, an understanding of R's date and date-time classes (`Date`, `POSIXct`), and basic familiarity with classical time-series concepts (trend, seasonality, forecast intervals).

6.36.3 Theory

Three structural elements appear in nearly every published time-series figure:

- A continuous date axis with carefully chosen `date_breaks` and `date_labels`. Defaults often produce overcrowded or sparse ticks; choose explicitly.
- A line geom for the observed series, optionally with a smooth or fitted-value overlay.
- An uncertainty layer — typically a `geom_ribbon()` for forecast prediction intervals or a faint shaded region for seasonal subsamples.

For multivariate decompositions, classical STL splits a series into trend, seasonal, and remainder components shown as a stacked panel. The `forecast::autoplot()` and `feasts::autoplot()` methods produce decomposition and forecast plots automatically; both are tidy-friendly and fit naturally into modern reporting workflows.

6.36.4 Assumptions

Time variable is monotonically ordered, on a regular grid (or explicitly handled if irregular), and on a class that `scale_x_date()` or `scale_x_datetime()` understands.

6.36.5 R Implementation

```r
library(ggplot2); library(forecast); library(ggfortify)

# Standard time series plot
ggplot(economics, aes(date, unemploy / 1000)) +
  geom_line(colour = "#2A9D8F", linewidth = 1) +
  scale_x_date(date_breaks = "10 years", date_labels = "%Y") +
  labs(x = NULL, y = "Unemployment (thousands)") +
  theme_minimal()

# Forecast with prediction intervals
fit <- auto.arima(AirPassengers)
fc  <- forecast(fit, h = 24)
autoplot(fc) +
  theme_minimal()

# STL decomposition
decomp <- stl(AirPassengers, s.window = "periodic")
autoplot(decomp) + theme_minimal()
```

6.36.6 Output & Results

Three figures: a long historical line plot of US unemployment with decade ticks, an ARIMA forecast with 80 % and 95 % prediction ribbons attached to the end of the historical series, and a four-panel STL decomposition of monthly air-passenger counts showing the rising trend, twelve-month seasonal cycle, and remaining residual variation.

6.36.7 Interpretation

A reporting sentence (figure caption): "Top: unemployment series 1967–2015 (thousands of persons). Middle: ARIMA(2,1,1)(0,1,0)

12

forecast for 24 months ahead with 80 % and 95 % prediction intervals. Bottom: STL decomposition (`s.window="periodic"`) of monthly air-passenger counts." Always show forecast intervals; a forecast line without uncertainty is misleading.

6.36.8 Practical Tips

- Convert text dates to `Date` or `POSIXct` before plotting; `lubridate::ymd()`, `dmy()`, and `as_datetime()` cover the common cases.
- Choose `date_breaks` to match the publication width: ten-year ticks for a multi-decade series, one-year ticks for a five-year window, monthly ticks for a single year.
- Log-transform the y-axis when the series exhibits multiplicative growth; the resulting plot reads percentage changes as constant slopes.
- For multiple series, prefer faceting (`facet_wrap(vars(metric),scales="free_y")`) when scales differ; reach for colour only when all series share comparable scale.
- Use the modern `tsibble` + `feasts` + `fable` stack for tidy workflows; `forecast::autoplot()` remains useful for legacy `ts` objects and quick exploration.
- Always annotate forecast plots with the model class and forecast horizon in the caption; a ribbon without context can be mistaken for a confidence band on observed data.

6.36.9 R Packages Used

`ggplot2` for the underlying grammar, `forecast` for `auto.arima()` and `forecast()` plus `autoplot()` methods, `ggfortify` for `ts` and STL `autoplot()` extensions, and the modern `tsibble` / `feasts` / `fable` stack for tidy time-series workflows.

6.36.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.36.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.37 Violin Plots

6.37.1 Introduction

A violin plot displays a kernel-density estimate vertically and reflects it around the central axis, producing the symmetric outline that gives the geom its name. It combines the clarity of a density plot with the side-by-side grouping that boxplots provide, and — crucially — reveals multi-modality and skew that boxplots flatten into a single rectangle. For comparing the *shape* of distributions across categories, violin plots are often the right default; for comparing only the median and quartile structure, boxplots remain more compact.

6.37.2 Prerequisites

A working understanding of kernel density estimation, boxplots, and `ggplot2` aesthetic mappings.

6.37.3 Theory

For each group, `geom_violin()` computes a kernel density $\hat{f}(x)$ on the response variable, draws the density curve vertically, and mirrors it across the group's central axis. The resulting width at each y-value is proportional to $\hat{f}(y)$, so wide regions correspond to high density. The `scale` argument controls how groups are sized relative to each other: `"area"` makes all violins equal area, `"count"` scales width proportional to the group's sample size, and `"width"` makes them all the same maximum width. The `trim` argument decides whether the density tails are truncated at the data extremes (`TRUE`, default) or extrapolated beyond them (`FALSE`).

6.37.4 Assumptions

A continuous outcome and a categorical (or ordered) grouping variable, with enough observations per group (at least 20–30) for the kernel density to be meaningful. Below that threshold the shape becomes unstable and a strip plot or jittered scatter is more honest.

6.37.5 R Implementation

```
library(ggplot2)

# Basic violin
ggplot(iris, aes(Species, Sepal.Length, fill = Species)) +
  geom_violin() +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()
```

```r
# Violin + boxplot overlay
ggplot(iris, aes(Species, Sepal.Length, fill = Species)) +
  geom_violin(alpha = 0.5, trim = FALSE) +
  geom_boxplot(width = 0.2, fill = "white", outlier.shape = NA) +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()

# Violin + jittered points
ggplot(iris, aes(Species, Sepal.Length, fill = Species)) +
  geom_violin(alpha = 0.5) +
  geom_jitter(width = 0.1, alpha = 0.5) +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()
```

6.37.6 Output & Results

Three views of the iris sepal-length data: a plain violin per species, a violin with a narrow boxplot overlay showing median and quartiles, and a violin with jittered raw points overlaid for full transparency. The third version is the most informative when the audience is unfamiliar with kernel-density visualisations.

6.37.7 Interpretation

A reporting sentence (figure caption): "Sepal length by species (n = 50 per group); violin width is proportional to kernel-density estimate (`scale="area"`), the inset boxplot summarises the median and IQR, and jittered points show individual observations." Always describe what the violin's width encodes; readers unfamiliar with the geom default to thinking it represents count.

6.37.8 Practical Tips

- For multi-modal distributions (e.g. Old Faithful eruption times), the violin reveals the second mode that a boxplot would hide entirely; this is the strongest single argument for the geom.
- Set `trim=FALSE` only when the density tails are biologically meaningful; otherwise the default trim avoids implausible extrapolation.
- Combine with a narrow boxplot (`width=0.15`) when readers expect numeric summaries; the inset preserves quartile information without dominating the visual.
- For paired or matched groups, `introdataviz::geom_split_violin()` produces left-right halves so two conditions share the same x-position.
- For large n, prefer violins over jittered points; violins scale to millions of observations whereas individual points stack into uninformative blobs.
- For small n (< 20 per group), a strip plot or a boxplot is more honest than a violin; the kernel density estimate is too noisy to trust.

6.37.9 R Packages Used

`ggplot2` for `geom_violin()`; `introdataviz` for split violins; `ggdist` for `stat_halfeye()` and other slabinterval geoms that combine violins with credible-interval annotations; `vioplot` for a base-R alternative when `ggplot2` is not available.

6.37.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

6.37.11 See also — labs in this chapter

- ggplot2 grammar and multi-panel layouts

6.38 Week 1, Session 5 — ggplot2 grammar and multi-panel layouts

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

6.38.1 Learning objectives

- Build a ggplot by naming data, aesthetics, geoms, and scales rather than by calling a plot type.
- Use faceting to display the same relationship across subgroups.
- Combine several plots into a single figure with patchwork.

6.38.2 Prerequisites

Labs 1.2 through 1.4.

6.38.3 Background

ggplot2 is built on a grammar of graphics. Rather than call a function named "scatterplot" or "boxplot", you declare what the plot contains: a dataset, mappings from variables to aesthetics (x, y, colour, size, shape), and one or more geoms that render those mappings. The result is that *any* plot you can picture can be built by the same small vocabulary. Once the vocabulary is second nature, you will spend less time searching for a plotting function and more time thinking about what the graph should show.

Multi-panel figures are the standard unit of publication. Faceting breaks one plot into small multiples — the same relationship repeated across a grouping variable — which is almost always a better answer than crowding more colours onto a single panel. The patchwork package takes independent ggplots and assembles them into a single figure using `+`, `|`, and `/` operators, so you can build a three-panel manuscript figure without leaving R.

A common beginner mistake is to treat ggplot as a collection of `geom_*` functions. It is cleaner to think of a plot as a data-plus- aesthetics object to which geoms are added. Changing the underlying data is then a substitution, not a rewrite.

6.38.4 Setup

```
library(tidyverse)
library(palmerpenguins)
library(patchwork)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

6.38.5 1. Goal

Build a three-panel figure that describes bill morphology and body mass in the `penguins` dataset, using a scatter plot, a boxplot, and a faceted scatter plot, assembled with patchwork.

6.38.6 2. Approach

```
p <- penguins |>
  drop_na(bill_length_mm, bill_depth_mm, body_mass_g, species, sex)
```

The plot object is built in three lines: data, aesthetics, geom.

```
p1 <- p |>
  ggplot(aes(bill_length_mm, bill_depth_mm, colour = species)) +
  geom_point(alpha = 0.7) +
  labs(x = "Bill length (mm)",
       y = "Bill depth (mm)",
       colour = NULL)
p1
```

6.38.7 3. Execution

```
p2 <- p |>
  ggplot(aes(species, body_mass_g, fill = species)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
```

```
  labs(x = NULL, y = "Body mass (g)") +
  theme(legend.position = "none")
p2
```

```
p3 <- p |>
  ggplot(aes(bill_length_mm, body_mass_g, colour = sex)) +
  geom_point(alpha = 0.6) +
  facet_wrap(~ species) +
  labs(x = "Bill length (mm)",
       y = "Body mass (g)",
       colour = NULL)
p3
```

Faceting shows the same relationship in three panels, one per species. The within-species slopes are visually separable from the between-species differences.

6.38.8 4. Check

Assemble the three panels with patchwork.

```
(p1 | p2) / p3 +
  plot_annotation(
    title = "Palmer penguins: bill morphology and body mass",
    tag_levels = "A"
  )
```

The layout operators: `|` places side-by-side, `/` stacks, `+` groups. `plot_annotation()` adds title and the familiar A/B/C labels used in manuscripts.

6.38.9 5. Report

> A three-panel figure was assembled from the `palmerpenguins` data (n = `nrow(p)` complete cases) using ggplot2 and patchwork. Panel A shows a clear separation of bill morphology among the three species. Panel B shows Gentoo as markedly heavier than the other two. Panel C, faceted by species, reveals a within-species positive association between bill length and body mass, with a sex offset visible in each panel.

Within-group and between-group variation are separate features of the data. Faceting is the simplest way to let a reader see both at once.

6.38.10 Common pitfalls

- Using colour to encode more than one thing at a time (group *and* magnitude — pick one).
- Mapping a continuous variable to shape.

- Overlaying more than four or five categories on a single scatter plot; use faceting instead.
- Defaulting to `theme_grey()`; the in-house convention is `theme_minimal(base_size=12)`.

6.38.11 Further reading

- Wickham H. *ggplot2: Elegant Graphics for Data Analysis*, chapters on the grammar and on faceting.
- Pedersen T. *patchwork* package documentation.

6.38.12 Session info

```
sessionInfo()
```

6.38.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 7

Regression & Modelling

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Linear models, GLMs, GAMs, non-linear least squares, the ANOVA family, ANCOVA, robust regression, residual diagnostics, calibration, and decision-curve analysis. The chapter treats regression as a single object viewed from different angles.

This chapter contains **51 method pages** and **17 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

7.1 Method pages

Method	Source slug
Added-Variable Plots	`added-variable-plots`
Best Subset Selection	`best-subset-selection`
Beta Regression	`beta-regression`
Centring and Scaling Predictors	`centered-scaled-predictors`
Contrasts in R	`contrasts-in-r`
Cook's Distance	`cooks-distance`
Cox Regression as Regression	`survival-regression-cox`
Dirichlet Regression	`dirichlet-regression`
Dummy Coding	`dummy-coding`
Effect (Deviation) Coding	`effect-coding`
Elastic Net	`elastic-net`
GAMMs: Additive Mixed Models	`gamm-additive-mixed`
GAMs: Introduction	`gam-introduction`
Generalised Estimating Equations	`generalized-estimating-equations`
glmer: Generalised Mixed Models	`glmer-generalized-mixed`

Method	Source slug
Hurdle Models	`hurdle-models`
Interactions in Regression	`interactions-in-regression`
Lasso Regression	`lasso-regression`
Leverage and Influence	`leverage-influence`
Linear Regression Assumptions	`regression-assumptions`
Logistic Regression	`logistic-regression`
Logistic Regression Diagnostics	`logistic-regression-diagnostics`
Mixed-Effects Models: Introduction	`mixed-models-intro`
Model Selection with AIC and BIC	`model-selection-aic-bic`
Multicollinearity and VIF	`multicollinearity-vif`
Multinomial Logistic Regression	`multinomial-logistic`
Multiple Linear Regression	`multiple-linear-regression`
Negative Binomial Regression	`negative-binomial-regression`
Nested vs Crossed Random Effects	`nested-crossed-random-effects`
Offsets in Regression	`offsets-in-regression`
Ordinal Logistic Regression	`ordinal-logistic`
Partial Correlation	`partial-correlation`
Poisson Regression	`poisson-regression`
Polynomial Regression	`polynomial-regression`
Probit Regression	`probit-regression`
Quantile Regression	`quantile-regression`
Quasi-Poisson Regression	`quasi-poisson`
R-Squared and Adjusted R-Squared	`r-squared-adjusted`
Random-Intercept Models	`random-intercepts`
Random-Slope Models	`random-slopes`
Residual Diagnostics	`residual-diagnostics`
Ridge Regression	`ridge-regression`
Robust Regression	`robust-regression`
ROC and AUC for Logistic Models	`roc-auc-for-logistic`
Simple Linear Regression	`simple-linear-regression`
Spline Regression	`spline-regression`
Stepwise Regression: Pitfalls	`stepwise-regression-pitfalls`
Tobit Regression	`tobit-regression`
Truncated Regression	`truncated-regression`
Worked lme4::lmer Examples	`lme4-lmer-examples`
Zero-Inflated Models	`zero-inflated-models`

7.2 Labs

Lab
Correlation vs regression; Model-I/II regression
Simple linear regression

Lab
Multiple regression: confounding, interaction, centring
Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
Robust regression, weighted LS, HC (sandwich) standard errors
One-way ANOVA and contrasts (emmeans)
Two-way / factorial ANOVA with interaction
RCBD and repeated measures → mixed models
GAMs with mgcv
Non-linear regression with nls
Logistic regression (binomial GLM)
ANCOVA in RCTs (baseline adjustment)
Ordinal and multinomial regression
Poisson and negative-binomial regression
Calibration, discrimination, ROC/AUC, Brier score
Dichotomisation, change scores, regression to the mean
Decision curves, NRI, IDI

7.3 Added-Variable Plots

7.3.1 Introduction

An added-variable plot (AVP), also called a partial regression plot, displays the relationship between a single predictor and the outcome after adjusting both for all other predictors in the model. It is the geometric expression of a regression coefficient: the slope of the OLS line in the AVP equals the coefficient in the full multiple regression. AVPs are the standard tool for diagnosing whether a partial linear effect is well-supported, whether non-linearity has been missed, and whether one or two observations are driving a particular coefficient.

7.3.2 Prerequisites

A working understanding of multiple regression, partial correlation, and the residualisation interpretation of partial effects.

7.3.3 Theory

For predictor X_j in a model with other predictors X_{-j}:

1. Regress Y on X_{-j} and save residuals $e_{Y|X_{-j}}$.
2. Regress X_j on X_{-j} and save residuals $e_{X_j|X_{-j}}$.
3. Plot $e_{Y|X_{-j}}$ against $e_{X_j|X_{-j}}$.

The OLS slope of this plot equals the coefficient on X_j in the full multiple regression — the famous Frisch-Waugh-Lovell theorem. Patterns in the AVP reveal

partial-effect non-linearity, leverage, and outliers specific to X_j that bivariate scatter plots cannot show.

7.3.4 Assumptions

Standard multiple regression assumptions; the residualisation interpretation requires that the other predictors are themselves correctly specified.

7.3.5 R Implementation

```r
library(car)

fit <- lm(mpg ~ wt + hp + disp + drat, data = mtcars)
avPlots(fit)

# Single AVP
avPlot(fit, variable = "hp")
```

7.3.6 Output & Results

`avPlots()` draws one scatter plot per predictor, each with the fitted partial regression line. The slope of each line equals the corresponding multiple-regression coefficient. Outlier and leverage diagnostics specific to each predictor become visible.

7.3.7 Interpretation

A reporting sentence: "The added-variable plot for horsepower showed a clear negative slope after adjusting for weight, displacement, and rear-axle ratio, with no evidence of non-linearity or individual influential observations; the AVP slope of -0.030 matches the multiple-regression coefficient ($p = 0.005$). Diagnostic plots support reporting the linear partial effect of horsepower." Use AVPs to check both the shape of a partial relationship and to flag observations driving specific coefficients.

7.3.8 Practical Tips

- AVPs are more informative than raw bivariate scatter plots for checking partial effects; bivariate plots ignore the other predictors and can mislead.
- Non-linear patterns in an AVP suggest a transformation, polynomial term, or spline for that specific predictor.
- Outliers in an AVP indicate observations influencing that particular coefficient; cross-check with DFBETAs.
- `car::avPlots(fit)` draws all AVPs in a single grid for compact diagnostic display.

- Distinguish from component-plus-residual plots (`crPlots`), which add the predictor's full effect rather than partialling out other predictors; the two answer related but different diagnostic questions.
- For GLMs and mixed models, AVP analogues exist; see `car::avPlots()` and `effects::predictorEffects()`.

7.3.9 R Packages Used

`car::avPlots()` for the canonical implementation; `car::crPlots()` for component-plus-residual plots; `effects::predictorEffects()` for marginal-effect plots; `ggeffects::ggpredict()` for ggplot-style predicted-effect plots that complement AVP diagnostics.

7.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.3.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.4 Best Subset Selection

7.4.1 Introduction

Best subset selection enumerates all 2^p possible predictor subsets and identifies the best subset by a chosen criterion (adjusted R^2, Mallows' C_p, BIC, cross-validation error). It is an exhaustive search and therefore avoids the sequence-dependence that plagues stepwise methods, but the combinatorial cost limits its practical use to modest p — typically up to about 30 predictors with branch-and-bound algorithms (`leaps`).

7.4.2 Prerequisites

A working understanding of multiple regression, model-selection criteria, and the bias-variance trade-off in choosing model complexity.

7.4.3 Theory

For each subset size $k \in \{0, 1, \ldots, p\}$, find the subset with the lowest residual sum of squares (equivalently, highest R^2). Then compare across sizes using a criterion that penalises complexity:

- **Adjusted** R^2: $1 - (1 - R^2)(n-1)/(n-k-1)$; rewards fit per parameter.
- **Mallows'** C_p: $\mathrm{RSS}_k/\hat{\sigma}^2 - n + 2k$; near k when the subset is adequate.
- **AIC**: $-2\ell + 2k$; favours predictive accuracy.
- **BIC**: $-2\ell + k \log n$; more parsimonious than AIC.
- **Cross-validation**: directly estimates predictive error.

`leaps::regsubsets` uses branch-and-bound to avoid evaluating every subset explicitly, making moderate-p best-subset feasible.

7.4.4 Assumptions

Standard linear-regression assumptions; criteria like AIC, BIC assume Gaussian residuals (other forms exist for GLMs).

7.4.5 R Implementation

```r
library(leaps)

d <- mtcars
fit <- regsubsets(mpg ~ ., data = d, nvmax = 5)
sum_fit <- summary(fit)

data.frame(
  n_pred = 1:5,
  adj_R2 = sum_fit$adjr2,
```

```
  Cp     = sum_fit$cp,
  BIC    = sum_fit$bic
)

# Show the best subset at each size
sum_fit$which

# Plot the criterion
plot(fit, scale = "bic")
```

7.4.6 Output & Results

`regsubsets()` returns the best subset of each size up to `nvmax`. The summary provides criterion values; `sum_fit$which` shows which predictors are included at each size; the plot displays subsets ordered by the chosen criterion.

7.4.7 Interpretation

A reporting sentence: "Best-subset selection on `mtcars` (with `nvmax=5`) identified weight (`wt`), quarter-mile time (`qsec`), and transmission (`am`) as the preferred three-predictor model under BIC (BIC = −84); cross-validated mean squared error confirmed this as the best subset, with substantially worse performance for both smaller and larger models." Always state the criterion and pair with cross-validation when stakes are high.

7.4.8 Practical Tips

- Best subset is computationally feasible up to about $p = 30$ with branch-and-bound; beyond that, switch to penalised methods (lasso, elastic net).
- Lasso and elastic net often perform similarly to best-subset selection with better scalability and less over-fitting; benchmark all three for high-dimensional problems.
- Like stepwise selection, best-subset chooses by criteria computed on the same data — cross-validate to assess true predictive performance.
- `regsubsets()` returns only the *single* best subset per size; near-ties can be missed and selection becomes unstable across resamples. Bootstrap to assess stability.
- Pre-specification of a subset based on substantive theory generally beats any data-driven search; reserve best-subset for genuinely exploratory work.
- For GLMs, use `bestglm::bestglm` or stepwise on AIC; pure best-subset enumeration is OLS-specific in `leaps`.

7.4.9 R Packages Used

`leaps::regsubsets` for branch-and-bound best-subset on linear models; `bestglm::bestglm` for GLMs; `glmnet` for lasso/elastic-net alternatives that scale to high p; `caret` and `tidymodels` for cross-validated subset comparison; `MuMIn::dredge` for model-averaging extensions.

7.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.4.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.5 Beta Regression

7.5.1 Introduction

Beta regression models continuous outcomes constrained to the open unit interval $(0, 1)$ — proportions, percentages expressed as fractions, biomarker ratios that range from 0 to 1. Ordinary least squares fails on such data: it can predict values outside $(0, 1)$, the residuals are typically heteroscedastic and asymmetric, and confidence intervals on the boundary are unreliable. Beta regression respects the bound, models heteroscedasticity through a precision parameter,

and returns interpretable coefficients on the logit scale (or any other supported link).

7.5.2 Prerequisites

A working understanding of the Beta distribution, generalised linear models, and the logit link as a transformation of probabilities to the real line.

7.5.3 Theory

The model parameterises Y as Beta with mean μ and precision ϕ:

$$Y \mid x \sim \text{Beta}(\mu\phi, (1-\mu)\phi),$$

with $\text{logit}(\mu) = x^\top \beta$ and $\phi > 0$ a precision parameter (larger ϕ means tighter concentration around μ). Variable-dispersion beta regression allows ϕ to depend on covariates as well, capturing heteroscedasticity directly. Coefficients are interpretable as log-odds ratios on the proportion: a unit increase in x_j multiplies the odds of "success-vs-failure" of the proportion by $\exp(\beta_j)$.

Values exactly 0 or 1 are excluded from the standard model; common remedies include a small linear shift ($y^* = (y(n-1)+0.5)/n$) or a zero-one-inflated beta model that adds explicit point-mass components.

7.5.4 Assumptions

Outcome strictly in $(0, 1)$ (no exact 0 or 1 unless using zero-one-inflated extensions); logit (or chosen) link appropriate; independent observations.

7.5.5 R Implementation

```r
library(betareg)

set.seed(2026)
x <- rnorm(200)
eta <- 0.2 + 0.8 * x
mu  <- plogis(eta)
phi <- 10
y <- rbeta(200, mu * phi, (1 - mu) * phi)

fit <- betareg(y ~ x)
summary(fit)

# Predictions on the probability scale
head(predict(fit, type = "response"))
```

7.5.6 Output & Results

`summary(fit)` reports coefficients on the logit scale (with Wald z-tests), the precision parameter $\hat{\phi}$, and pseudo-R^2. `predict(...,type="response")` returns predictions back on the $(0, 1)$ scale.

7.5.7 Interpretation

A reporting sentence: "Beta regression of the bounded outcome on x estimated $\hat{\beta} = 0.79$ on the logit scale (odds ratio 2.20, 95 % CI 1.79 to 2.71, $p < 0.001$); the precision parameter was $\hat{\phi} = 9.6$, indicating moderate concentration around the conditional mean. A 1-SD increase in x is associated with a roughly two-fold increase in the odds of a higher proportion." Always report both the coefficient and the precision; the precision controls the implied uncertainty.

7.5.8 Practical Tips

- Outcomes exactly equal to 0 or 1 need either a small shift transformation ($y^* = (y(n-1) + 0.5)/n$) or a zero-one-inflated beta model (`gamlss::gamlss(family=BEINF)` or `zoib::zoib`).
- Variable-dispersion beta regression — modelling ϕ on covariates — handles heteroscedasticity directly via `betareg(y~x|z)` syntax.
- For longitudinal or clustered proportion data, `glmmTMB(family=beta_family())` adds random effects to a beta GLM.
- The default link is logit; alternatives (probit, cloglog) are available via the `link` argument and rarely change conclusions materially.
- Pearson and quantile residuals are the right diagnostic tools; standard residuals can be misleading for bounded outcomes.
- Compare beta regression against a quasi-binomial GLM with a logit link; the latter is sometimes simpler and gives similar conclusions for moderate dispersion.

7.5.9 R Packages Used

`betareg` for `betareg()` with fixed and variable dispersion; `gamlss` and `gamlss.dist` for zero-one-inflated beta and other bounded distributions; `glmmTMB` for beta GLMMs; `zoib` for fully Bayesian zero-one-inflated beta regression; `DHARMa` for residual diagnostics across non-Gaussian families.

7.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.5.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.6 Centring and Scaling Predictors

7.6.1 Introduction

Centring (subtracting the mean) and scaling (dividing by the standard deviation) are simple transformations of continuous predictors that change coefficient interpretation without changing model fit. Centring shifts the intercept to a meaningful value (the predicted outcome at the mean of every predictor) and reduces collinearity between main effects and interaction terms. Scaling makes coefficient magnitudes comparable across predictors that are on different units. Both transformations are routine pre-processing steps in regression with interactions or in any analysis that benchmarks predictor importance by coefficient size.

7.6.2 Prerequisites

A working understanding of multiple linear regression, intercepts, and the geometric meaning of regression coefficients.

7.6.3 Theory

Centring: $X^c = X - \bar{X}$. The intercept becomes the predicted Y at the mean of every centred predictor; the slopes are unchanged. For models with interactions $X \cdot Z$, centring reduces the artefactual correlation between X, Z, and $X \cdot Z$ that inflates standard errors and complicates interpretation.

Scaling: $X^s = X/\mathrm{SD}(X)$. The coefficient on X^s is the predicted change in Y per one-standard-deviation increase in X, on whatever scale Y is measured. Combined with centring, the result is the standard z-score; standardised coefficients are directly comparable across predictors on different units.

Neither transformation changes R^2, predictions, residual structure, or significance tests; only the coefficient values and their interpretation change.

7.6.4 Assumptions

Continuous predictors. For categorical (factor) predictors, "centring" via sum-to-zero contrasts achieves an analogous effect on the intercept interpretation.

7.6.5 R Implementation

```r
d <- mtcars

# Centred predictors
d$wt_c <- d$wt - mean(d$wt)
d$hp_c <- d$hp - mean(d$hp)

# Centred + scaled (z-scored)
d$wt_z <- scale(d$wt)[, 1]
d$hp_z <- scale(d$hp)[, 1]

fit_raw  <- lm(mpg ~ wt + hp,     data = d)
fit_cent <- lm(mpg ~ wt_c + hp_c, data = d)
fit_std  <- lm(mpg ~ wt_z + hp_z, data = d)

# Intercept differs; slopes for raw & centred equal; slopes for
 ↪  standardised are per-SD
coef(fit_raw);  coef(fit_cent);  coef(fit_std)
```

7.6.6 Output & Results

Three fits with identical R^2, F-statistic, and residuals but different coefficient interpretations. The raw model's intercept is Y at $X = 0$ (often outside the data range); the centred model's intercept is Y at the mean of every predictor; the standardised model's slopes are per-SD changes in Y.

7.6.7 Interpretation

A reporting sentence: "Standardised coefficients show that a 1-SD increase in vehicle weight is associated with a 4.2-mpg decrease in fuel economy, while a 1-SD increase in horsepower is associated with a 2.3-mpg decrease (95 % CI −5.1 to −3.3 and −3.5 to −1.0 respectively); per standard deviation, weight is the

stronger predictor of fuel economy in this sample." Standardised coefficients are the right summary when comparing strengths of predictors on different units.

7.6.8 Practical Tips

- Centre predictors before adding interactions; this reduces the mechanical correlation between main and interaction terms and stabilises SEs.
- Scale predictors when benchmarking importance across units; a coefficient of 0.001 on annual income looks small but might be the strongest predictor on a per-SD basis.
- `scale()` returns a matrix with attributes; index with `[,1]` to get a plain numeric vector.
- Store means and SDs from training data when applying the same standardisation to test data; using the test-set mean/SD leaks information.
- Do not centre or scale the outcome unless interpretation requires it; transformations on Y change predictions and R^2 in ways that may be undesirable.
- For penalised regression (lasso, ridge, elastic net), `glmnet` standardises predictors internally and returns coefficients on the original scale; do not pre-standardise.

7.6.9 R Packages Used

Base R `scale()` for centring and scaling; `recipes::step_normalize()` and `step_center()` for tidymodels-flavoured pre-processing pipelines; `parameters::standardize_parameters()` for post-hoc standardisation of fitted-model coefficients.

7.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.6.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv

- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.7 Contrasts in R

7.7.1 Introduction

A contrast matrix maps a factor's levels to the numeric indicator columns that enter the regression design matrix. R ships with four built-in schemes — treatment, sum, Helmert, polynomial — each testing different hypotheses about group differences. Beyond these, custom contrasts let analysts test pre-specified comparisons (e.g. "first arm vs the average of the other two") directly in the coefficient table, without post-hoc multiple comparisons. Choosing and reporting the contrast scheme is a routine but consequential modelling decision.

7.7.2 Prerequisites

A working understanding of dummy coding, the role of the design matrix in regression, and basic familiarity with factor handling in R.

7.7.3 Theory

For a factor with k levels, any contrast scheme produces $k-1$ numeric columns. Built-in choices:

- **Treatment (`contr.treatment`)**: first level as reference; coefficients are level-vs-reference differences. Default for unordered factors.
- **Sum (`contr.sum`)**: sum-to-zero deviation coding; intercept becomes the grand mean and coefficients are level-vs-grand-mean deviations.
- **Helmert (`contr.helmert`)**: orthogonal successive comparisons (level j vs the mean of levels $1, \ldots, j-1$). Useful for ordered or sequential designs.
- **Polynomial (`contr.poly`)**: linear, quadratic, cubic, etc. trends for ordered factors. Default for ordered factors.

Custom contrasts are matrices whose columns define the comparisons of interest; the resulting coefficients represent exactly those comparisons. For orthogonal contrasts (columns sum to zero and pairwise products sum to zero), tests of each coefficient are independent and no multiplicity adjustment is needed within the pre-planned set.

7.7.4 Assumptions

Standard regression assumptions; the contrast choice does not change fit or predictions, only the coefficient interpretation.

7.7.5 R Implementation

```r
d <- mtcars
d$cyl <- factor(d$cyl, levels = c("4", "6", "8"))

# Compare built-in contrasts
contr.treatment(3)
contr.sum(3)
contr.helmert(3)
contr.poly(3)

# Apply a specific contrast to a factor
contrasts(d$cyl) <- contr.helmert(3)
fit <- lm(mpg ~ cyl, data = d)
summary(fit)

# Custom contrast: 4 vs (6 + 8), and 6 vs 8
custom <- cbind("4_vs_68" = c(2, -1, -1),
                "6_vs_8"  = c(0, 1, -1))
contrasts(d$cyl) <- custom
fit_custom <- lm(mpg ~ cyl, data = d)
summary(fit_custom)
```

7.7.6 Output & Results

Each coefficient in the regression output tests the comparison defined by the corresponding column of the contrast matrix. Treatment contrasts produce level-vs-reference comparisons; sum contrasts produce level-vs-grand-mean; custom contrasts produce whatever comparisons the user specifies.

7.7.7 Interpretation

A reporting sentence: "Custom orthogonal contrasts compared 4-cylinder cars against the average of 6- and 8-cylinder cars (estimate -9.36, 95 % CI -12.0 to -6.7, $p < 0.001$) and 6-cylinder against 8-cylinder cars (estimate -4.65, 95 % CI -7.6 to -1.7, $p = 0.003$); orthogonal contrasts allow these two pre-specified hypotheses to be tested without multiplicity adjustment." Always state the contrast scheme.

7.7.8 Practical Tips

- Match the contrast scheme to the scientific question rather than the software default; treatment contrasts are convenient but rarely optimal for ANOVA-style questions.
- Custom contrasts must be linearly independent (rank $k-1$); orthogonal contrasts add the property that pairwise within-column products sum to zero.
- For Type III sums of squares in unbalanced ANOVA designs, use sum-to-zero (`contr.sum`) or Helmert; treatment contrasts produce non-orthogonal SS that confound main effects with interactions.
- For ordered factors, the polynomial default tests linear, quadratic, etc. trends; if you want pairwise group differences, override with `contr.treatment`.
- Centring continuous predictors is conceptually analogous to using effect-coding for factors; both shift the intercept to a meaningful reference and reduce collinearity with interactions.
- Document the contrast scheme in the methods section: "treatment contrasts with placebo as reference" or "Helmert contrasts" leaves no room for misreading.

7.7.9 R Packages Used

Base R `contrasts()`, `contr.treatment()`, `contr.sum()`, `contr.helmert()`, `contr.poly()`; `MASS::contr.sdif()` for successive-differences contrasts; `multcomp::glht()` for testing arbitrary linear combinations of coefficients with multiplicity correction; `emmeans` for marginal means and contrast-stable post-hoc inference.

7.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.7.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models

- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.8 Cook's Distance

7.8.1 Introduction

Cook's distance is the most widely used single-number summary of an observation's influence on a regression fit. It answers a direct question: how much would the fitted values change if we removed this observation? Cook's distance combines the residual (how badly the model fits this point) and the leverage (how unusual this point is in the predictor space) into one quantity, making it the first port of call for influence diagnostics in linear regression.

7.8.2 Prerequisites

A working understanding of leverage (hat values), standardised residuals, and the leave-one-out perspective on regression diagnostics.

7.8.3 Theory

For observation i with standardised residual r_i and hat value h_{ii},

$$D_i = \frac{r_i^2}{p+1} \cdot \frac{h_{ii}}{(1-h_{ii})^2},$$

where $p+1$ is the number of regression parameters. The product structure makes Cook's distance large only when both the residual is sizable *and* the leverage is non-trivial; either alone is insufficient.

Conventional thresholds: $D_i > 1$ indicates strong influence (the regression effectively hinges on this point); $D_i > 4/n$ indicates moderate influence and warrants investigation. The relative magnitude across observations is often more informative than absolute thresholds — a few points dominating the Cook's distance distribution deserve scrutiny regardless of cutoffs.

7.8.4 Assumptions

Classical OLS assumptions, including residual Normality (Cook's distance is sensitive to outliers in the response). For robust alternatives, DFBETA (per-coefficient influence) or robust regression with leverage diagnostics is preferable.

7.8.5 R Implementation

```r
fit <- lm(mpg ~ wt + hp, data = mtcars)

cooks <- cooks.distance(fit)
head(sort(cooks, decreasing = TRUE))

# Visualise
plot(fit, which = 4)
abline(h = 4 / nrow(mtcars), col = "red", lty = 2)

# Influential cars
mtcars[cooks > 4 / nrow(mtcars), ]
```

7.8.6 Output & Results

`cooks.distance(fit)` returns a vector of D_i for every observation. The diagnostic plot (`plot(fit,which=4)`) displays them as sticks, with the largest values labelled. Subsetting at the threshold identifies candidate influential observations for further inspection.

7.8.7 Interpretation

A reporting sentence: "Three vehicles had Cook's distance exceeding $4/n = 0.125$ — the Maserati Bora, the Cadillac Fleetwood, and the Lincoln Continental — but refitting without them changed the coefficient on weight from -3.88 to -3.65 (95 % CI -5.65 to -1.65 vs -5.20 to -2.10 originally), with conclusions unchanged." Always run a sensitivity analysis after identifying influential points; never silently remove them.

7.8.8 Practical Tips

- Cook's distance is sensitive to residual outliers as well as leverage; for robust alternatives, examine DFBETAs and consider robust regression.
- An observation with $D_i > 1$ essentially guarantees the regression hinges on it; investigate the data point for errors before deciding whether to retain or exclude.
- Report sensitivity analyses with influential points removed; do not silently exclude.

- Cook's distance summarises overall influence; DFBETAs identify which specific coefficients are most affected by each observation.
- For GLMs, use `influence.measures(fit)`, which returns DFBETAs, hat values, and Cook's-distance analogues for the chosen family.
- Influence diagnostics are not a substitute for substantive judgement — a high-leverage observation may be the most informative data point in the sample if it is correct.

7.8.9 R Packages Used

Base R `cooks.distance()`, `influence.measures()`, `dfbetas()`, `hatvalues()`; `car::influencePlot()` for an integrated visual; `performance::check_outliers()` for a tidy diagnostic interface; `MASS::rlm()` and `robustbase::lmrob()` for robust regression alternatives.

7.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.8.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.9 Dirichlet Regression

7.9.1 Introduction

Dirichlet regression models compositional outcomes — vectors of proportions that sum to 1, such as relative cell-type composition in a tissue, time budgets across activities, microbial relative abundances, or budget shares across categories. The Dirichlet distribution is the multivariate generalisation of the Beta and lives on the unit simplex; Dirichlet regression places covariate effects on each component while preserving the unit-sum constraint inherent in compositional data.

7.9.2 Prerequisites

A working understanding of beta regression, the unit simplex, and compositional data as fundamentally different from independent proportions.

7.9.3 Theory

A composition $Y = (Y_1, \dots, Y_k)$ with $\sum_j Y_j = 1$ and $Y_j > 0$ follows a Dirichlet distribution $\mathrm{Dir}(\alpha_1, \dots, \alpha_k)$. Dirichlet regression parameterises the component means as

$$\mu_j = \frac{\exp(x^\top \beta_j)}{\sum_l \exp(x^\top \beta_l)},$$

with $\alpha_j = \mu_j \phi$ where ϕ is a precision parameter. The structure mirrors multinomial logistic regression, with one reference component fixed for identifiability and the other $k - 1$ sets of coefficients estimated.

For very small samples or high precision, the Dirichlet can be unstable; weakly informative Bayesian priors (`brms` with `family=dirichlet()`) are an alternative.

7.9.4 Assumptions

Components sum to 1 with all components strictly positive (zeros require perturbation or zero-inflated extensions); independent observations; the Dirichlet variance structure is appropriate for the data.

7.9.5 R Implementation

```
library(DirichletReg)

set.seed(2026)
n <- 150
x <- rnorm(n)
```

```
a1 <- exp(0.3 * x); a2 <- exp(-0.2 * x); a3 <- exp(0.5 * x)
Y <- t(apply(cbind(a1, a2, a3), 1,
             function(a) {
               phi <- 20
               rd <- rgamma(length(a), a * phi, 1)
               rd / sum(rd)
             }))

d <- data.frame(Y = DR_data(as.data.frame(Y)), x = x)
fit <- DirichReg(Y ~ x, data = d)
summary(fit)
```

7.9.6 Output & Results

`DirichReg()` returns coefficients per non-reference component on the multinomial logit scale plus the precision parameter. Predictions back on the simplex come from `predict()`; they always satisfy the unit-sum constraint by construction.

7.9.7 Interpretation

A reporting sentence: "Dirichlet regression showed that x positively associated with components 1 and 3 (multinomial-logit slopes 0.5 and 0.7 vs the reference) and negatively with component 2 (-0.4); predicted compositions at $x = -1, 0, +1$ shifted from (0.30, 0.40, 0.30) to (0.33, 0.30, 0.37) to (0.36, 0.22, 0.42), preserving the unit-sum constraint." Always communicate compositional results as predicted compositions; raw multinomial-logit coefficients are difficult to interpret.

7.9.8 Practical Tips

- Exact zeros or exact ones in the composition need perturbation before fitting; `compositions::clr()` and `zCompositions::cmultRepl()` provide standard zero-replacement algorithms.
- For compositional data with many components ($k > 6$), consider aggregating to fewer meaningful categories or using additive-log-ratio (ALR) or centred-log-ratio (CLR) transformations followed by ordinary multivariate regression.
- For zero-inflated compositions (many exact zeros), use zero-inflated Dirichlet or Zadeh-Aitchison ZI-logratio models.
- Interpretation is cleanest via predicted compositions plotted against covariate values; raw coefficients on the multinomial-logit scale are rarely communicative.
- For paired or longitudinal compositional data, a paired Dirichlet model or a mixed-effects extension is needed; ordinary Dirichlet regression assumes

independence across observations.
- For Bayesian Dirichlet regression with weakly informative priors, `brms(family=dirichlet())` provides a familiar interface.

7.9.9 R Packages Used

`DirichletReg` for the canonical `DirichReg()` implementation; `compositions` and `zCompositions` for compositional data pre-processing; `brms` for Bayesian Dirichlet regression; `aldex2` for differential-abundance compositional analysis in microbiome work.

7.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.9.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.10 Dummy Coding

7.10.1 Introduction

Categorical predictors in regression must be encoded as numeric indicator variables before they enter the linear predictor. R's default is treatment-contrast

coding (also called dummy coding), where one level is the implicit reference and the remaining $k-1$ levels each get a 0/1 indicator. The choice of reference level is partly arbitrary and partly substantive: the reference becomes the baseline against which all other levels are compared, so it should be the level that makes scientific sense — placebo in a clinical trial, "no symptom" in symptom-based predictors, the largest stratum in observational studies.

7.10.2 Prerequisites

A working understanding of categorical variables, factor handling in R, and the role of design matrices in regression.

7.10.3 Theory

For a factor with k levels, treatment coding produces a $k \times (k-1)$ contrast matrix. The reference level has all-zero indicator entries; each other level has a 1 in its own column and 0 elsewhere. Substituting into the linear model:

- Intercept = mean of the reference level (in a model with only the factor as predictor).
- Coefficient on indicator j = difference $\mu_j - \mu_{\text{ref}}$.

Alternative coding schemes — sum-to-zero (`contr.sum`), Helmert (`contr.helmert`), polynomial (`contr.poly`) — change the meaning of the intercept and coefficients but produce identical fits. Ordered factors default to polynomial contrasts in R, which decompose the trend into linear, quadratic, etc. components; override with `contrasts(f)<-contr.treatment` if you want familiar group differences.

7.10.4 Assumptions

Same as the underlying regression: linearity in the linear predictor, etc. The choice of contrast affects coefficient interpretation but not model fit, residuals, or predictions.

7.10.5 R Implementation

```r
# Default: treatment contrasts with first level as reference
fit <- lm(mpg ~ factor(cyl), data = mtcars)
summary(fit)

# Change reference level
mtcars$cyl <- factor(mtcars$cyl)
mtcars$cyl <- relevel(mtcars$cyl, ref = "6")
fit2 <- lm(mpg ~ cyl, data = mtcars)
summary(fit2)
```

```r
# Inspect contrast matrix
contrasts(mtcars$cyl)
```

7.10.6 Output & Results

The fitted intercept equals the reference-level mean; each coefficient equals the difference from the reference. Re-leveling changes the intercept and coefficients but does not change predictions or residuals.

7.10.7 Interpretation

A reporting sentence: "Compared to 4-cylinder cars (reference; mean 26.7 mpg), 6-cylinder cars averaged 6.9 mpg less ($p < 0.001$) and 8-cylinder cars averaged 11.6 mpg less ($p < 0.001$). Treatment contrasts with 4-cylinder as reference were used throughout." Always state the reference level explicitly; reviewers and reanalysts cannot tell from coefficients alone.

7.10.8 Practical Tips

- Choose the reference level to match the scientific question — placebo, baseline, control, or "most common" — rather than relying on alphabetic default.
- Use `relevel(f,ref="...")` or `forcats::fct_relevel()` to set a specific reference; in `lm()`/`glm()`, the first factor level is always the reference.
- For ordered factors, R defaults to polynomial contrasts (`contr.poly`); override with `contrasts(f)<-contr.treatment` if you want pairwise group differences.
- For ANOVA-style hypothesis testing, sum-to-zero contrasts (`contr.sum`) make the intercept the grand mean and produce orthogonal Type II / Type III sums of squares; specify with `options(contrasts=c("contr.sum","contr.poly"))`.
- Inspect contrast matrices with `contrasts(f)` to confirm what is actually being fitted; mismatch between expected and actual contrasts is a common source of misinterpreted output.
- Document the contrast type in methods: "treatment contrasts with placebo as reference" leaves no room for ambiguity.

7.10.9 R Packages Used

Base R `factor()`, `relevel()`, `contrasts()`, `contr.treatment()`, `contr.sum()`, `contr.helmert()`, `contr.poly()`; `forcats` for tidy factor manipulation including `fct_relevel()`; `emmeans::emmeans()` for marginal means and post-hoc contrasts that are interpretation-stable across coding choices.

7.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.10.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.11 Effect (Deviation) Coding

7.11.1 Introduction

Effect coding — also called deviation coding or sum-to-zero coding — represents a factor's levels as deviations from the grand mean rather than as differences from a single reference level. The intercept becomes the grand mean and each coefficient becomes a level-vs-grand-mean deviation, with all level deviations summing to zero. This symmetric interpretation is the right framing for factorial ANOVA and for any analysis where no single level is naturally privileged as "reference".

7.11.2 Prerequisites

A working understanding of dummy (treatment) coding, factorial ANOVA, and the difference between Type I, II, and III sums of squares in unbalanced designs.

7.11.3 Theory

For a k-level factor, effect coding produces $k-1$ coded columns. Each column codes one non-reference level as $+1$, the implicit reference level as -1, and all other levels as 0. The intercept then equals the grand mean of cell means; the coefficient on each column is the deviation of that level from the grand mean. The implicit reference level's deviation is the negative sum of the explicit coefficients (since deviations sum to zero by construction).

Effect coding matters most in factorial ANOVA with unbalanced designs: Type III sums of squares (the standard ANOVA approach in many disciplines) require orthogonal contrasts to give interpretable interaction tests, and effect coding (or Helmert) is the right choice. Treatment contrasts give Type II SS that confound main effects with interactions.

7.11.4 Assumptions

Same as regression; the contrast choice does not change fit or predictions, only coefficient interpretation.

7.11.5 R Implementation

```r
d <- mtcars
d$cyl <- factor(d$cyl)

# Default: treatment contrasts
contrasts(d$cyl)

# Effect (sum-to-zero) contrasts
contrasts(d$cyl) <- contr.sum(3)
contrasts(d$cyl)

fit_eff <- lm(mpg ~ cyl, data = d)
summary(fit_eff)

# Intercept = grand mean of cell means
mean(tapply(d$mpg, d$cyl, mean))
```

7.11.6 Output & Results

The intercept equals the grand mean (or grand mean of cell means in unbalanced designs). Each coefficient on a coded column equals the deviation of that level from the grand mean. The deviation of the implicit reference level equals minus the sum of the explicit coefficients.

7.11.7 Interpretation

A reporting sentence: "Under effect coding (`contr.sum`), the intercept (22.5 mpg) is the grand mean of cell means; 4-cylinder cars averaged 4.4 mpg above the grand mean ($p < 0.001$), 6-cylinder cars 0.5 mpg below ($p = 0.66$), and 8-cylinder cars (implied as the negative sum of the other two coefficients) 3.9 mpg below. Effect coding was used to support Type III SS interpretation of interactions in the factorial design." Always state the contrast choice when reporting ANOVA-style coefficients.

7.11.8 Practical Tips

- Use effect coding for factorial ANOVA with Type III SS; combine with `car::Anova(fit,type=3)` for interpretable interaction tests in unbalanced designs.
- Set effect coding via `contrasts(factor)<-contr.sum(k)` *before* fitting; changing contrasts on a fitted model does not retroactively reinterpret coefficients.
- Effect coding's interpretation is symmetric across levels; no level is privileged as "reference".
- Treatment (dummy) coding is more common in clinical reporting where a placebo or control condition naturally serves as reference; effect coding dominates experimental psychology and factorial ANOVA.
- For Type III SS with unbalanced factorial designs, effect coding (or Helmert) is essentially required; treatment coding produces non-orthogonal SS that depend on factor order.
- The implicit reference level (typically the last) is silent in the output; the user must compute its deviation as the negative sum of the explicit coefficients.

7.11.9 R Packages Used

Base R `contr.sum()` and `contrasts<-`; `car::Anova()` with `type=3` for orthogonal-contrast Type III ANOVA; `emmeans` for marginal means and contrast-stable post-hoc inference; `afex` for repeated-measures ANOVA with effect-coded contrasts as the default.

7.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.11.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.12 Elastic Net

7.12.1 Introduction

Elastic net combines the L1 penalty of lasso with the L2 penalty of ridge into a single hybrid regulariser. It performs variable selection (like lasso) while retaining the stability under correlated predictors that lasso famously lacks (which is what ridge provides). The mixing parameter $\alpha \in [0, 1]$ moves continuously between the two extremes — $\alpha = 0$ is pure ridge, $\alpha = 1$ is pure lasso, and intermediate values blend their behaviours. For high-dimensional biomedical data with groups of correlated predictors (gene expression, multiple imaging features), elastic net is now the standard default.

7.12.2 Prerequisites

A working understanding of ridge regression, lasso regression, and the differing geometries of L1 versus L2 penalties.

7.12.3 Theory

Elastic net minimises

$$\|y - X\beta\|^2 + \lambda[\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2^2]\,.$$

The L1 component drives some coefficients exactly to zero (selection), while the L2 component shares shrinkage across correlated predictors so that all-or-

nothing selection failures are avoided. Where lasso would pick one of two perfectly correlated predictors arbitrarily and zero the other, elastic net keeps both with equal coefficients (the "grouping effect"), often a more honest reflection of the underlying signal.

Two hyperparameters need tuning: α (ratio of L1 to total penalty) and λ (overall strength). Cross-validation over a grid of α values, with λ tuned within each α via `cv.glmnet()`, finds the joint optimum.

7.12.4 Assumptions

Standard linear-model assumptions on residuals; predictors are standardised (`glmnet` does this internally). For inference on selected coefficients, post-selection adjustments apply just as in lasso.

7.12.5 R Implementation

```r
library(glmnet)

X <- as.matrix(mtcars[, -1])
y <- mtcars$mpg

set.seed(2026)
# Two-dimensional CV: choose alpha and lambda jointly
alpha_grid <- c(0, 0.25, 0.5, 0.75, 1)
results <- lapply(alpha_grid, function(a) {
  cv <- cv.glmnet(X, y, alpha = a)
  c(alpha = a, lambda_min = cv$lambda.min, cvm_min = min(cv$cvm))
})
do.call(rbind, results)

# Best alpha:
best_alpha <- 0.5
fit_best <- cv.glmnet(X, y, alpha = best_alpha)
coef(fit_best, s = "lambda.min")
```

7.12.6 Output & Results

A grid of cross-validated mean squared errors across α values shows the best joint tuning; refitting with the chosen α produces shrunken coefficients with some at zero. The combination typically achieves lower CV error than either lasso or ridge alone for correlated predictors.

7.12.7 Interpretation

A reporting sentence: "Elastic net with $\alpha = 0.5$ and cross-validated λ_{1se} retained four of ten predictors with shrunken coefficients; the resulting cross-validated MSE was 6.0 mpg², lower than the lasso ($\alpha = 1$, MSE 6.3) and ridge ($\alpha = 0$, MSE 7.2) alternatives, reflecting the grouping benefit on correlated engine-size predictors." Report all three CV errors when justifying the elastic-net choice.

7.12.8 Practical Tips

- A common starting value is $\alpha = 0.5$; tune jointly with λ over a grid.
- Elastic net is the default for high-dimensional biomedical data with correlated predictors (gene expression, imaging features); rarely should pure lasso or pure ridge be chosen without trying elastic net.
- `caret::train(method="glmnet")` and `tidymodels` automate joint tuning across both hyperparameters.
- For mostly uncorrelated sparse predictors, pure lasso ($\alpha = 1$) may suffice and is computationally cheaper.
- For maximum stability under severe multicollinearity, lean closer to ridge (α near 0); the variable-selection benefit is small when nearly every predictor matters.
- For non-Gaussian outcomes, the family argument extends elastic net to logistic, Poisson, and Cox regression.

7.12.9 R Packages Used

`glmnet` for the canonical elastic-net implementation with cross-validated tuning; `caret` and `tidymodels` for unified joint α-λ tuning workflows; `selectiveInference` for post-selection inference; `coxnet` (in `glmnet`) for elastic-net Cox regression on survival outcomes.

7.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.12.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors

- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.13 GAMs: Introduction

7.13.1 Introduction

Generalised additive models (GAMs) replace the strict linear predictor of GLMs with a sum of smooth functions of predictors: $g(\mathbb{E}[Y]) = \beta_0 + \sum_j s_j(X_j)$. Each s_j is a smooth, data-driven function estimated alongside the parametric components, so GAMs combine the interpretability of linear regression (additivity, coefficient-style summaries) with the flexibility of non-parametric smoothers. They are the standard tool whenever a continuous predictor's effect is suspected to be non-linear and theoretical grounds for a specific functional form are absent.

7.13.2 Prerequisites

A working understanding of generalised linear models, basis functions for smooth curves (B-splines, thin-plate splines), and the bias-variance trade-off that motivates smoothing penalties.

7.13.3 Theory

Each smooth function s_j is represented as a linear combination of basis functions with a smoothing penalty controlling wiggliness. `mgcv::gam()` defaults to penalised thin-plate regression splines, with the smoothing parameter selected automatically by REML (recommended) or generalised cross-validation (GCV). The effective degrees of freedom (edf) for each smooth quantify fitted complexity: edf = 1 means the smooth has reduced to a straight line, higher edf indicates more curvature. The penalty trades fit against smoothness and is what prevents over-fitting at high basis dimensions.

For categorical predictors, factor effects enter as ordinary parametric terms; for interactions between continuous predictors, tensor product smooths (`te()`) or `ti()` for pure interactions are available.

7.13.4 Assumptions

Standard GLM assumptions on the conditional distribution of the outcome; smooth effects are additive (no automatic interaction detection); the basis dimension is large enough to capture the underlying smoothness (`gam.check()` diagnoses this).

7.13.5 R Implementation

```r
library(mgcv)

set.seed(2026)
d <- data.frame(
  x = runif(200, 0, 10),
  z = rnorm(200)
)
d$y <- 2 + sin(d$x) * 2 + 0.4 * d$z + rnorm(200, 0, 1)

fit <- gam(y ~ s(x) + z, data = d)
summary(fit)
plot(fit, pages = 1, shade = TRUE)

# Check and predict
check.gam <- gam.check(fit)
predict(fit, newdata = data.frame(x = 5, z = 0))
```

7.13.6 Output & Results

`summary(fit)` reports parametric coefficients (with t tests) and approximate F tests for each smooth (with edf and reference degrees of freedom). `plot(fit,shade=TRUE)` displays each smooth with a shaded confidence band around the estimated curve. `gam.check()` reports residual diagnostics and the basis-dimension k-index, which flags when more basis functions are needed.

7.13.7 Interpretation

A reporting sentence: "A GAM with a smooth function of x (edf 8.5, $F = 22.4$, $p < 0.001$) and a linear effect of z ($\hat{\beta} = 0.41$, $p < 0.001$) recovered the simulated sinusoidal pattern in x and the additive linear contribution of z; deviance explained 76 %, $\hat{R}^2_{\text{adj}} = 0.74$." Report edf for smooths and parametric coefficients for linear terms; never quote a single p-value for a smooth without its edf.

7.13.8 Practical Tips

- `s(x)` uses thin-plate regression splines by default; `s(x,bs="cr")` for cubic regression splines (faster for one-dimensional smooths); `te(x,z)` for tensor

product interactions between two continuous predictors.
- The basis dimension `k` defaults work for most smooth functions; raise it if `gam.check()` reports a low k-index, lower it for parsimony.
- Use `method="REML"` for smoothing-parameter selection; it is more reliable than the default GCV for finite samples.
- For non-Gaussian outcomes, switch the family: `gam(y~s(x),family=binomial)`, `family=poisson`, etc.
- `gratia::draw(fit)` produces ggplot-style smooth-effect plots with cleaner formatting than base `plot.gam()`.
- For very large datasets, use `mgcv::bam()` with `discrete=TRUE` for substantial speed-ups; the API is otherwise identical.

7.13.9 R Packages Used

`mgcv` for `gam()`, `bam()`, smooth bases, and REML smoothing-parameter selection; `gratia` for ggplot-style GAM diagnostics and component plots; `mgcViz` for additional GAM visualisations including QQ plots and 3D smooth surfaces; `gam` (not to be confused with `mgcv`) for the older Hastie-Tibshirani backfitting implementation.

7.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.13.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean

- Decision curves, NRI, IDI

7.14 GAMMs: Additive Mixed Models

7.14.1 Introduction

Generalised additive mixed models (GAMMs) bring together two powerful regression frameworks: smooth non-linear effects from generalised additive models (GAMs) and random effects from linear / generalised linear mixed models. The `mgcv::gam()` interface handles both in a single unified framework by treating random effects as a specific type of penalised smooth, making the combined model no harder to fit than a plain GAM. The result is the natural choice when clustered data also exhibit non-linear effects of continuous covariates — longitudinal trajectories, growth curves, ecological time series with site-level structure.

7.14.2 Prerequisites

A working understanding of generalised additive models (smooth effects, basis functions, penalised regression splines), linear mixed models, and the role of random effects in clustered data.

7.14.3 Theory

In `mgcv`, random effects are smooths with the basis specification `bs="re"`. A smooth of a grouping factor with this basis is mathematically equivalent to a random intercept; the smoothing penalty plays the role of the random-effect variance, and REML estimation of the smoothing parameter recovers the variance estimate. More complex random structures — random slopes, factor-by-smooth interactions — are constructed via tensor-product smooths or `s(group,bs="re")` combined with appropriate factor coding.

For non-Gaussian outcomes, change the `family` argument; the random-effect basis trick generalises to GLMM territory automatically.

7.14.4 Assumptions

Smooth-function assumptions (the true relationship is reasonably represented by penalised splines), random-effect Normality, conditional independence given random effects, and the appropriate family for the outcome.

7.14.5 R Implementation

```
library(mgcv)

set.seed(2026)
d <- data.frame(
```

```
  subject = factor(rep(1:20, each = 30)),
  time    = rep(seq(0, 10, length.out = 30), 20)
)
d$y <- rnorm(nrow(d)) + sin(d$time) * 2 +
       rep(rnorm(20, 0, 1.5), each = 30) +
       0.3 * d$time

fit <- gam(y ~ s(time) + s(subject, bs = "re"), data = d)
summary(fit)
plot(fit, pages = 1)
```

7.14.6 Output & Results

`summary(fit)` reports the effective degrees of freedom (edf) for each smooth, parametric coefficients, and an approximate F-test for each smooth term. The random-effect smooth contributes a single variance component summarised in the model summary; `plot(fit,pages=1)` displays each smooth on a single page.

7.14.7 Interpretation

A reporting sentence: "A GAMM with a smooth of time (edf 7.2, $F = 24.5$, $p < 0.001$) and a subject-level random intercept (variance 2.18) explained 72 % of the deviance; the smooth recovered the simulated sine-plus-linear trend faithfully and the random-intercept SD matched the simulation parameter to within Monte Carlo error." Always report edf for smooths and the random-effect variance.

7.14.8 Practical Tips

- `bs="re"` turns a smooth into a random effect; `s(group,bs="re")` is a random intercept; `s(group,x,bs="re")` is a random slope on x.
- Tensor product smooths (`te(x,z)`) model smooth interactions between two continuous predictors; `ti(x,z)` builds the pure interaction without the main effects, useful for ANOVA-like decomposition.
- For GLMM-type responses (binomial, Poisson, beta), specify `family=binomial(link="logit")` (etc.) and the same random-effect smooth syntax works.
- Use `mgcv::gam.check(fit)` for diagnostic plots of residuals and basis-dimension adequacy; raise `k` when basis dimensions are too small.
- `gratia::draw(fit)` produces ggplot-style visualisations of all model components, far cleaner than base `plot.gam()` for publications.
- For very large datasets ($n > 10{,}000$), use `mgcv::bam()` with `discrete=TRUE` for substantial speed-ups; the API is otherwise identical.

7.14.9 R Packages Used

`mgcv` for `gam()`, `bam()`, smooth bases, and REML smoothing-parameter selection; `gratia` for ggplot-style GAM diagnostics and component plots; `lme4` for comparative LMM fits; `gamm4` for GAMMs implemented via `lme4` for very large random-effect structures.

7.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.14.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.15 Generalised Estimating Equations

7.15.1 Introduction

Generalised estimating equations (GEE) fit marginal regression models for clustered data — repeated measures, longitudinal outcomes, familial correlations — without specifying random effects. Where mixed models target subject-specific (conditional) effects, GEE targets population-averaged (marginal) effects, answering the question "what is the average effect across the population?" rather than "what is the effect for a typical subject?" The two interpretations coincide

for linear models with identity link but diverge for non-linear links (logistic, log).

7.15.2 Prerequisites

A working understanding of generalised linear models, the difference between conditional and marginal effects, and basic familiarity with cluster-robust standard errors.

7.15.3 Theory

GEE specifies a mean structure $g(\mu_{ij}) = X_{ij}^\top \beta$ for observation j in cluster i and a *working correlation* matrix $R(\alpha)$ governing within-cluster dependence. Standard choices:

- **Independence**: ignore correlation; coefficients still consistent.
- **Exchangeable**: equal correlation between all pairs within cluster.
- **AR(1)**: correlation decays geometrically with time gap.
- **Unstructured**: each pairwise correlation estimated separately.

Coefficient estimates are consistent even under correlation misspecification; standard errors use the sandwich (robust) variance estimator. The working-correlation choice affects efficiency but not asymptotic consistency.

QIC (quasi-likelihood information criterion) generalises AIC for GEE and is used for selecting both the working correlation and the mean structure.

7.15.4 Assumptions

Independent clusters, correctly specified mean model, and missing-completely-at-random (MCAR) for missing observations. The MCAR requirement is stricter than the MAR assumption of mixed models, so heavy missingness can favour the latter.

7.15.5 R Implementation

```
library(geepack)

data(respiratory, package = "geepack")
fit <- geeglm(outcome ~ treat + age + sex, id = id, data = respiratory,
              family = binomial, corstr = "exchangeable")
summary(fit)

# AR(1) correlation
fit_ar <- geeglm(outcome ~ treat + age + sex, id = id, data = respiratory,
                 family = binomial, corstr = "ar1")
```

```
# QIC for correlation structure selection
QIC(fit); QIC(fit_ar)
```

7.15.6 Output & Results

`summary(fit)` returns marginal coefficients with sandwich-robust standard errors and Wald z-tests. `QIC()` provides a quasi-likelihood information criterion for comparing models with different working correlations or different mean structures.

7.15.7 Interpretation

A reporting sentence: "A GEE logistic regression with exchangeable working correlation estimated the population-averaged odds ratio for treatment at 3.7 (95 % CI 2.0 to 6.6, $p < 0.001$); standard errors used the sandwich variance estimator to protect against working-correlation misspecification, and QIC favoured exchangeable over AR(1) (ΔQIC $= 4$)." Always specify the working-correlation structure and use sandwich SEs.

7.15.8 Practical Tips

- Sandwich SEs are the standard inferential tool for GEE; do not rely on naive model-based SEs.
- Use QIC for selecting the working correlation; AIC and BIC do not apply because GEE is quasi-likelihood, not likelihood.
- Choose between mixed models and GEE based on the inferential question: mixed for subject-specific effects, GEE for population-averaged.
- For longitudinal binary outcomes, exchangeable working correlation is a sensible default; AR(1) when measurements are evenly spaced and correlation decays in time.
- For small numbers of clusters (< 40), sandwich SEs can be biased downward; use small-sample corrections (`geesmv` package) or fall back on mixed models with appropriate denominator df.
- Missing data: GEE requires MCAR for unbiased estimation, while mixed models tolerate MAR; consider this trade-off when choosing.

7.15.9 R Packages Used

`geepack` for `geeglm()` and QIC; `gee` for an alternative implementation; `geeM` for high-performance GEE on large datasets; `geesmv` for small-sample SE corrections; `multgee` for GEE with multinomial outcomes.

7.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.15.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.16 glmer: Generalised Mixed Models

7.16.1 Introduction

`lme4::glmer()` fits generalised linear mixed models (GLMMs) — binary, count, and other non-Gaussian outcomes with random effects. The function inherits the random-effects formula syntax of `lmer()` and the family / link conventions of `glm()`, combining them into a single estimation problem. For binary outcomes (logistic mixed models), Poisson rates with cluster structure, and other GLM-style problems with repeated measures or hierarchical sampling, `glmer()` is the standard tool. The newer `glmmTMB` package is faster, more flexible (it handles overdispersed and zero-inflated families natively), and increasingly the recommended default.

7.16.2 Prerequisites

A working understanding of generalised linear models, mixed-effects models, and link functions; familiarity with `lme4` formula syntax for random effects.

7.16.3 Theory

The GLMM is

$$g(\mathbb{E}[Y \mid u]) = X^\top \beta + Z^\top u, \qquad u \sim \mathcal{N}(0, \Sigma),$$

with g the link function (logit for binary, log for Poisson, log-odds-cumulative for ordinal). The random effects are placed on the linear-predictor scale, so interpreting their variance requires caution: a random-intercept SD of 1 on the logit scale corresponds to substantial between-cluster variability in baseline probability (± 1 SD spans a 0.27-to-0.73 baseline range from a centre of 0.50).

Estimation uses Laplace approximation by default; adaptive Gauss-Hermite quadrature (`nAGQ>1`) gives better accuracy at higher computational cost.

7.16.4 Assumptions

GLM-family assumptions on the conditional distribution, multivariate Normal random effects, conditional independence given random effects, and the appropriate link function for the outcome family.

7.16.5 R Implementation

```r
library(lme4); library(glmmTMB)

set.seed(2026)
# Binary outcome: 20 subjects x 10 trials
d <- expand.grid(subject = factor(1:20), trial = 1:10)
d$x <- rnorm(nrow(d))
subj_int <- rnorm(20, 0, 1.2)
d$lp <- -0.5 + subj_int[as.integer(d$subject)] + 0.5 * d$x
d$y <- rbinom(nrow(d), 1, plogis(d$lp))

# lme4 glmer (binomial)
fit <- glmer(y ~ x + (1 | subject), data = d, family = binomial)
summary(fit)

# glmmTMB equivalent with more features
fit_tmb <- glmmTMB(y ~ x + (1 | subject), data = d, family = binomial)
```

7.16.6 Output & Results

`summary(fit)` reports fixed-effect coefficients on the linear-predictor (logit, log) scale with standard errors, Wald z statistics, and p-values, plus random-effect variances and covariances and the AIC. Exponentiating fixed-effect coefficients gives odds ratios (binomial), rate ratios (Poisson), or other family-appropriate effect-size measures.

7.16.7 Interpretation

A reporting sentence: "A binomial GLMM estimated the population-level slope at 0.51 on the log-odds scale (OR = 1.66, 95 % CI 1.34 to 2.04, $p < 0.001$); the random-intercept standard deviation was 1.10, indicating substantial between-subject variability in baseline log-odds. Coefficients are conditional on subject; for marginal effects use `ggeffects::ggpredict`." Always state whether fixed-effect interpretations are conditional or marginal — they are different scales.

7.16.8 Practical Tips

- Laplace approximation is the default; for binary outcomes with small clusters, raise `nAGQ` to 7 or 11 for better accuracy.
- Convergence warnings are common with binary data; try `glmmTMB` for a more robust optimiser, or switch to Bayesian fitting (`brms`).
- Fixed-effect coefficients are conditional on the random effects; for marginal (population-averaged) effects, use generalised estimating equations (GEE) or post-process with `ggeffects::ggpredict()` or `marginaleffects::avg_predictions()`.
- For overdispersed counts, `glmmTMB(family=nbinom2)` is far cleaner than the deprecated `glmer.nb()`.
- For zero-inflated outcomes, `glmmTMB` handles the zero-inflation component natively; `lme4` does not.
- Monitor singular fits — random-effect variances estimated near zero — and simplify the random structure when they appear.

7.16.9 R Packages Used

`lme4` for `glmer()` with Laplace and adaptive Gauss-Hermite quadrature; `glmmTMB` for fast TMB-based GLMM fitting with overdispersed and zero-inflated families; `pbkrtest` for parametric-bootstrap LRTs on variance components; `ggeffects` and `marginaleffects` for marginal effects; `brms` for Bayesian GLMMs.

7.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.16.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring

- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.17 Hurdle Models

7.17.1 Introduction

Hurdle models split count outcomes into two distinct processes: a Bernoulli decision for zero versus non-zero, and a zero-truncated count distribution for the positive counts. They are appropriate when the question of "did anything happen?" is conceptually different from "given that something happened, how much?" — for example, healthcare visits (decided to see a doctor at all, then number of visits), insurance claims, hospital admissions for a rare condition.

7.17.2 Prerequisites

A working understanding of Poisson and negative binomial regression, the zero-inflated alternative, and the conceptual distinction between two-part and mixture models for zero-heavy counts.

7.17.3 Theory

The hurdle model is

$$\mathrm{P}(Y = 0) = 1 - \pi, \qquad \mathrm{P}(Y = y) = \pi \cdot \frac{f(y)}{1 - f(0)}, \quad y \geq 1,$$

with π the probability of clearing the hurdle (logistic regression on covariates) and $f(y)$ a count distribution (Poisson or NB) truncated at zero. The two parts are estimated jointly by maximum likelihood; covariates can differ between the zero and count submodels.

The contrast with zero-inflated models is conceptual: hurdle treats *all* zeros as arising from one Bernoulli process, while zero-inflated treats some zeros as

"structural" (always zero) and others as "sampling" (could have been positive but happened to be zero). Choose based on the scientific story.

7.17.4 Assumptions

A count outcome with a meaningful two-stage data-generating process; the chosen count distribution (Poisson or NB) is appropriate for the positive counts.

7.17.5 R Implementation

```r
library(pscl)

set.seed(2026)
d <- data.frame(x = rnorm(300))
d$y <- ifelse(rbinom(300, 1, plogis(0 + 0.4 * d$x)) == 0, 0,
              rpois(300, lambda = exp(1 + 0.5 * d$x)))

fit_hurdle <- hurdle(y ~ x | x, data = d)
summary(fit_hurdle)

# Zero-inflated for comparison
fit_zi <- zeroinfl(y ~ x | x, data = d)
AIC(fit_hurdle, fit_zi)
```

7.17.6 Output & Results

`summary(fit_hurdle)` returns two coefficient tables: the count part (zero-truncated Poisson or NB) and the zero hurdle (binomial). Each part has its own intercept and slope estimates with Wald z-tests. The AIC comparison against `zeroinfl` quantifies which framework better matches the data.

7.17.7 Interpretation

A reporting sentence: "A hurdle model decomposed the data into a binomial hurdle (each unit increase in x raised the log-odds of any positive count by 0.40, $p < 0.001$) and a zero-truncated Poisson count model (rate ratio 1.65 per unit increase in x among positive counts, 95 % CI 1.40 to 1.95). Hurdle was favoured over zero-inflated by AIC ($\Delta = 4$)." Always describe both parts and use the AIC comparison to justify the framework choice.

7.17.8 Practical Tips

- Hurdle is cleaner than zero-inflated when the scientific story has a clear two-stage interpretation; mixing the two approaches without justification confuses readers.

- For overdispersed positive counts, use `dist="negbin"` in the count part; this is essential for most real count data.
- Different covariates can drive the two parts; use `y~x_count|x_zero` to specify them separately.
- Zero-inflated assumes some zeros are structural and could not have been any other value; hurdle assumes all zeros come from one decision process.
- AIC and BIC compare hurdle and zero-inflated objectively when both are well-fitted; do not rely on visual inspection alone.
- For hierarchical hurdle models, `glmmTMB(family=truncated_nbinom2(),zif ormula=~...)` provides random-effect extensions.

7.17.9 R Packages Used

`pscl::hurdle()` for canonical hurdle models with Poisson, NB, or geometric count parts; `glmmTMB` for hurdle and zero-inflated mixed-effects extensions; `coun treg` for additional count-model variants and rootogram diagnostics; `DHARMa` for residual diagnostics across hurdle and zero-inflated families.

7.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.17.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.18 Interactions in Regression

7.18.1 Introduction

An interaction term — a product of two predictors entered alongside the main effects — expresses the substantive idea that the effect of one predictor depends on the value of another. Without interactions, a linear model assumes purely additive effects: a unit increase in X_1 produces the same change in Y regardless of X_2. Interactions relax this assumption and often capture the actual scientific hypothesis (treatment effect varies by age, biomarker effect varies by sex, dose-response curve depends on baseline severity). Reporting interactions correctly requires moving beyond main-effect coefficients to conditional effects at specific covariate values.

7.18.2 Prerequisites

A working understanding of multiple linear regression, the geometric meaning of regression coefficients, and centring continuous predictors as a tool for reducing collinearity.

7.18.3 Theory

For two continuous predictors X_1, X_2 and an interaction:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \varepsilon.$$

The conditional effect of X_1 is $\beta_1 + \beta_3 X_2$ — a function of X_2 rather than a single number. The "main effect" β_1 is the conditional effect of X_1 when $X_2 = 0$; if $X_2 = 0$ is outside the data range or not a meaningful reference, the main-effect coefficient is uninterpretable.

Centring continuous predictors before adding interactions makes $X_2 = 0$ correspond to the mean of X_2 and gives the main-effect coefficient the interpretation "effect at the average of the other predictor". It also reduces the mechanical collinearity between X_1, X_2, and $X_1 X_2$.

For categorical-by-continuous interactions, the continuous coefficient is the effect within the reference category and the interaction is the difference for the non-reference category.

7.18.4 Assumptions

Same as the underlying regression: linearity in the linear predictor (after the interaction is included), independent observations, etc. The interaction itself is a model choice, not an assumption.

7.18.5 R Implementation

```r
library(emmeans); library(ggeffects)

d <- mtcars
d$wt_c <- scale(d$wt, center = TRUE, scale = FALSE)[, 1]
d$hp_c <- scale(d$hp, center = TRUE, scale = FALSE)[, 1]

fit <- lm(mpg ~ wt_c * hp_c, data = d)
summary(fit)

# Marginal effects at selected levels
emtrends(fit, ~ hp_c, var = "wt_c",
         at = list(hp_c = c(-50, 0, 50)))

# Interaction plot
ggpredict(fit, terms = c("wt_c", "hp_c [-50, 0, 50]")) |> plot()
```

7.18.6 Output & Results

The interaction coefficient is on the product scale. `emtrends()` returns the conditional slope of one variable at chosen values of the other. `ggpredict()` plots predicted Y against X_1 at multiple levels of X_2, the canonical visualisation of an interaction.

7.18.7 Interpretation

A reporting sentence: "The effect of weight on mpg was more negative at high horsepower than at low (interaction coefficient 0.030, 95 % CI 0.007 to 0.053, $p = 0.01$); at the mean horsepower, each 1000-lb increase in weight reduced mpg by 3.5 (95 % CI −4.6 to −2.4); at horsepower 50 above the mean, the slope was −4.9, and at 50 below it was −2.0." Always report simple slopes at meaningful covariate values, not just the interaction coefficient.

7.18.8 Practical Tips

- Centre continuous predictors before adding interactions; this gives interpretable main effects ("effect at the mean of the other") and reduces collinearity between main and interaction terms.
- Never interpret "main effects" when a significant interaction is present without specifying the reference value of the moderator; the main effect alone is incomplete.
- Use `emmeans::emtrends()` or `marginaleffects::slopes()` to compute simple slopes at meaningful covariate values; pair with confidence intervals.
- Visualise interactions with `ggeffects::ggpredict()` plots showing Y against the focal predictor at several values of the moderator; this

communicates the interaction far better than coefficient tables.
- For categorical-by-categorical interactions, an interaction plot with separate lines per group is the standard display; for categorical-by-continuous, use facets or colour groups.
- Higher-order interactions (three-way and beyond) require much larger samples and careful interpretation; usually best avoided in confirmatory analyses.

7.18.9 R Packages Used

`emmeans` for marginal means, simple slopes, and contrast-stable post-hoc inference; `ggeffects` for ggplot-style predicted-effect plots; `marginaleffects` for unified marginal-effect computation across many model classes; `interactions` for purpose-built interaction probing tools.

7.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.18.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.19 Lasso Regression

7.19.1 Introduction

The lasso (least absolute shrinkage and selection operator) adds an L1 penalty to ordinary least squares, simultaneously shrinking coefficients and driving some of them exactly to zero. The L1 geometry — the absolute-value penalty has corners on each axis — makes the lasso the canonical method for high-dimensional regression where simultaneous estimation and variable selection is required. It dominates ridge regression when the truth is sparse (a small number of predictors carry the signal) and competes with stepwise selection while having dramatically better statistical properties.

7.19.2 Prerequisites

A working understanding of linear regression, ridge regression as the L2-penalised counterpart, cross-validation, and the bias-variance trade-off.

7.19.3 Theory

The lasso minimises

$$\|y - X\beta\|^2 + \lambda \sum_{j=1}^{p} |\beta_j|.$$

Larger λ produces more shrinkage and more zero coefficients. The L1 penalty's non-differentiability at zero is what produces sparse solutions: the optimisation finds corner points of the constraint region where some $\hat{\beta}_j = 0$ exactly. Tuning λ via k-fold cross-validation is standard; the package `glmnet` returns both the prediction-optimal $\lambda_{\min}$ and the more conservative $\lambda_{1\text{se}}$ within one standard error.

7.19.4 Assumptions

Standard linear-model assumptions on the residual structure; predictors should be standardised (`glmnet` does this internally and back-transforms). For statistical inference on selected coefficients, naive p-values are invalid because of the post-selection bias; tools like `selectiveInference` provide adjusted intervals.

7.19.5 R Implementation

```
library(glmnet)

X <- as.matrix(mtcars[, -1])
y <- mtcars$mpg
```

```r
set.seed(2026)
cvfit <- cv.glmnet(X, y, alpha = 1)   # alpha = 1 -> lasso

plot(cvfit)

# Selected predictors at lambda.1se
as.matrix(coef(cvfit, s = "lambda.1se"))
```

7.19.6 Output & Results

`cv.glmnet()` returns the cross-validation curve (mean squared error vs. log λ) plus the two recommended λ values. `plot(cvfit)` displays the CV curve with confidence intervals and vertical lines marking λ_{min} and λ_{1se}. The selected predictors are those with non-zero coefficients at the chosen λ.

7.19.7 Interpretation

A reporting sentence: "Lasso regression with cross-validation-selected λ_{1se} retained three of ten predictors (weight, horsepower, year); the remaining seven coefficients were shrunk to zero exactly. Cross-validated mean squared error was 6.3 mpg^2 at the chosen λ." Always report which λ rule was used; λ_{min} tends to over-fit, while λ_{1se} is more parsimonious.

7.19.8 Practical Tips

- The lasso is unstable when predictors are strongly correlated — it tends to pick one and zero the others arbitrarily; elastic net combines L1 and L2 penalties to mitigate this.
- Report the selected subset and the cross-validated fit, but never report naive p-values after lasso selection; they are invalid.
- For inference on selected coefficients, use `selectiveInference::fixedLassoInf()` for properly adjusted confidence intervals; alternatively, debias the lasso with `hdi::lasso.proj()`.
- For binary or count outcomes, switch the family: `cv.glmnet(X,y,family="binomial")` for logistic lasso, `family="poisson"` for Poisson lasso.
- The relaxed lasso (re-fitting OLS on the selected variables) often improves predictive performance; available via `glmnet`'s `relax=TRUE`.
- Always set the random seed before `cv.glmnet()`; the cross-validation folds affect the chosen λ and reproducibility requires a fixed seed.

7.19.9 R Packages Used

`glmnet` for the canonical lasso, ridge, and elastic-net implementations with cross-validated tuning; `selectiveInference` for post-selection confidence intervals; `hdi`

for high-dimensional inference; `caret` and `tidymodels` for unified cross-validation against alternative penalised methods.

7.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.19.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.20 Leverage and Influence

7.20.1 Introduction

Leverage and influence diagnostics identify observations whose removal would substantially change a regression fit. The two concepts are distinct: leverage measures how unusual an observation is in the predictor space, influence measures how much it actually affects the fitted coefficients. High leverage without a large residual is harmless; high residual without leverage produces a poor fit but not influential coefficients; only the combination of both produces an observation that disproportionately drives the regression.

7.20.2 Prerequisites

A working understanding of residual diagnostics, the hat matrix, and the leave-one-out perspective on regression.

7.20.3 Theory

Hat values h_{ii} are the diagonal of the projection matrix $H = X(X^\top X)^{-1}X^\top$. They measure each observation's leverage in the predictor space; the trace $\sum h_{ii} = p+1$ (number of parameters), so the average is $(p+1)/n$. Conventional threshold: $h_{ii} > 2(p+1)/n$ is high leverage.

DFBETA for coefficient j and observation i: standardised change in $\hat{\beta}_j$ when observation i is removed. Threshold: $|\text{DFBETA}_{ij}| > 2/\sqrt{n}$.

Cook's distance: combines residual and leverage into a single influence summary; thresholds $D_i > 4/n$ or $D_i > 1$.

DFFITS: standardised change in fitted value at observation i; analogous to DFBETA but for predictions rather than coefficients.

7.20.4 Assumptions

Classical OLS assumptions; the diagnostics above are linear-regression-specific. For GLMs, `influence.measures()` provides analogues based on one-step approximations.

7.20.5 R Implementation

```r
library(car)

fit <- lm(mpg ~ wt + hp, data = mtcars)

# Hat values
h <- hatvalues(fit)
which(h > 2 * (length(coef(fit))) / nrow(mtcars))

# DFBETAs
dfb <- dfbetas(fit)
which(apply(abs(dfb), 1, max) > 2 / sqrt(nrow(mtcars)))

# Cook's distance
cooks <- cooks.distance(fit)
which(cooks > 4 / nrow(mtcars))

# Influence plot
influencePlot(fit)
```

7.20.6 Output & Results

Each diagnostic flags candidate observations. `car::influencePlot()` produces a single integrated plot with leverage on the x-axis, studentised residual on the y-axis, and point size proportional to Cook's distance — the canonical three-way diagnostic.

7.20.7 Interpretation

A reporting sentence: "Influence diagnostics flagged three observations as candidates for further inspection (Maserati Bora: high leverage and large residual; Cadillac Fleetwood, Lincoln Continental: high leverage with moderate residuals); refitting without these observations changed the coefficient on weight by less than 0.3 SE units, with substantive conclusions unchanged." Sensitivity analyses are the right way to report influence diagnostics; pointing out flagged observations without consequence checks adds noise rather than signal.

7.20.8 Practical Tips

- Leverage is determined by the predictor matrix alone; outliers in the response are residual problems, not leverage.
- Only the combination of leverage and large residual produces influence; investigate flagged points before removing them.
- Never delete influential observations without investigating their cause — they may be the most informative data points or signal genuine measurement errors.
- Report sensitivity analyses with influential points removed; transparency about robustness builds reader confidence.
- For GLMs, `influence.measures(fit)` returns DFBETAs, hat values, and Cook's-distance equivalents based on one-step approximations.
- Robust regression (`MASS::rlm`, `robustbase::lmrob`) automatically downweights influential observations; useful when influential points are genuine and should be retained but should not dominate the fit.

7.20.9 R Packages Used

Base R `hatvalues()`, `dfbetas()`, `dffits()`, `cooks.distance()`, `influence.measures()`; `car::influencePlot()` for the three-way diagnostic visualisation; `performance::check_outliers()` for tidy diagnostics; `MASS::rlm()` and `robustbase::lmrob()` for robust regression alternatives.

7.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.20.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.21 Worked lme4::lmer Examples

7.21.1 Introduction

`lme4::lmer()` is the standard R implementation of linear mixed-effects models and the de facto reference for the formula syntax used across the broader mixed-models ecosystem (`lmerTest`, `glmmTMB`, `brms`, `nlme`). Mastering the random-effects syntax — what `(1|group)`, `(x|group)`, `(x||group)`, and nested vs. crossed grouping notation actually mean — is a prerequisite for fluent mixed-models work. The output structure is equally important: each block (fixed effects, random effects, residual variance) carries different inferential weight and must be reported correctly.

7.21.2 Prerequisites

A working understanding of mixed-effects models, the difference between fixed and random effects, and basic familiarity with `lme4` formula notation.

7.21.3 Theory

Core random-effects syntax in `lmer()`:

- `(1|g)`: random intercept per level of g.
- `(x|g)`: random intercept *and* correlated random slope for x within g.
- `(x||g)` or equivalently `(1|g)+(0+x|g)`: uncorrelated random intercept and random slope.
- `(1|g/h)`: h nested within g (each h value belongs to one g).
- `(1|g)+(1|h)`: g and h as crossed random factors.
- `(0+x|g)`: random slope without random intercept; rarely scientifically motivated.

The fixed-effects part of the formula (`Reaction~Days`) is interpreted as in `lm()`; the random-effects part adds the variance components.

7.21.4 Assumptions

Linearity in the linear predictor, conditional independence given random effects, multivariate Normal random effects, Normal residuals with constant variance, and independence of residuals from random effects.

7.21.5 R Implementation

```
library(lme4); library(lmerTest)

data(sleepstudy, package = "lme4")

# Random intercept only
m1 <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)

# Random intercept + correlated slope
m2 <- lmer(Reaction ~ Days + (Days | Subject), data = sleepstudy)

# Random intercept + uncorrelated slope
m3 <- lmer(Reaction ~ Days + (1 | Subject) + (0 + Days | Subject),
           data = sleepstudy)

# Nested example (sleepstudy doesn't have nesting; use simulated)
set.seed(2026)
dat <- data.frame(g = gl(5, 20), subj = factor(1:100),
                  x = rnorm(100), y = rnorm(100))
m_nest <- lmer(y ~ x + (1 | g/subj), data = dat)

anova(m1, m2, m3)
summary(m2)
```

7.21.6 Output & Results

`summary()` reports four blocks: fixed-effect coefficients with standard errors and (with `lmerTest`) Satterthwaite degrees of freedom and p-values; random-effect variance components and correlations between them; residual variance; and overall fit statistics including REML criterion. `anova()` between models compares them via likelihood-ratio tests when fitted with ML.

7.21.7 Interpretation

Reading `summary(m2)`:

- **Fixed effects**: population-level intercept and slope with standard errors and (with `lmerTest`) p-values.
- **Random effects**: per-subject intercept and slope variances, plus their correlation; the SD entries quantify between-subject heterogeneity.
- **Residual SD**: within-subject (after partialling out the random effects) error.
- **Number of obs / groups**: confirms the model is fitting the structure you intended.

A reporting sentence: "A random-intercept-and-slope model gave a population-level reaction-time slope of 10.5 ms per day of sleep deprivation (95 % CI 7.6 to 13.4) with substantial between-subject heterogeneity in slope (SD = 5.92, correlation with random intercept −0.04) and residual SD of 25.6 ms." Report variance components alongside fixed effects.

7.21.8 Practical Tips

- Use `lmerTest` for Satterthwaite or Kenward-Roger degrees of freedom; `lme4` alone returns no p-values for fixed effects.
- For convergence warnings, rescale predictors (`scale()` continuous variables) and check with `allFit()` across multiple optimisers; a stable estimate across optimisers is reassuring.
- Singular fits (correlation between random effects estimated as ± 1) usually mean the random structure is over-specified for the data; simplify by removing slope-intercept correlations or random slopes.
- For likelihood-ratio tests of variance components, the null is on the boundary; use the half-half χ^2 mixture or the parametric bootstrap (`pbkrtest::PBmodcomp`) rather than the standard χ^2.
- For non-Gaussian outcomes, `glmer()` extends the same formula syntax with `family=binomial`, `poisson`, etc.
- For Bayesian fits with identical formulas, `brms::brm()` is a near drop-in replacement and adds proper credible intervals.

7.21.9 R Packages Used

`lme4` for `lmer()` and `glmer()`; `lmerTest` for Satterthwaite / Kenward-Roger p-values; `pbkrtest` for parametric-bootstrap likelihood-ratio tests; `glmmTMB` for an alternative implementation that handles overdispersed and zero-inflated families; `brms` for Bayesian mixed models with the same formula syntax.

7.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.21.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.22 Logistic Regression Diagnostics

7.22.1 Introduction

Logistic regression diagnostics fall into three families that probe different aspects of model adequacy: calibration (do predicted probabilities match observed event rates across the probability range?), discrimination (how well does the model separate cases from controls?), and the functional form of continuous predictors (is linearity on the logit scale a reasonable assumption?). All three should be checked before reporting hazard ratios or recommending a clinical prediction model.

7.22.2 Prerequisites

A working understanding of logistic regression, the logit link, and the distinction between calibration and discrimination as separate aspects of predictive performance.

7.22.3 Theory

- **Calibration plot**: observed event proportion against predicted probability, typically binned into deciles; perfect calibration lies on the identity line. Deviations indicate over- or under-prediction at specific probability ranges.
- **Hosmer-Lemeshow test**: chi-squared comparison of observed vs. expected counts in deciles of predicted probability. Has known weaknesses (low power, bin-choice dependence) and should not be relied upon alone.
- **Deviance residuals**: approximately Normal under the model; plotted against the fitted logit, they reveal mean-structure misspecification.
- **DFBETA / Cook's distance**: influence diagnostics from `influence.measures()` flag observations that disproportionately affect coefficients.
- **Functional form** (linearity on the logit): component-plus-residual plots or `mgcv::gam` with smooth terms expose non-linearity in continuous predictors.

7.22.4 Assumptions

Standard logistic-regression assumptions: binary outcome, independent observations, linearity on the logit scale, and no severe multicollinearity.

7.22.5 R Implementation

```
library(ResourceSelection); library(performance)

d <- mtcars
d$am <- factor(d$am)
fit <- glm(am ~ wt + hp, data = d, family = binomial)

# Hosmer-Lemeshow
hoslem.test(as.numeric(d$am) - 1, fitted(fit))

# Calibration plot via performance
check_model(fit)

# Deviance residuals
resid(fit, type = "deviance") |> plot()
```

7.22.6 Output & Results

`hoslem.test()` returns the chi-squared statistic and p-value. `performance::check_model()` produces a multi-panel diagnostic figure including calibration, posterior predictive checks, residual plots, and influence diagnostics. Deviance residuals plotted against fitted values reveal mean-structure problems.

7.22.7 Interpretation

A reporting sentence: "Calibration plots showed observed and expected event probabilities aligned across deciles, with the Hosmer-Lemeshow test not rejecting adequate calibration ($\chi^2_8 = 5.4$, $p = 0.72$); deviance residuals were approximately Normal with no systematic pattern against fitted logit, supporting the linearity assumption for both continuous predictors." Always pair Hosmer-Lemeshow with a calibration plot; the test alone is unreliable.

7.22.8 Practical Tips

- Hosmer-Lemeshow has well-known weaknesses (low power, sensitive to bin choice); prefer calibration plots and the Brier score for primary calibration assessment.
- For non-linearity in continuous predictors, add restricted cubic splines (`rms::rcs(x,4)`) or fit a `mgcv::gam` with `s(x)` and compare AICs.
- Outliers in logistic regression are jointly driven by leverage and residual; use `influence.measures(fit)` and `dfbetas(fit)` to identify candidate points and run sensitivity analyses with them removed.
- Report both discrimination (AUC) and calibration; they are independent aspects of model quality and a model can be excellent on one and poor on the other.
- For small samples, the Hosmer-Lemeshow test can fail to reject simply because of low power; visual calibration is more informative.
- The `rms::val.prob()` function gives a comprehensive calibration summary including intercept, slope, and bias-corrected $E_{\max}$.

7.22.9 R Packages Used

`ResourceSelection::hoslem.test()` for the Hosmer-Lemeshow test; `performance::check_model()` for an integrated multi-panel diagnostic; `pROC` for ROC and AUC; `rms::val.prob()` for calibration intercept and slope; `mgcv::gam()` for non-linearity testing via smooth terms.

7.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.22.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.23 Logistic Regression

7.23.1 Introduction

Logistic regression models binary outcomes via the logit link, mapping the probability of "event" to the entire real line so that linear regression machinery applies on the log-odds scale. It is the most widely used GLM in clinical, epidemiological, and behavioural research, producing odds ratios that directly answer the question "how does the odds of the outcome change with this covariate?" The fit is by iteratively reweighted least squares (IRLS), maximising the binomial likelihood.

7.23.2 Prerequisites

A working understanding of the binomial distribution, maximum likelihood estimation, the logit link, and the distinction between odds ratios and relative risks.

7.23.3 Theory

The model is

$$Y \sim \text{Bernoulli}(p), \qquad \text{logit}(p) = \log \frac{p}{1-p} = \beta_0 + \sum_j \beta_j X_j.$$

IRLS produces $\hat{\beta}$ on the log-odds scale; exponentiating gives odds ratios, which are interpretable as the multiplicative change in the odds of the outcome per unit increase in the covariate, all else equal. For rare events (prevalence below ~10 %), the odds ratio approximates the relative risk; for common events the two diverge and odds ratios overstate the relative risk.

7.23.4 Assumptions

Binary outcome, independent observations, linearity on the logit scale (the log-odds is a linear function of the covariates), absence of severe multicollinearity, and adequate events per variable (EPV $\geq$ 10 as a working rule for stable estimation).

7.23.5 R Implementation

```r
library(broom); library(pROC); library(performance)

d <- mtcars
d$am <- factor(d$am)

fit <- glm(am ~ wt + hp, data = d, family = binomial)
tidy(fit, conf.int = TRUE, exponentiate = TRUE)
glance(fit)

# Discrimination
pred <- predict(fit, type = "response")
roc(d$am, pred, quiet = TRUE) |> auc()

check_model(fit)
```

7.23.6 Output & Results

`tidy(fit,exponentiate=TRUE)` returns odds ratios with confidence intervals; `glance()` provides AIC, deviance, and pseudo-R^2 summaries. The AUC quantifies discrimination on the same data; for honest assessment, cross-validate or hold out a test set.

7.23.7 Interpretation

A reporting sentence: “A multivariable logistic regression of transmission type on weight and horsepower estimated that each additional 1000 lbs of weight

reduced the odds of manual transmission by a factor of 0.10 (95 % CI 0.02 to 0.43, $p = 0.003$), adjusted for horsepower; the model's in-sample AUC was 0.93 (95 % CI 0.84 to 1.00)." Always report odds ratios, not raw coefficients on the log-odds scale.

7.23.8 Practical Tips

- Report odds ratios with confidence intervals; coefficients on the log-odds scale are rarely the most communicative summary for clinical readers.
- For rare events (prevalence < 10 %), OR $\approx$ RR; for common events, OR overstates RR. Switch to `logbin` or modified Poisson for direct relative-risk regression when needed.
- Small samples can produce complete separation, where MLE drifts to $\pm\infty$; use Firth correction (`logistf::logistf`) or weakly informative Bayesian priors (`brms`) for finite estimates.
- Calibration matters as much as discrimination; assess with `performance::check_model()` calibration plots and the Hosmer-Lemeshow test (with caveats about its known limitations).
- For matched case-control data, use conditional logistic regression (`survival::clogit`) which preserves the matched-pair structure.
- For high-dimensional or correlated predictors, switch to penalised logistic regression via `glmnet(family="binomial")`.

7.23.9 R Packages Used

Base R `glm()` with `family=binomial`; `broom::tidy()` and `glance()` for tidy summaries; `pROC` for ROC and AUC; `performance::check_model()` for diagnostic plots; `logistf` for Firth-penalised logistic regression; `glmnet` for elastic-net logistic regression; `survival::clogit` for matched designs.

7.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.23.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)

- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.24 Mixed-Effects Models: Introduction

7.24.1 Introduction

Mixed-effects models (also called multilevel or hierarchical models) handle data with a clustered or nested structure: patients within hospitals, repeated measures within subjects, plants within plots. The model has **fixed effects** for population-level structure and **random effects** capturing cluster-specific deviations.

7.24.2 Prerequisites

Linear regression.

7.24.3 Theory

Simplest form: random intercept model

$$Y_{ij} = \beta_0 + \beta_1 X_{ij} + u_i + \varepsilon_{ij}, \quad u_i \sim \mathcal{N}(0, \sigma_u^2),\ \varepsilon \sim \mathcal{N}(0, \sigma^2).$$

u_i are subject-specific deviations; ε_{ij} are residuals. Intraclass correlation: $\rho = \sigma_u^2/(\sigma_u^2 + \sigma^2)$.

Extensions: random slopes (u_{1i}), nested or crossed groupings, non-normal responses (GLMM).

7.24.4 Assumptions

- Clusters are a random sample from a population of clusters.
- Random effects normal; residuals normal.
- Linearity, independence across clusters.

7.24.5 R Implementation

```r
library(lme4); library(lmerTest)

data(sleepstudy, package = "lme4")
head(sleepstudy)

# Random intercept
fit_ri <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)
summary(fit_ri)

# Random intercept + slope
fit_rs <- lmer(Reaction ~ Days + (Days | Subject), data = sleepstudy)
summary(fit_rs)

# Test random slope vs. intercept only
anova(fit_ri, fit_rs)
```

7.24.6 Output & Results

Fixed-effect table with estimates, SEs, df (approximated), p-values; random-effect variance / SD summary; likelihood-ratio test between nested random-effects structures.

7.24.7 Interpretation

"Each day of sleep deprivation increased reaction time by 10.5 ms (SE = 1.5, $p < 0.001$). Between-subject variation in intercept was substantial (SD = 37 ms), and random-slope comparison showed better fit for a model allowing slope variation (chi-squared = 42, $p < 0.001$)."

7.24.8 Practical Tips

- Specify random effects by the grouping factor.
- Use REML by default; switch to ML for likelihood-ratio tests between fixed-effect specifications.
- Singular fits often arise from over-parameterised random-effects structures; simplify.
- For p-values, use `lmerTest` (Kenward-Roger or Satterthwaite approximations) – standard `lme4` does not provide them.
- For hierarchical binary or count outcomes, use `glmer` or `glmmTMB`.

7.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.24.10 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.25 Model Selection with AIC and BIC

7.25.1 Introduction

AIC (Akaike information criterion) and BIC (Bayesian information criterion) are penalised likelihood scores for comparing models on the same data. Both reward fit (high likelihood) and penalise complexity (number of parameters). They differ in how aggressively they penalise: BIC's penalty grows with sample size, so BIC favours simpler models in large samples while AIC retains a constant per-parameter cost. Both are essential tools in any model-selection workflow that goes beyond likelihood-ratio tests.

7.25.2 Prerequisites

A working understanding of maximum likelihood, the log-likelihood as a measure of fit, and the bias-variance trade-off in model complexity.

7.25.3 Theory

$$\mathrm{AIC} = -2\log L + 2k, \qquad \mathrm{BIC} = -2\log L + k\log n,$$

with k the number of estimated parameters and n the sample size. Lower is better. Conventional thresholds for AIC differences (ΔAIC): < 2 inconclusive, 4–7 moderate preference, > 10 decisive. Relative model weights $\exp(-\Delta\text{AIC}/2)$ give the relative likelihood of each model among those compared.

AIC is asymptotically equivalent to leave-one-out cross-validation under regularity conditions; it targets predictive accuracy. BIC approximates the posterior model probability under a specific prior; it targets identification of the true model when one is in the candidate set. BIC's $\log n$ penalty makes it consistent (it picks the true model as $n \to \infty$ if the true model is among candidates) but conservative in finite samples.

7.25.4 Assumptions

Models compared must be fitted on the same data, with the same response (no transformation differences). For nested models, both AIC and likelihood-ratio tests apply; for non-nested, only AIC and BIC make sense.

7.25.5 R Implementation

```r
fit1 <- lm(mpg ~ wt, data = mtcars)
fit2 <- lm(mpg ~ wt + hp, data = mtcars)
fit3 <- lm(mpg ~ wt + hp + cyl + disp, data = mtcars)

AIC(fit1, fit2, fit3)
BIC(fit1, fit2, fit3)

# Relative likelihood of a model
delta_aic <- AIC(fit1, fit2, fit3)$AIC - min(AIC(fit1, fit2, fit3)$AIC)
exp(-delta_aic / 2)     # relative likelihood
```

7.25.6 Output & Results

`AIC()` and `BIC()` return tabular comparisons with degrees of freedom and the criterion value. The relative likelihoods, computed as $\exp(-\Delta\text{AIC}/2)$, give the proportional support each model has from the data.

7.25.7 Interpretation

A reporting sentence: "Model 2 (mpg ~ wt + hp) had the lowest AIC at 164.0; Model 1 (wt only) differed by $\Delta\text{AIC} = 8.4$, decisive evidence against Model 1, while Model 3 (with cyl and disp added) had $\Delta\text{AIC} = 1.2$, indicating no meaningful improvement over Model 2 despite the additional predictors. BIC also selected Model 2 ($\Delta\text{BIC} = 6.1$ vs Model 1; Model 3 worse by 5.5)." Always quote Δ values rather than absolute AICs; only differences are interpretable across model sets.

7.25.8 Practical Tips

- AIC and BIC can disagree; AIC favours larger models, BIC smaller. Choose by question: AIC for prediction, BIC for parsimony or near-true-model selection.
- Compare only models fitted on the same data with the same response scale; AICs across log-transformed and raw outcomes are not comparable.
- For nested models, likelihood-ratio tests and AIC both work; for non-nested models, AIC and BIC are the only options.
- Model averaging (`MuMIn::dredge` and `model.avg`) often outperforms single-model selection for prediction; the resulting predictions average across high-AIC-weight models.
- AICc (corrected AIC) adds a finite-sample correction and is preferable for small n relative to k; the formula is $\text{AICc} = \text{AIC} + 2k(k+1)/(n-k-1)$.
- Reporting one fit's AIC alone is uninformative; always quote the comparison set.

7.25.9 R Packages Used

Base R `AIC()` and `BIC()`; `MuMIn::dredge()` and `model.avg()` for exhaustive subset selection and model averaging; `AICcmodavg::AICc()` for small-sample corrections; `performance::compare_performance()` for tidy multi-criterion comparison; `lmtest::lrtest()` for likelihood-ratio tests on nested models.

7.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.25.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)

- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.26 Multicollinearity and VIF

7.26.1 Introduction

Multicollinearity describes the situation in which predictors in a regression are strongly linearly related to each other. Coefficient estimates remain unbiased under multicollinearity, but their standard errors inflate, making individual effects hard to attribute reliably and producing the classic "the model is significant overall but no individual coefficient is" pattern. Diagnosing collinearity, deciding whether it matters for the inferential question, and choosing an appropriate remedy are routine parts of every multivariable regression workflow.

7.26.2 Prerequisites

A working understanding of multiple linear regression, the geometric interpretation of regression coefficients, and the bias-variance trade-off.

7.26.3 Theory

The variance inflation factor for predictor j is

$$\mathrm{VIF}_j = \frac{1}{1 - R_j^2},$$

where R_j^2 comes from regressing X_j on all other predictors. $\mathrm{VIF}_j \geq 5$ conventionally flags concern; ≥ 10 indicates severe collinearity. The square root of VIF is the multiplier on the coefficient's standard error: a VIF of 16 inflates the SE by a factor of 4 relative to an uncorrelated predictor.

The condition number κ of the (scaled) design matrix is the ratio of the largest to smallest singular value; $\kappa > 30$ signals strong collinearity. Pairwise correlations expose specific offending pairs but miss multivariate collinearity that VIF and the condition number catch.

Remedies include dropping one of the collinear predictors, combining them (sum, average, principal component), and shifting to ridge or elastic-net regression that penalises the coefficient norm and stabilises estimates under collinearity.

7.26.4 Assumptions

Standard multiple-regression setup; VIF is defined for the same predictor set used in the focal regression.

7.26.5 R Implementation

```r
library(car)

fit <- lm(mpg ~ wt + hp + cyl + disp + drat, data = mtcars)

vif(fit)

# Condition number
X <- model.matrix(fit)[, -1]
kappa(X)

# Correlation matrix of predictors
cor(mtcars[, c("wt", "hp", "cyl", "disp", "drat")])
```

7.26.6 Output & Results

`vif()` returns one VIF per predictor; values above 10 signal severe collinearity. `kappa()` returns the condition number on the scaled design matrix; pairwise correlations identify which specific predictors are colinear with each other.

7.26.7 Interpretation

A reporting sentence: "Cylinder count and displacement had VIFs of 12.4 and 14.8 respectively, with a Pearson correlation of $r = 0.90$; we retained cylinders for biological interpretability and dropped displacement, reducing the maximum VIF to 4.5 and tightening the SE on the retained predictors by approximately 30 %." Always report the action taken in response to high VIF.

7.26.8 Practical Tips

- High VIF is not always a problem: if prediction is the goal and individual coefficient interpretation is not, leaving collinear predictors in is acceptable and can even improve prediction.
- Centring predictors before adding interactions reduces the mechanical collinearity between main and interaction terms; do this routinely.
- Ridge and elastic-net regression handle collinearity gracefully through the L2 penalty; they shrink coefficients on collinear predictors proportionally rather than picking one and inflating its SE.
- Check VIF after each model change; adding or removing predictors can change the collinearity structure dramatically.

- For polynomial or spline terms, use orthogonal polynomials (`poly(x,degree)`) which avoid the inherent collinearity between $x, x^2, x^3, ...$
- Report VIF in the model diagnostics whenever multicollinearity is plausible; reviewers often request it.

7.26.9 R Packages Used

`car::vif()` for variance-inflation factors; base R `kappa()` for the condition number; `performance::check_collinearity()` for tidy collinearity diagnostics; `glmnet` for ridge and elastic-net alternatives that handle collinearity through regularisation.

7.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.26.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.27 Multinomial Logistic Regression

7.27.1 Introduction

Multinomial logistic regression extends binary logistic regression to outcomes with three or more unordered categories. Where binary logistic models the log-odds of one category vs another, multinomial models the log-relative-risk-ratios of each non-reference category against a chosen reference, simultaneously across all $k-1$ contrasts. It is the standard tool for predicting voting choice, transportation mode, disease subtype, or any categorical outcome whose categories have no inherent ordering.

7.27.2 Prerequisites

A working understanding of binary logistic regression, the softmax function, and the relative-risk-ratio interpretation of exponentiated coefficients.

7.27.3 Theory

For outcome $Y \in \{1, \dots, k\}$ with category 1 as reference:

$$\log \frac{\mathrm{P}(Y=j)}{\mathrm{P}(Y=1)} = \beta_0^{(j)} + \sum_l \beta_l^{(j)} X_l, \qquad j = 2, \dots, k.$$

The model fits $k-1$ linear equations simultaneously by maximum likelihood. Exponentiating $\beta_l^{(j)}$ gives the relative-risk ratio: the multiplicative change in $\mathrm{P}(Y=j)/\mathrm{P}(Y=1)$ per unit increase in X_l, holding other predictors fixed.

The independence-of-irrelevant-alternatives (IIA) assumption — that the ratio of probabilities between any two categories does not depend on which other categories are available — is the key restriction. When IIA fails, nested or mixed logit alternatives generalise the model.

7.27.4 Assumptions

Unordered categorical outcome (use ordinal logistic for ordered outcomes), independent observations, IIA, and a sufficiently large sample to estimate $k-1$ sets of coefficients (rule of thumb: ≥ 10 events in the smallest category per coefficient).

7.27.5 R Implementation

```
library(nnet); library(broom); library(gtsummary)

data(iris)
fit <- multinom(Species ~ Sepal.Length + Sepal.Width, data = iris, trace
 ↪  = FALSE)
```

```r
# Coefficients (log-RRR)
summary(fit)

# Exponentiated as relative-risk ratios
tidy(fit, exponentiate = TRUE, conf.int = TRUE)

# Predicted probabilities
head(predict(fit, type = "probs"))

# Publication-ready table
tbl_regression(fit, exponentiate = TRUE)
```

7.27.6 Output & Results

`summary(fit)` returns coefficient tables per non-reference category; `tidy(fit,exponentiate=TRUE)` provides relative-risk ratios with confidence intervals. `predict(...,type="probs")` returns predicted category probabilities for new observations.

7.27.7 Interpretation

A reporting sentence: "Compared to setosa (reference), each 1-cm increase in sepal length multiplied the relative risk of versicolor over setosa by 10.8 (95 % CI 3.7 to 31.2, $p < 0.001$); the corresponding RRR for virginica was 9.0 (95 % CI 3.1 to 26.0, $p < 0.001$). The reference category (setosa) was chosen because it is one of the three iris species and serves as the baseline group." Always state the reference category and the assumed IIA.

7.27.8 Practical Tips

- Test IIA with the Hausman-McFadden test (`mlogit::hmftest`); if violated, switch to nested or mixed logit (`mlogit`).
- Choose the reference category for the most interpretable contrasts — typically the most common category or a substantively meaningful baseline.
- `nnet::multinom()` uses BFGS optimisation; for very large problems with many categories or predictors, `mclogit` or `mlogit` scale better.
- For ordered outcomes, ordinal logistic regression is more efficient than multinomial; check ordering of levels before defaulting to multinomial.
- Communicate results via predicted probability plots (`ggeffects::ggpredict`); readers absorb absolute probabilities better than relative-risk ratios.
- Variable selection in multinomial models is awkward because each predictor produces $k - 1$ coefficients; consider penalised multinomial logistic (`glmnet(family="multinomial")`).

7.27.9 R Packages Used

`nnet::multinom()` for the canonical implementation; `mlogit` for choice models with rich panel and nested-logit extensions; `glmnet(family="multinomial")` for penalised multinomial logistic; `broom::tidy()` for tidy output; `gtsummary::tbl_regression()` for publication-ready tables; `ggeffects` for predicted probability plots.

7.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.27.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.28 Multiple Linear Regression

7.28.1 Introduction

Multiple linear regression extends simple regression to include two or more predictors. Each coefficient represents the partial effect of its predictor, holding all others constant. This allows confounding adjustment, multi-factor prediction, and partial correlations.

7.28.2 Prerequisites

Simple linear regression.

7.28.3 Theory

Model: $Y = \beta_0 + \beta_1 X_1 + ... + \beta_p X_p + \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

OLS estimates: $\hat{\beta} = (X^\top X)^{-1} X^\top Y$. Each coefficient is the **partial slope**: change in $E[Y]$ per unit of X_j, holding other predictors fixed.

Collinearity: highly correlated predictors inflate SE of individual coefficients. Diagnose with variance inflation factors (VIF): $\text{VIF}_j = 1/(1 - R_j^2)$, where R_j^2 is the R^2 of regressing X_j on the others. VIF > 5 is concerning; > 10 is severe.

R^2 is the proportion of variance explained; adjusted R^2 penalises model complexity.

7.28.4 Assumptions

Linearity of each predictor, independence, homoscedasticity, normality of residuals, no severe collinearity.

7.28.5 R Implementation

```r
library(broom); library(car); library(performance)

data(mtcars)
fit <- lm(mpg ~ wt + hp + cyl + disp, data = mtcars)

tidy(fit, conf.int = TRUE)
glance(fit)
vif(fit)

check_model(fit)
```

7.28.6 Output & Results

Coefficients with SE, t-statistic, p-values, 95 % CIs. R^2, adjusted R^2, residual SE. VIF values per predictor.

7.28.7 Interpretation

"After adjusting for horsepower, cylinders, and displacement, each additional 1000 lbs of weight reduced mpg by 3.8 (95 % CI -6.8 to -0.7). The model explained 84 % of the variance in mpg ($R^2 = 0.84$)."

7.28.8 Practical Tips

- Centre continuous predictors for interpretable intercepts.
- For highly collinear predictors, drop one, combine them, or use ridge regression.
- Always check residual plots before trusting p-values.
- Report both raw coefficients and standardised ones when comparing effect magnitudes across predictors on different scales.
- For prediction accuracy, cross-validate – in-sample R^2 is optimistic.

7.28.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.28.10 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.29 Negative Binomial Regression

7.29.1 Introduction

Negative binomial (NB) regression is the standard remedy for overdispersed count data — counts whose variance exceeds the mean, which is most real-world count data once any unobserved heterogeneity is present. It generalises Poisson regression by adding a dispersion parameter θ that explicitly models the

additional variance, so confidence intervals and tests respect the true uncertainty rather than the artificially narrow Poisson ones. For epidemiological count data, hospital admissions, ecological abundance counts, and almost any context where Poisson dispersion checks fail, NB is the natural next step.

7.29.2 Prerequisites

A working understanding of Poisson regression, the log link, and the concept of overdispersion as the failure of $\mathrm{Var}(Y) = \mathbb{E}[Y]$.

7.29.3 Theory

The NB distribution arises as a Poisson-Gamma mixture: $Y \mid \lambda \sim \mathrm{Poisson}(\lambda)$ with $\lambda \sim \mathrm{Gamma}(\theta, \theta/\mu)$. Marginalising over the Gamma yields $Y \sim \mathrm{NegBin}(\mu, \theta)$ with mean μ and variance

$$\mathrm{Var}(Y) = \mu + \mu^2/\theta.$$

The dispersion parameter θ controls extra-Poisson variance; as $\theta \to \infty$, NB reduces to Poisson, and as $\theta \to 0$, the variance grows quadratically with the mean.

The regression model uses the log link $\log \mu = X^\top \beta$. Coefficients are interpreted as log-rate ratios identically to Poisson; the difference is that standard errors and inference reflect the genuine over-Poisson variance.

7.29.4 Assumptions

Non-negative integer outcomes, independent observations conditional on covariates, and the NB variance structure $\mu + \mu^2/\theta$. For excess zeros beyond what NB predicts, zero-inflated extensions are needed.

7.29.5 R Implementation

```r
library(MASS)

set.seed(2026)
d <- data.frame(x = rnorm(300))
d$y <- rnbinom(300, mu = exp(1 + 0.5 * d$x), size = 2)

fit <- glm.nb(y ~ x, data = d)
summary(fit)
exp(coef(fit))   # rate ratios
fit$theta
```

7.29.6 Output & Results

`glm.nb()` returns regression coefficients on the log scale (Wald z-tests, confidence intervals) plus the estimated dispersion parameter $\hat{\theta}$. Exponentiating coefficients gives rate ratios; comparing AIC against a Poisson fit on the same data confirms whether NB is justified.

7.29.7 Interpretation

A reporting sentence: "Negative binomial regression with estimated dispersion $\hat{\theta} = 2.1$ (indicating substantial overdispersion relative to Poisson, $\Delta\text{AIC} = 87$) estimated a rate ratio of 1.65 per unit increase in x (95 % CI 1.40 to 1.95, $p < 0.001$); a Poisson fit underestimated the standard error by 22 %, illustrating the consequence of ignoring overdispersion." Always report $\hat{\theta}$ alongside the rate ratios; it carries information about residual heterogeneity.

7.29.8 Practical Tips

- Compare NB and Poisson by AIC or likelihood-ratio test (`lmtest::lrtest`); the NB additional parameter θ has well-defined inference.
- For zero-inflation in addition to overdispersion, use `pscl::zeroinfl(..., dist="negbin")` or `glmmTMB(family=nbinom2(),ziformula=...)`.
- For count data with hierarchical structure, `glmmTMB(family=nbinom2())` adds random effects on top of the NB likelihood; `lme4::glmer.nb()` is the older, less reliable alternative.
- Very large $\hat{\theta}$ (> 100) suggests Poisson would suffice and the NB extension is unnecessary; very small values (< 1) indicate severe overdispersion that may signal mis-specification rather than just Poisson failure.
- The NB1 parameterisation (`glmmTMB(family=nbinom1())`) gives variance proportional to the mean (linear); NB2 (the default in `MASS::glm.nb`) gives quadratic variance. Choose based on the variance-mean relationship in residual plots.
- Pearson residuals from the NB fit should look approximately uniform; systematic patterns reveal model misspecification.

7.29.9 R Packages Used

`MASS::glm.nb()` for canonical NB regression with offset support; `glmmTMB` for mixed-effects NB1 / NB2 / zero-inflated models with cleaner overdispersion handling; `pscl::zeroinfl()` for zero-inflated count models; `lmtest::lrtest()` for likelihood-ratio comparisons against Poisson; `DHARMa` for residual diagnostics across count-model families.

7.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.29.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.30 Nested vs Crossed Random Effects

7.30.1 Introduction

Mixed-effects models require careful specification of the random-effects structure, and the most common confusion is between nested and crossed designs. Nested means one grouping factor's levels sit strictly within another's: patients within hospitals, schools within districts, plots within fields. Crossed means levels of one factor appear across levels of another: subjects each rate the same items, items rated by multiple subjects, students each take the same set of exams. Misspecifying one as the other produces wrong variance components and biased standard errors.

7.30.2 Prerequisites

A working understanding of mixed-effects models, random intercepts, and the geometric meaning of "grouping factor" in a hierarchical or crossed design.

7.30.3 Theory

Nested random effects: each level of the inner factor belongs to exactly one level of the outer factor. Patient 17 in hospital A is a different conceptual

entity from patient 17 in hospital B even if the IDs coincide. `lme4` syntax: `(1|outer/inner)` or, equivalently, `(1|outer)+(1|outer:inner)`. The forward-slash shorthand expands automatically; explicit interaction notation removes ambiguity for readers.

Crossed random effects: each level of one factor combines with multiple levels of the other. Subjects rate items, items are rated by subjects. `lme4` syntax: `(1|subject)+(1|item)`. The variance components for the two grouping factors are estimated independently, and there is no implied hierarchy.

A practical heuristic: if you can ask "which A does this B belong to?" and get a unique answer, the design is nested. If Bs are reused across As, the design is crossed.

7.30.4 Assumptions

The grouping structure is correctly identified; random effects within each grouping factor are independent and Normal; for crossed designs, the two grouping factors are conditionally independent given the model.

7.30.5 R Implementation

```r
library(lme4)

# Nested example (simulated)
set.seed(2026)
hosp <- gl(5, 20)           # 5 hospitals
pt   <- factor(seq_along(hosp))   # 100 unique patients
y    <- rnorm(100) + rep(rnorm(5, 0, 3), each = 20)

fit_nested <- lmer(y ~ 1 + (1 | hosp))
# Crossed
item <- gl(10, 10, length = 100)
y2 <- rnorm(100) + rep(rnorm(10, 0, 2), length.out = 100) +
                   rep(rnorm(10, 0, 1), length.out = 100)
fit_crossed <- lmer(y2 ~ 1 + (1 | hosp) + (1 | item))
summary(fit_crossed)
```

7.30.6 Output & Results

`summary(fit)` reports variance components for each random effect. For nested designs the variance is decomposed by hierarchical level; for crossed designs the two grouping factors contribute independently. Likelihood-ratio tests against simpler models confirm whether each random effect is needed.

7.30.7 Interpretation

A reporting sentence: "Crossed random effects for subject and item revealed substantial subject-level variation ($\hat{\sigma}_{\text{subject}} = 2.1$) and smaller item-level variation ($\hat{\sigma}_{\text{item}} = 1.0$); the residual SD was 1.5. The two grouping factors are conceptually independent — every subject saw every item — so the structure is correctly specified as crossed rather than nested." Always describe the grouping structure explicitly in methods.

7.30.8 Practical Tips

- Nested IDs must be unique within each parent level; if patient IDs repeat across hospitals (e.g. patient 17 in hospital A and a different patient 17 in hospital B), use the explicit `hospital:patient` interaction to disambiguate.
- Crossed designs with small per-factor cluster counts (< 5 levels) can struggle; verify convergence and watch for singular fits.
- `(1|A/B)` shorthand is convenient but can be opaque; documenting the structure as `(1|A)+(1|A:B)` explicitly removes ambiguity.
- `glmmTMB` handles many complex random structures (crossed + non-Normal outcomes, structured covariance) more gracefully than `lme4`.
- For longitudinal data with crossed item / occasion structure, include both random factors plus any temporal autocorrelation as a within-subject correlation term.
- A common diagnosis tool: `xtabs(~A+B,data=d)` shows the crossing pattern directly. A diagonal-block table indicates nesting; a dense table indicates crossing.

7.30.9 R Packages Used

`lme4` for `lmer()` and `glmer()` with nested and crossed random effects; `lmerTest` for p-values via Satterthwaite or Kenward-Roger; `glmmTMB` for complex random structures and non-Normal outcomes; `pbkrtest` for parametric-bootstrap LRTs on variance components.

7.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.30.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression

- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.31 Offsets in Regression

7.31.1 Introduction

In count regression, an offset is a predictor whose coefficient is fixed at 1. It is the standard mechanism for modelling rates per unit exposure: events per person-year of follow-up, infections per 1000 hospital admissions, mutations per kilobase of DNA. Without an offset, the model fits expected counts and treats exposure as a confounder; with an offset, the model fits expected rates and exposure is correctly accounted for. Forgetting the offset is a leading cause of biased coefficient estimates in epidemiological count-data work.

7.31.2 Prerequisites

A working understanding of Poisson and negative-binomial regression, the log link, and the distinction between counts and rates as inferential targets.

7.31.3 Theory

For events Y_i observed over exposure E_i, modelling the rate Y_i/E_i via Poisson regression is equivalent to modelling the count Y_i with $\log E_i$ as a known offset:

$$\log \mathbb{E}[Y_i] = \log E_i + X_i^\top \beta \quad \Leftrightarrow \quad \log \mathbb{E}[Y_i/E_i] = X_i^\top \beta.$$

The offset $\log E_i$ enters the linear predictor with a fixed coefficient of 1, so its contribution is treated as known rather than estimated. The remaining coefficients β are interpretable as log-rate ratios; exponentiating gives rate ratios per unit covariate change.

The same construction extends to negative-binomial regression (with overdispersion), to logistic regression with `offset=log(odds_baseline)` for known baseline rates, and to Cox regression where person-time at risk plays an analogous role.

7.31.4 Assumptions

Exposure $E_i > 0$ for every observation; the rate is proportional to exposure (a doubling of exposure doubles the expected count, all else equal); the count distribution (Poisson, NB) is appropriate for the data.

7.31.5 R Implementation

```
set.seed(2026)
d <- data.frame(
  x = rnorm(200),
  person_years = runif(200, 0.5, 5)
)
d$events <- rpois(200, lambda = d$person_years * exp(-2 + 0.5 * d$x))

# With offset
fit <- glm(events ~ x, data = d, family = poisson,
           offset = log(person_years))
summary(fit)
exp(coef(fit))      # rate ratios per unit x

# WITHOUT offset (wrong!): models expected count ignoring exposure
fit_no_off <- glm(events ~ x, data = d, family = poisson)
summary(fit_no_off)
```

7.31.6 Output & Results

With the offset, the slope coefficient is the log-rate ratio per unit increase in x; exponentiating gives the rate ratio (close to the data-generating value of $\exp(0.5) = 1.65$). Without the offset, coefficients absorb the variation in exposure and are biased; the comparison illustrates the importance of correctly specifying the rate model.

7.31.7 Interpretation

A reporting sentence: "After accounting for variable follow-up via an offset of log(person-years), each unit increase in x was associated with a 65 % higher event rate per person-year (rate ratio 1.65, 95 % CI 1.32 to 2.05, $p < 0.001$); omitting the offset gives a misleading 'count ratio' of 1.43 that conflates the rate effect with average follow-up." Always state the exposure unit (person-years, 1000 admissions, kilobases) when reporting rate ratios.

7.31.8 Practical Tips

- The offset is log(exposure), not exposure itself; using exposure directly is a common bug.
- The `offset=log(exposure)` argument in `glm()` keeps the coefficient fixed at 1; including `log(exposure)` as a regular predictor estimates the coefficient and produces a different model.
- Offsets generalise to negative-binomial, zero-inflated, and survival models; check the family-specific syntax (`glmmTMB`, `MASS::glm.nb`, `survival::coxph`).
- For binomial regression, the analogous construction uses `offset=log(p_baseline/(1-p_baseline))` when a known baseline is to be respected.
- Always inspect whether exposure varies meaningfully across observations before deciding to include or omit an offset; ignoring a varying exposure biases all coefficient estimates.
- Report rate ratios after exponentiating; coefficient on the log scale is rarely the most communicative summary for clinical readers.

7.31.9 R Packages Used

Base R `glm()` with `offset=` and `family=poisson`; `MASS::glm.nb()` for negative-binomial regression with offsets; `glmmTMB` for mixed-effects count models with offsets and overdispersion; `survival::coxph()` for survival analyses where exposure enters as time at risk.

7.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.31.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)

- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.32 Ordinal Logistic Regression

7.32.1 Introduction

Ordinal logistic regression — most commonly the proportional-odds model — handles ordered categorical outcomes such as Likert-scale responses, disease severity grades, NYHA functional class, or cancer stage. It exploits the ordering via a cumulative-probability model: the cumulative odds of being in or below each category share the same set of regression coefficients across all cut-points. Multinomial logistic ignores the ordering and estimates separate coefficients for each non-reference category, which is wasteful when ordering is real and informative.

7.32.2 Prerequisites

A working understanding of binary logistic regression, ordered categorical variables, and the cumulative-probability framing of ordinal data.

7.32.3 Theory

The proportional-odds model is

$$\log \frac{\mathrm{P}(Y \le j)}{\mathrm{P}(Y > j)} = \alpha_j - X^\top \beta, \qquad j = 1, \dots, k-1,$$

with k ordered categories. The cut-points α_j shift with j, but the regression coefficients β are constrained to be identical across all cut-points — the proportional-odds assumption. Exponentiated coefficients are cumulative odds ratios: per unit increase in X, the odds of being in a higher category multiplied by $\exp(\beta)$, regardless of which threshold is being crossed.

The proportional-odds assumption is testable with Brant's test or with a likelihood-ratio test against a more flexible partial-proportional-odds or generalised ordered logit model.

7.32.4 Assumptions

Ordered categorical outcome with at least three levels, proportional odds across all cut-points, independent observations, and adequate counts in each category

(rule of thumb: at least 5–10 events per coefficient in the smallest category).

7.32.5 R Implementation

```r
library(MASS); library(brant)

d <- data.frame(
  y = factor(sample(c("low", "med", "high"), 200, replace = TRUE),
             levels = c("low", "med", "high"), ordered = TRUE),
  x = rnorm(200)
)
d$y <- factor(as.character(d$y)[sample(1:200,
                                       prob = 1 + plogis(d$x), replace =
                                        ↪  TRUE)],
              levels = c("low", "med", "high"), ordered = TRUE)

fit <- polr(y ~ x, data = d, Hess = TRUE)
summary(fit)
exp(coef(fit))

# Test proportional odds
brant(fit)
```

7.32.6 Output & Results

`polr()` returns coefficients on the cumulative-logit scale plus the cut-point estimates. Exponentiating gives cumulative odds ratios. `brant()` reports an omnibus test for proportional odds plus per-coefficient tests; significant results indicate the assumption fails for that covariate.

7.32.7 Interpretation

A reporting sentence: "Ordinal logistic regression estimated that each 1-SD increase in x multiplied the cumulative odds of being in a higher category by 2.10 (95 % CI 1.50 to 2.94, $p < 0.001$); Brant's test did not reject the proportional-odds assumption (omnibus $p = 0.44$, all per-covariate $p > 0.20$). The cumulative odds-ratio interpretation applies uniformly across the low-vs-(med + high) and (low + med)-vs-high cut-points." Always test and report the proportional-odds assumption.

7.32.8 Practical Tips

- Always test the proportional-odds assumption with Brant's test or LRT against a partial-proportional-odds model; reporting cumulative ORs without a check is incomplete.

- When proportional odds fails for one or more covariates, use partial-proportional-odds (`VGAM::vglm` with `cumulative(parallel=FALSE~x)`) or generalised ordered logit (`oglmx`).
- For three or four categories with severe non-proportionality, multinomial logistic regression is sometimes simpler than partial-proportional-odds, at the cost of ignoring the ordering.
- `ordinal::clm()` provides a modern implementation with mixed-effects extensions (`clmm`) and richer post-hoc tools.
- Report cumulative odds ratios with confidence intervals; the phrasing "odds of being in a higher category" is the clearest way to communicate the cumulative-OR interpretation.
- For predictions on the absolute-probability scale, use `predict(fit,type="probs")`; readers absorb predicted category probabilities better than cumulative odds ratios.

7.32.9 R Packages Used

`MASS::polr()` for the canonical proportional-odds implementation; `ordinal::clm()` for a modern alternative with mixed-effects extensions; `brant` for proportional-odds testing; `VGAM` for partial-proportional-odds and other ordinal extensions; `oglmx` for generalised ordered logit; `gtsummary::tbl_regression()` for publication-ready tables.

7.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.32.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression

- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.33 Partial Correlation

7.33.1 Introduction

Partial correlation measures the linear association between two variables after removing the linear influence of one or more covariates from each. Where ordinary correlation captures bivariate association including any indirect effect through confounders, partial correlation isolates the direct linear relationship. It is the bivariate analogue of an adjusted regression coefficient and one of the simplest tools for thinking about confounding in observational data.

7.33.2 Prerequisites

A working understanding of Pearson correlation, multiple linear regression, and the concept of confounding.

7.33.3 Theory

For three variables X, Y, Z, the partial correlation of X and Y adjusted for Z is

$$r_{XY\cdot Z} = \frac{r_{XY} - r_{XZ}r_{YZ}}{\sqrt{(1 - r_{XZ}^2)(1 - r_{YZ}^2)}}.$$

It equals the Pearson correlation of the residuals from regressing X on Z and Y on Z. The construction generalises to controlling for multiple covariates: regress out a vector of covariates from each, correlate the residuals.

The semi-partial (or part) correlation removes the influence of Z from X only, retaining Y's association with Z. In multiple regression, the partial r of predictor j equals $\operatorname{sign}(\hat{\beta}_j)\sqrt{t_j^2/(t_j^2 + \mathrm{df}_{\mathrm{res}})}$.

7.33.4 Assumptions

Linearity in each pairwise relationship; multivariate Normality (for the standard inference test); accurately measured covariates; the partial-correlation interpretation is causal only under additional assumptions about the covariate set.

7.33.5 R Implementation

```r
library(ppcor)

# Partial correlation of wt and mpg given hp
pcor.test(mtcars$wt, mtcars$mpg, mtcars$hp)

# All partial correlations in a dataset
pcor(mtcars[, c("mpg", "wt", "hp", "disp")])$estimate

# Manual via residuals
res_wt  <- resid(lm(wt ~ hp, data = mtcars))
res_mpg <- resid(lm(mpg ~ hp, data = mtcars))
cor(res_wt, res_mpg)
```

7.33.6 Output & Results

`pcor.test()` returns the partial correlation estimate, its t-statistic, and a p-value. `pcor()` returns the full partial-correlation matrix, useful when many adjustments are simultaneously of interest. The manual residual-based computation confirms the algebraic identity.

7.33.7 Interpretation

A reporting sentence: "After adjusting for horsepower, weight and miles-per-gallon had a partial correlation of -0.66 ($p < 0.001$, $n = 32$); the negative association persists strongly even after controlling for the engine's power output, suggesting an independent role for vehicle weight beyond its correlation with engine size." Distinguish partial from semi-partial correlation in reports; readers often confuse them.

7.33.8 Practical Tips

- Partial correlation is the bivariate analogue of an adjusted regression coefficient; if you have already fitted a multiple regression, the corresponding partial r comes for free from the t-statistic.
- Semi-partial correlations describe "unique variance contribution" in multiple regression; often what is reported as ΔR^2 when adding one predictor to an existing model.
- For non-Normal or ordinal data, use partial Spearman correlation (`ppcor::pcor(method="spearman")`); the rank-based version is more robust to outliers.
- For graphical / network analyses, partial correlations are the natural input to Gaussian graphical models (`qgraph`, `huge`); they encode conditional dependencies in a multivariate Normal.

- Causal interpretation of partial correlations requires that the conditioning set blocks all confounding paths; statistical independence after conditioning does not by itself imply causation.
- For high-dimensional data (p near n), partial correlations from the inverse covariance matrix are unstable; use regularised graphical-lasso estimators (`glasso`).

7.33.9 R Packages Used

`ppcor` for `pcor()` and `pcor.test()`; `psych::partial.r()` for an alternative interface; `qgraph` and `huge` for partial-correlation networks; `glasso` for regularised inverse-covariance estimation in high dimensions.

7.33.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.33.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.34 Poisson Regression

7.34.1 Introduction

Poisson regression models count data with a log link, making it the canonical GLM for events per unit exposure: hospital admissions per month, mutations per kilobase, falls per resident-year. The log-linear structure means coefficients are interpretable as log-rate ratios, and the model accommodates varying exposures naturally through an offset term. The defining assumption — equidispersion, $\mathrm{Var}(Y) = \mathbb{E}[Y]$ — is also its main limitation: real count data are typically overdispersed, and either quasi-Poisson, negative binomial, or zero-inflated extensions are then required.

7.34.2 Prerequisites

A working understanding of the Poisson distribution, generalised linear models, the log link, and the role of an offset for modelling rates rather than counts.

7.34.3 Theory

The model is $Y \sim \mathrm{Poisson}(\mu)$ with $\log \mu = X^\top \beta$. Exponentiated coefficients are rate ratios: $\exp(\beta_j)$ multiplies the expected count per unit increase in X_j, all else equal. For rate-per-exposure problems, an offset of $\log(\text{exposure})$ converts the model into a rate model:

$$\log \mathbb{E}[Y] = \log(\text{exposure}) + X^\top \beta.$$

The offset's coefficient is fixed at 1, so the remaining β represent log-rate ratios per unit covariate change.

Equidispersion is the central assumption; if the variance exceeds the mean, standard errors are underestimated and inference is anti-conservative. Pearson chi-squared divided by residual degrees of freedom is the standard dispersion check; values much greater than 1 motivate quasi-Poisson or negative binomial.

7.34.4 Assumptions

Non-negative integer outcomes, independent observations, equidispersion (variance equal to mean), and a linear relationship between covariates and the log-rate.

7.34.5 R Implementation

```
library(broom)

# Simulated: event counts per patient-year
```

```r
set.seed(2026)
d <- data.frame(
  age = rnorm(300, 60, 12),
  smoker = rbinom(300, 1, 0.3),
  exposure_years = runif(300, 0.5, 5)
)
d$events <- rpois(nrow(d),
  lambda = d$exposure_years * exp(-2 + 0.02 * d$age + 0.6 * d$smoker))

fit <- glm(events ~ age + smoker, data = d, family = poisson,
           offset = log(exposure_years))
tidy(fit, conf.int = TRUE, exponentiate = TRUE)
glance(fit)

# Check overdispersion
dispersion <- sum(resid(fit, type = "pearson")^2) / fit$df.residual
dispersion
```

7.34.6 Output & Results

`tidy(fit,exponentiate=TRUE)` returns rate ratios and confidence intervals; `glance()` provides AIC, deviance, and other model-level summaries. The dispersion ratio close to 1 confirms Poisson is appropriate; values $\gg 1$ suggest overdispersion that needs different family handling.

7.34.7 Interpretation

A reporting sentence: "After adjusting for age (rate ratio 1.02 per year, 95 % CI 1.01 to 1.03), smokers had an event rate 1.81 times that of non-smokers (95 % CI 1.40 to 2.34, $p < 0.001$); the dispersion ratio of 1.05 is consistent with the Poisson assumption." Always report rate ratios on the multiplicative scale and check dispersion.

7.34.8 Practical Tips

- Check the dispersion statistic immediately after fitting; if $\gg 1$, switch to quasi-Poisson (`family=quasipoisson`) for inflated SEs or to negative binomial (`MASS::glm.nb`) for explicit overdispersion modelling.
- Include an offset whenever exposure varies across observations; without it, coefficients absorb the exposure variation and are biased.
- Coefficients are on the log-rate scale; exponentiate for rate ratios in publication-friendly units.
- For excessive zero counts beyond what Poisson predicts, fit a zero-inflated or hurdle model (`pscl::zeroinfl`, `glmmTMB`).
- For count data with hierarchical structure (repeated measures, clustered

observations), use Poisson mixed models (`glmer(family=poisson)` or `glmmTMB(family=poisson)`).
- Pearson residuals should look approximately uniformly scattered; systematic patterns suggest model misspecification (missing covariates, non-linear effects, overdispersion).

7.34.9 R Packages Used

Base R `glm()` with `family=poisson`; `MASS::glm.nb()` for negative binomial; `pscl::zeroinfl()` for zero-inflated Poisson; `glmmTMB` for mixed-effects Poisson, NB, and zero-inflated count models with cleaner overdispersion handling; `broom::tidy()` and `glance()` for tidy summaries.

7.34.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.34.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.35 Polynomial Regression

7.35.1 Introduction

Polynomial regression fits the outcome as a sum of powers of a predictor: linear, quadratic, cubic, and higher. It is the simplest way to model a non-linear relationship without moving outside the linear-model framework. Quadratic and cubic terms can capture the curvature of biological dose-response, growth, and saturation curves, and the resulting model retains the familiar OLS inference tools. The principal weakness — wild extrapolation outside the data range and instability at high degrees — motivates the more disciplined alternative of restricted cubic splines.

7.35.2 Prerequisites

A working understanding of simple linear regression, the geometric meaning of coefficients, and the bias-variance trade-off as model complexity grows.

7.35.3 Theory

The polynomial model is

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_d X^d + \varepsilon.$$

Two parameterisations coexist:

- **Orthogonal polynomials** (`poly(X,d)`): basis functions are mutually uncorrelated. Numerically stable for high degrees; individual coefficients lose intuitive interpretation but predictions and overall fit are identical to raw polynomials.
- **Raw polynomials** (`poly(X,d,raw=TRUE)` or `I(X^2)`): retain the natural "per unit" interpretation but introduce collinearity that inflates standard errors of individual coefficients.

High-degree polynomials over-fit and extrapolate poorly; degrees 2 or 3 cover most applications, and beyond that splines (natural cubic, restricted cubic) are usually preferable because they constrain tail behaviour.

7.35.4 Assumptions

Standard regression assumptions on residuals; the polynomial degree is high enough to capture the underlying curvature but not so high that it chases noise.

7.35.5 R Implementation

```
# Simulated quadratic relationship
```

```r
set.seed(2026)
x <- runif(100, 0, 10)
y <- 2 + 3 * x - 0.4 * x^2 + rnorm(100, 0, 2)

# Orthogonal quadratic
fit_ortho <- lm(y ~ poly(x, 2))
summary(fit_ortho)

# Raw quadratic (interpretable coefficients)
fit_raw <- lm(y ~ poly(x, 2, raw = TRUE))
summary(fit_raw)

# Visualise
plot(x, y, pch = 20)
curve(predict(fit_raw, newdata = data.frame(x)), add = TRUE,
      col = "#2A9D8F", lwd = 2)
```

7.35.6 Output & Results

Both fits produce identical R^2, residuals, and predictions; the coefficient values differ because of the parameterisation. Visualising the fitted curve overlaid on the data is the primary inferential output, since coefficients on individual polynomial terms are not directly interpretable.

7.35.7 Interpretation

A reporting sentence: "A quadratic regression of y on x recovered the simulated curvature ($R^2 = 0.78$); the raw-polynomial coefficients give a peak at $\hat{x} = -\hat{\beta}_1/(2\hat{\beta}_2) = 3.75$, consistent with the data-generating function. A linear-only fit performed substantially worse ($R^2 = 0.30$) and missed the curvature entirely." Always show the fitted curve; coefficient tables alone do not communicate non-linear structure.

7.35.8 Practical Tips

- Degrees 2 or 3 usually suffice for biological and clinical applications; beyond that, splines are more disciplined.
- Orthogonal polynomials are more numerically stable for high degrees and avoid the matrix-conditioning problems that raw polynomials produce.
- Use `anova(fit1,fit2)` to test whether a higher-order term improves the fit; the partial F-test is the right tool.
- Do not extrapolate beyond the range of X; polynomials diverge to $\pm\infty$ rapidly outside the support.
- For multiple predictors with non-linear effects, GAMs (`mgcv::gam`) automate smoothness selection and avoid the manual choice of polynomial

degree.
- Restricted cubic splines (`rms::rcs`) are usually preferable to high-order polynomials in clinical reporting because they constrain tail behaviour to be linear.

7.35.9 R Packages Used

Base R `lm()` with `poly()` for orthogonal or raw polynomials; `splines::ns()` and `rms::rcs()` for spline alternatives; `mgcv::gam()` for non-parametric smooth-effect modelling; `car::residualPlots()` for diagnostic comparison of polynomial degrees.

7.35.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.35.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.36 Probit Regression

7.36.1 Introduction

Probit regression models a binary outcome through the inverse of the standard Normal cumulative distribution function — the same idea as logistic regression but with a Normal-tailed link instead of a logistic-tailed one. The two methods produce essentially identical predicted probabilities; their coefficients differ by a fixed scaling factor of about 1.6. Probit is the natural choice when a latent Normal-variable interpretation matches the substantive theory; logistic is more common in clinical reporting because odds ratios are easier to communicate.

7.36.2 Prerequisites

A working understanding of logistic regression, the standard Normal CDF, and the latent-variable interpretation of binary regression.

7.36.3 Theory

The probit model is

$$\mathrm{P}(Y = 1 \mid X) = \Phi(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p),$$

where Φ is the standard Normal CDF. Coefficients are interpretable on the probit scale: a one-unit increase in X_j corresponds to a β_j-standard-deviation change in the underlying latent Normal propensity. The inverse relationship $\Phi^{-1}(p) = X^\top \beta$ is the probit link.

Probit coefficients are approximately logistic coefficients divided by 1.6 ($\sigma_{\text{logistic}} = \pi/\sqrt{3} \approx 1.81$ vs. $\sigma_{\text{Normal}} = 1$). The link functions agree near $p = 0.5$ but diverge slightly in the tails — logistic has heavier tails — so coefficient comparisons across the two scales require the rescaling.

7.36.4 Assumptions

Binary outcome, independent observations, latent linear relationship on the probit scale, and adequate sample size relative to predictors (events-per-variable ≥ 10).

7.36.5 R Implementation

```
d <- mtcars
d$am <- factor(d$am)

fit_probit <- glm(am ~ wt + hp, data = d,
                  family = binomial(link = "probit"))
```

```r
summary(fit_probit)

# Compare with logit
fit_logit <- glm(am ~ wt + hp, data = d, family = binomial)

# Predictions should be similar
pred_p <- predict(fit_probit, type = "response")
pred_l <- predict(fit_logit, type = "response")
cor(pred_p, pred_l)
```

7.36.6 Output & Results

`summary(fit_probit)` returns coefficients on the probit scale with Wald z-tests. The correlation between probit and logit predicted probabilities is typically above 0.999, confirming that the two methods are interchangeable for prediction purposes.

7.36.7 Interpretation

A reporting sentence: "Probit regression estimated that each 1000-lb increase in vehicle weight reduced the latent propensity for manual transmission by 1.8 SDs (probit coefficient -1.8, 95 % CI -3.0 to -0.7, $p = 0.002$); the average marginal effect on the probability scale was -0.27 per 1000 lbs (95 % CI -0.42 to -0.13, computed via `margins`)." Always pair probit coefficients with average marginal effects when communicating to non-statistical audiences.

7.36.8 Practical Tips

- Probit and logistic give essentially identical predicted probabilities; choose based on convention or latent-variable interpretation, not on prediction accuracy.
- Probit coefficients lack the clean odds-ratio interpretation of logistic; report average marginal effects (`margins::margins`, `marginaleffects::avg_predictions`) for clinical communication.
- For multilevel or Bayesian binary models, probit links are sometimes computationally simpler than logistic; `brms` and `Stan` handle both.
- The probit-vs-logit choice is almost always a matter of reporting convention rather than statistical superiority.
- For ordinal extensions, probit gives the cumulative-probit (or threshold) model; for survival analysis, the probit link appears in some accelerated-failure-time formulations.
- Predictions from both models should agree to within 1–2 percentage points across the full range of fitted probabilities; large discrepancies signal model misspecification.

7.36.9 R Packages Used

Base R `glm()` with `family=binomial(link="probit")`; `margins::margins()` and `marginaleffects::avg_predictions()` for average marginal effects; `mfx::probit` `mfx()` for direct marginal-effect output; `mvProbit` for multivariate probit when multiple binary outcomes are jointly modelled.

7.36.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.36.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.37 Quantile Regression

7.37.1 Introduction

Quantile regression estimates a chosen conditional quantile of the outcome (median, lower percentile, upper percentile) rather than the conditional mean. The result is a richer description of how covariates affect the outcome distribution: the same covariate may have a different effect at the 10th percentile (where it shifts the lower tail) than at the 90th (where it shifts the upper tail). Median regression is the L1 alternative to OLS and robust to outliers; multi-quantile fits expose heteroscedasticity and asymmetric effects that mean regression hides.

7.37.2 Prerequisites

A working understanding of linear regression, conditional quantiles, and the difference between modelling the mean versus the median or other percentiles of a distribution.

7.37.3 Theory

Mean regression minimises squared residuals; quantile regression minimises an asymmetric absolute-value loss:

$$\sum_i \rho_\tau(y_i - x_i^\top \beta), \qquad \rho_\tau(u) = u(\tau - \mathbf{1}\{u < 0\}).$$

Setting $\tau = 0.5$ gives median regression. The objective is convex but non-differentiable; standard solvers use linear programming. Standard errors come from rank-based or bootstrap methods. Inference at multiple τ values produces a quantile process — the trajectory of each coefficient as τ varies — which is the most informative summary of how covariates shift the conditional distribution.

7.37.4 Assumptions

Fewer than OLS: independence and correct specification at the chosen quantile. No Normality, no homoscedasticity, no constant variance assumption. The robustness comes from the bounded influence of any single observation.

7.37.5 R Implementation

```r
library(quantreg)

fit_median <- rq(mpg ~ wt + hp, data = mtcars, tau = 0.5)
summary(fit_median)

# Multiple quantiles
fit_q <- rq(mpg ~ wt, data = mtcars, tau = c(0.1, 0.25, 0.5, 0.75, 0.9))
summary(fit_q)
plot(summary(fit_q))

# Comparison to OLS
fit_lm <- lm(mpg ~ wt, data = mtcars)
coef(fit_lm)
coef(rq(mpg ~ wt, data = mtcars, tau = 0.5))
```

7.37.6 Output & Results

`rq()` returns coefficient estimates per requested τ with rank-based standard errors. The `plot(summary())` method shows each coefficient as a function of τ, with confidence bands — the canonical visualisation of how covariate effects shift across the conditional distribution.

7.37.7 Interpretation

A reporting sentence: "Quantile regression of mpg on weight at five quantiles ($\tau \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$) showed that each 1000-lb increase in weight reduced mpg by 6.8 at the 10th percentile, 5.7 at the median, and 4.2 at the 90th percentile (all $p < 0.001$); the gradient indicates heteroscedasticity that ordinary mean regression would obscure." Always plot the quantile process when reporting multi-τ results; it communicates more than a coefficient table.

7.37.8 Practical Tips

- Median regression ($\tau = 0.5$) is the L1 alternative to OLS and robust to outliers; reach for it when residual diagnostics show heavy tails.
- Multi-quantile fits expose heteroscedasticity and distributional shape changes; report at least three quantiles ($\tau \in \{0.25, 0.5, 0.75\}$) to characterise the conditional distribution.
- Bootstrap standard errors (`se="boot"`) are more reliable than rank-based in small samples; specify `R=999` resamples.
- For non-linear effects, `quantreg::nlrq()` and `quantreg::rqss()` add non-linear and smoothing-spline extensions.
- For longitudinal or clustered data, `lqmm::lqmm()` or `qrLMM::QRLMM()` extend quantile regression to mixed-effects.
- Quantile regression does not require homoscedasticity and is often more informative than OLS plus robust SEs for heteroscedastic data.

7.37.9 R Packages Used

`quantreg` for `rq()`, `rqss()`, and bootstrap SEs; `lqmm` for linear quantile mixed-effects models; `qrLMM` for quantile regression in linear mixed models with random intercepts; `quantregForest` for random-forest extensions to quantile prediction.

7.37.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.37.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.38 Quasi-Poisson Regression

7.38.1 Introduction

Quasi-Poisson regression is the simplest fix for overdispersion in count data: keep the Poisson mean structure $\log \mu = X^\top \beta$, but introduce a free dispersion parameter ϕ that scales the variance to $\phi\mu$ rather than the Poisson-mandated μ. Coefficient estimates are identical to Poisson; only the standard errors change, multiplied by $\sqrt{\phi}$. It is the workhorse "quick fix" when a Poisson dispersion check fails by a moderate amount and you do not need a likelihood-based comparison.

7.38.2 Prerequisites

A working understanding of Poisson regression, generalised linear models, and the concept of overdispersion.

7.38.3 Theory

Quasi-Poisson uses quasi-likelihood: it specifies only the first two moments $\mathbb{E}[Y] = \mu$ and $\mathrm{Var}(Y) = \phi\mu$, without committing to a particular distribution. The dispersion parameter is estimated by Pearson chi-squared divided by residual degrees of freedom. Coefficient estimates are unchanged from Poisson; standard errors are inflated by $\sqrt{\hat{\phi}}$, and confidence intervals widen accordingly.

Unlike the negative binomial — which has a genuine likelihood and AIC —

quasi-Poisson is a quasi-likelihood model: no AIC, no BIC, no likelihood-ratio tests in the usual sense. F-type tests via `anova(fit,test="F")` are appropriate for comparing nested quasi-Poisson models.

7.38.4 Assumptions

Non-negative integer outcomes, the variance-mean relationship $\text{Var}(Y) = \phi\mu$ (linear in the mean), and independent observations. For quadratic variance-mean (typical of strongly overdispersed counts), negative binomial is more appropriate.

7.38.5 R Implementation

```r
set.seed(2026)
d <- data.frame(x = rnorm(200))
d$y <- rnbinom(200, mu = exp(1 + 0.5 * d$x), size = 2)   # overdispersed

fit_pois <- glm(y ~ x, data = d, family = poisson)
fit_qp   <- glm(y ~ x, data = d, family = quasipoisson)

# Dispersion estimate
summary(fit_qp)$dispersion
# Compare SE
summary(fit_pois)$coefficients
summary(fit_qp)$coefficients
```

7.38.6 Output & Results

The `quasipoisson` family produces the same point estimates as Poisson but with inflated standard errors. The dispersion estimate is in `summary(fit)$dispersion`; comparing the two coefficient summaries shows how much SEs change.

7.38.7 Interpretation

A reporting sentence: "Quasi-Poisson regression estimated a dispersion of $\hat{\phi} = 2.8$, indicating substantial overdispersion; standard errors were inflated by $\sqrt{2.8} = 1.67$ relative to Poisson, giving a 95 % confidence interval for the rate ratio per unit x of 1.30 to 2.00 (Poisson misleadingly gave 1.45 to 1.85)." Always quote the dispersion alongside the inflated SEs.

7.38.8 Practical Tips

- Quasi-Poisson is acceptable for estimation and confidence intervals, but it lacks a proper likelihood — no AIC, BIC, or LRTs in the usual sense.

- For formal model selection by AIC, switch to negative binomial (`MASS::glm.nb` or `glmmTMB(family=nbinom2)`); the additional dispersion parameter is genuinely estimated.
- Quasi-Poisson assumes linear variance-mean ($\mathrm{Var}(Y) = \phi\mu$); NB assumes quadratic ($\mathrm{Var}(Y) = \mu + \mu^2/\theta$). Inspect Pearson residuals against fitted values to choose between them.
- For large dispersion ($\hat{\phi} > 3$), neither quasi-Poisson nor NB may suffice; consider zero-inflated or mixture models.
- Use F-tests (`anova(fit,test="F")`) for nested quasi-Poisson model comparisons; LRTs are not available.
- For longitudinal or clustered counts, `glmmTMB(family=nbinom2)` is preferable to quasi-Poisson because it provides random effects and a proper likelihood.

7.38.9 R Packages Used

Base R `glm()` with `family=quasipoisson`; `MASS::glm.nb()` for negative binomial as the recommended alternative; `glmmTMB` for mixed-effects count models with proper likelihoods; `DHARMa` for residual diagnostics across count-model families.

7.38.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.38.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean

- Decision curves, NRI, IDI

7.39 R-Squared and Adjusted R-Squared

7.39.1 Introduction

R^2 is the proportion of variance in the outcome explained by the regression — the most commonly reported summary of regression fit. Its weakness is well known: R^2 never decreases when predictors are added, so it cannot distinguish a useful predictor from a noise variable, and high R^2 alone does not imply a useful model. Adjusted R^2 penalises for the number of predictors and decreases when an added predictor does not earn its complexity. For predictive purposes, cross-validated R^2 is more honest still.

7.39.2 Prerequisites

A working understanding of multiple regression, residual sum of squares, and the bias-variance trade-off in model selection.

7.39.3 Theory

$$R^2 = 1 - \frac{\mathrm{SS}_{\mathrm{res}}}{\mathrm{SS}_{\mathrm{tot}}}.$$

It equals the squared correlation between observed and fitted values for OLS with an intercept. Always between 0 and 1; always non-decreasing as predictors are added.

Adjusted R^2 corrects for the penalty:

$$R^2_{\mathrm{adj}} = 1 - (1 - R^2) \cdot \frac{n - 1}{n - p - 1},$$

with p the number of predictors. Adjusted R^2 can decrease when an added predictor does not improve fit enough to compensate for the lost degree of freedom; it can even be negative for very poorly fitting models.

For prediction, both are in-sample summaries; cross-validated R^2 (`1-SSE_{CV}/SS_{tot}`) replaces the residual sum with a CV-based estimate of out-of-sample error and corrects for the optimism of in-sample fit.

7.39.4 Assumptions

Standard regression assumptions for the underlying linear model. R^2 is well-defined for any model with an intercept; without an intercept, definitions differ across software.

7.39.5 R Implementation

```r
fit <- lm(mpg ~ wt + hp + disp, data = mtcars)
s <- summary(fit)
c(R2 = s$r.squared, R2_adj = s$adj.r.squared)

# Cross-validated R^2
library(caret)
set.seed(2026)
cv <- train(mpg ~ wt + hp + disp, data = mtcars,
            method = "lm",
            trControl = trainControl(method = "cv", number = 10))
cv$results
```

7.39.6 Output & Results

`summary(fit)$r.squared` and `$adj.r.squared` give in-sample values; `caret::train()` returns cross-validated R^2 and RMSE under 10-fold CV. The gap between in-sample and CV R^2 quantifies optimism — large gaps indicate overfitting.

7.39.7 Interpretation

A reporting sentence: "The model explained 83 % of the variance in mpg ($R^2 = 0.83$, adjusted $R^2 = 0.81$); 10-fold cross-validated R^2 was 0.79, indicating modest optimism in the in-sample estimate. Adjusted R^2 confirmed that all three predictors contributed beyond their penalty." Always report adjusted R^2 alongside R^2 for multi-predictor models, and CV R^2 when prediction is the goal.

7.39.8 Practical Tips

- Report adjusted R^2, not just R^2, whenever predictors number more than one; the unadjusted version misleads about model parsimony.
- Adjusted R^2 can be negative for very poorly fitting models; this is informative, not a bug.
- For predictive modelling, prefer cross-validated R^2 or RMSE; in-sample R^2 is optimistic by design.
- Do not compare R^2 across different datasets with different outcome variance; the comparison is meaningless.
- Avoid causal language: "the model explains 83 % of the variance" is correct; "the model predicts 83 % of the outcome" is not.
- For GLMs, pseudo-R^2 measures (Cox-Snell, Nagelkerke, McFadden) generalise the concept; report the specific variant used because they give different numerical answers.

7.39.9 R Packages Used

Base R `summary.lm()` and `summary.glm()`; `performance::r2()` for tidy R^2 across model classes; `caret` and `tidymodels` for cross-validated R^2; `MuMIn::r.squaredGLMM()` for marginal and conditional R^2 in mixed models.

7.39.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.39.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.40 Random-Intercept Models

7.40.1 Introduction

A random-intercept model allows each cluster (subject, hospital, school) its own baseline intercept while constraining all clusters to share the same regression slope. It is the simplest mixed-effects model and the natural starting point for repeated-measures, longitudinal, or clustered data. The single new parameter beyond ordinary regression — the variance of the random intercept — captures between-cluster heterogeneity in baseline level and is the primary handle for partial pooling and intraclass correlation.

7.40.2 Prerequisites

A working understanding of mixed-effects models, the difference between fixed and random effects, and basic familiarity with `lme4` formula syntax.

7.40.3 Theory

The random-intercept model is

$$Y_{ij} = \beta_0 + u_i + \beta_1 X_{ij} + \varepsilon_{ij}, \qquad u_i \sim \mathcal{N}(0, \sigma_u^2), \qquad \varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2).$$

The cluster-specific intercept $\beta_0 + u_i$ varies around the population intercept β_0 with variance σ_u^2. The intraclass correlation coefficient is

$$\rho = \frac{\sigma_u^2}{\sigma_u^2 + \sigma^2},$$

interpretable as the proportion of total variance attributable to between-cluster differences. A high ICC (> 0.30) indicates substantial cluster-level structure; a low ICC (< 0.05) suggests the random effect is unnecessary and ordinary regression suffices.

7.40.4 Assumptions

Random intercepts are independent and Normally distributed across clusters; residuals are independent and Normal conditional on random effects; all clusters share a common slope (relax via random slopes when this fails).

7.40.5 R Implementation

```r
library(lme4); library(lmerTest)

data(sleepstudy, package = "lme4")
fit <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)
summary(fit)

# ICC
vc <- as.data.frame(VarCorr(fit))
icc <- vc$vcov[1] / sum(vc$vcov)
icc
```

7.40.6 Output & Results

`summary(fit)` reports the fixed effects (common slope and population intercept) with `lmerTest`-supplied p-values, plus the random-intercept SD and residual SD.

The ICC is computed from the variance components and is the primary single-number summary of cluster-level structure.

7.40.7 Interpretation

A reporting sentence: "A random-intercept model estimated the population-level slope at 10.47 ms per day of sleep deprivation (95 % CI 8.92 to 12.02, $p < 0.001$); between-subject variation in baseline reaction time was substantial ($\hat{\sigma}_u = 37$ ms), with intraclass correlation 0.39 indicating that 39 % of total variance is attributable to between-subject differences." Always report the ICC alongside fixed effects.

7.40.8 Practical Tips

- Always report the ICC; it is the headline measure of cluster-level structure and tells readers whether the random-intercept model was warranted.
- For small numbers of clusters (< 5), random-effect variance estimates are unstable; consider a simpler model with fixed effects per cluster, or move to Bayesian estimation with weakly informative priors.
- Compare against a no-random-effects model via likelihood-ratio test (with the χ^2_0 / χ^2_1 mixture reference for the boundary null).
- Use `confint(fit,method="profile")` for profile-likelihood confidence intervals on random-effect variances; Wald intervals are unreliable on the boundary.
- If slopes clearly vary across clusters (visible in spaghetti plots), add random slopes via `(1+X|cluster)`; reporting only random intercepts when slopes vary biases inference.
- For non-Gaussian outcomes, the same syntax extends via `glmer()` with the appropriate family.

7.40.9 R Packages Used

`lme4` for `lmer()`, `glmer()`, and `VarCorr()`; `lmerTest` for Satterthwaite or Kenward-Roger p-values; `performance::icc()` for tidy ICC computation; `pbkrtest::PBmodcomp()` for parametric-bootstrap LRTs; `glmmTMB` for an alternative implementation that handles complex error structures.

7.40.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.40.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.41 Random-Slope Models

7.41.1 Introduction

Random-slope models allow each cluster its own slope for one or more predictors, capturing between-cluster heterogeneity in how a covariate affects the outcome. Where random-intercept models assume that all clusters share the same effect of X but differ in baseline, random-slope models drop the shared-slope assumption and let the slope vary. They are essential for any longitudinal analysis where individual trajectories diverge over time and where the population-average slope may hide substantial subject-to-subject variability.

7.41.2 Prerequisites

A working understanding of random-intercept models, mixed-effects model syntax in `lme4`, and the bivariate Normal distribution for joint random effects.

7.41.3 Theory

For a predictor X varying within clusters:

$$Y_{ij} = (\beta_0 + u_{0i}) + (\beta_1 + u_{1i})X_{ij} + \varepsilon_{ij},$$

with $(u_{0i}, u_{1i})^\top \sim \mathcal{N}(0, \Sigma)$ where Σ contains intercept variance σ_0^2, slope variance σ_1^2, and correlation ρ_{01}. The "correlated random slope" formulation `(X|cl`

`uster)` estimates all three parameters jointly; the "uncorrelated" form `(X||cluster)` constrains $\rho_{01} = 0$.

A non-zero ρ_{01} indicates systematic dependence between baseline level and rate of change — for instance, subjects who start higher progress faster (positive correlation) or slower (negative correlation).

7.41.4 Assumptions

Random effects bivariate Normal, independent across clusters; conditional Normal residuals; the slope variation is real and not driven by a few influential clusters.

7.41.5 R Implementation

```r
library(lme4)

data(sleepstudy, package = "lme4")

# Random intercept + slope (correlated)
fit_rs <- lmer(Reaction ~ Days + (Days | Subject), data = sleepstudy)
summary(fit_rs)

# Uncorrelated random slope
fit_rs_unc <- lmer(Reaction ~ Days + (1 | Subject) + (0 + Days | Subject),
                   data = sleepstudy)

# Compare with random-intercept-only
fit_ri <- lmer(Reaction ~ Days + (1 | Subject), data = sleepstudy)
anova(fit_ri, fit_rs)
```

7.41.6 Output & Results

`summary(fit_rs)` reports the population-level fixed effects, the standard deviations of the random intercept and random slope, and their correlation. `anova()` between nested models compares them via a likelihood-ratio test (with the boundary-mixture reference for variance components).

7.41.7 Interpretation

A reporting sentence: "Adding a random slope on `Days` substantially improved fit ($\chi^2(2) = 42.1$, $p < 0.001$); between-subject standard deviation of the slope was 5.9 ms per day, with intercept-slope correlation 0.07. The population-level slope was 10.5 ms per day (95 % CI 7.6 to 13.4)." Always report the random-slope variance and the intercept-slope correlation alongside fixed effects.

7.41.8 Practical Tips

- Random slopes require enough observations per cluster (rule of thumb: at least 10 per cluster); with fewer, the slope variance is unstable and may produce singular fits.
- Near-zero random-slope variance signals over-specification; drop the random slope or constrain $\rho_{01} = 0$ via the `||` syntax.
- Centring the predictor (`X-mean(X)`) often reduces intercept-slope correlation that arises mechanically from the parameterisation rather than the data.
- Check the random-effects variance-covariance with `VarCorr(fit)`; correlation estimates near ± 1 are a red flag for over-fitting.
- Use parametric bootstrap (`pbkrtest::PBmodcomp`) for small-sample LRTs on random-slope variance; the asymptotic χ^2 mixture can be unreliable.
- For non-Gaussian outcomes, the same syntax extends via `glmer()` with the appropriate family.

7.41.9 R Packages Used

`lme4` for `lmer()` and `glmer()`; `lmerTest` for p-values on fixed effects; `pbkrtest` for parametric-bootstrap LRTs on variance components; `glmmTMB` for an alternative implementation that handles complex error structures; `merTools` for prediction intervals that respect random-slope uncertainty.

7.41.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.41.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)

- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.42 Linear Regression Assumptions

7.42.1 Introduction

Linear regression rests on four classical assumptions — linearity, independence, homoscedasticity, and Normality of residuals — each with characteristic diagnostics and remedies when violated. Distinguishing the four is essential because their failures have very different consequences: non-linearity biases coefficients, dependence deflates standard errors, heteroscedasticity gives unbiased coefficients with wrong SEs, and Normality matters mainly for small-sample inference. A responsible analysis checks all four and reports the verdict explicitly.

7.42.2 Prerequisites

A working understanding of linear regression, residuals, and the standard regression diagnostics.

7.42.3 Theory

- **Linearity**: $\mathbb{E}[Y \mid X] = \beta_0 + \sum \beta_j X_j$. Failure produces biased coefficients regardless of sample size.
- **Independence**: residuals uncorrelated with each other (and with predictors). Time-series, repeated-measures, or clustered data violate this; the consequence is downwardly biased standard errors and inflated Type I error.
- **Homoscedasticity**: $\mathrm{Var}(\varepsilon \mid X) = \sigma^2$ constant across predictor space. Failure produces unbiased coefficients but wrong standard errors; the fix is robust (sandwich) SEs or weighted least squares.
- **Normality of residuals**: needed for exact small-sample t-test and confidence-interval validity. The central limit theorem makes this assumption mild for large n, where coefficient sampling distributions are approximately Normal regardless of residual distribution.

7.42.4 Assumptions

These four are themselves the assumptions. Diagnostics aim to detect violations, not to "prove" the assumptions hold.

7.42.5 R Implementation

```r
library(car); library(performance); library(lmtest)

fit <- lm(mpg ~ wt + hp, data = mtcars)

# Residuals-vs-fitted: linearity and homoscedasticity
plot(fit, which = 1)

# Q-Q plot of residuals: normality
plot(fit, which = 2)

# Scale-location: homoscedasticity
plot(fit, which = 3)

# Residuals vs leverage: influence
plot(fit, which = 5)

# Formal tests
bptest(fit)     # Breusch-Pagan for heteroscedasticity
shapiro.test(residuals(fit))
dwtest(fit)     # Durbin-Watson for autocorrelation

# performance::check_model combines all
check_model(fit)
```

7.42.6 Output & Results

Four diagnostic plots from `plot.lm()` plus three formal tests: Breusch-Pagan for heteroscedasticity, Shapiro-Wilk for residual Normality, Durbin-Watson for first-order autocorrelation. `performance::check_model()` produces an integrated multi-panel diagnostic with annotations.

7.42.7 Interpretation

A reporting sentence: "Diagnostic plots showed no systematic patterns: residuals-vs-fitted scattered randomly around zero, scale-location was approximately horizontal, and the Q-Q plot showed only mild tail deviation. Formal tests confirmed: Breusch-Pagan $p = 0.41$ (no evidence of heteroscedasticity), Shapiro-Wilk $p = 0.32$ (residuals consistent with Normality), Durbin-Watson 1.95 (no autocorrelation). All four classical assumptions are reasonably supported." Always report what was checked and the verdict.

7.42.8 Practical Tips

- Graphical diagnostics are more informative than formal tests, especially at small samples; formal tests have low power below $n = 50$ and reject too easily above $n = 1{,}000$.
- Heteroscedasticity is common; use robust (HC3) SEs from `sandwich::vcovHC` combined with `lmtest::coeftest` if detected. Alternatively, transform the response or use weighted least squares.
- Clustering or repeated measures invalidate the independence assumption; fit a linear mixed model (`lme4::lmer`) rather than ordinary regression.
- Non-linearity is often fixed by a transformation (log, square root), polynomial terms, or splines. The right fix depends on the substantive functional form.
- Gross departures from Normality at large n typically still allow valid CLT-based inference; the assumption matters most for t-tests and CIs at small n.
- Reporting "all assumptions met" without specifying which were checked and how is uninformative; cite the specific diagnostics used.

7.42.9 R Packages Used

Base R `plot.lm()` and `shapiro.test()`; `lmtest::bptest()` and `dwtest()` for heteroscedasticity and autocorrelation; `car::ncvTest()` and `durbinWatsonTest()` for alternatives; `performance::check_model()` for integrated diagnostics; `sandwich` and `lmtest::coeftest()` for robust SE inference.

7.42.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.42.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls

- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.43 Residual Diagnostics

7.43.1 Introduction

Residual diagnostics are the post-fit checklist that exposes most assumption violations in regression. Four standard plots do nearly all the heavy lifting: residuals against fitted values, the Normal quantile-quantile plot, scale-location, and residuals-against-leverage. Reading these plots fluently — recognising a funnel as heteroscedasticity, a U-shape as missed non-linearity, heavy tails in the Q-Q plot as non-Normal errors — is one of the core skills of applied regression analysis.

7.43.2 Prerequisites

A working understanding of regression assumptions (linearity, homoscedasticity, Normal residuals, independence) and basic familiarity with quantile-quantile plots.

7.43.3 Theory

1. **Residuals-vs-fitted**: checks linearity and equal variance simultaneously. Random scatter around zero indicates the linear-predictor structure is adequate; a systematic trend suggests missed non-linearity; a funnel shape indicates heteroscedasticity.
2. **Normal Q-Q**: ordered standardised residuals plotted against theoretical Normal quantiles. Points on the reference line indicate Normality; heavy tails appear as S-shaped curves; skew shows as asymmetric departure.
3. **Scale-location** ($\sqrt{|r_i|}$ against fitted values): isolates the variance check. A horizontal line indicates homoscedasticity; an upward or downward trend indicates the residual variance changes with the mean.
4. **Residuals-vs-leverage**: identifies influential points. Cook's distance contours overlaid on the plot flag observations whose removal would substantially change the fit.

Studentised residuals (`rstudent()`) divide by an estimate of the residual SD that excludes observation i — preferable to ordinary standardised residuals when leverage is high.

7.43.4 Assumptions

Classical OLS assumptions; analogous diagnostics exist for GLMs (deviance and Pearson residuals) and mixed models (within- and between-cluster residuals).

7.43.5 R Implementation

```r
library(car); library(performance)

fit <- lm(mpg ~ wt + hp, data = mtcars)

# Built-in 4-panel diagnostic
par(mfrow = c(2, 2))
plot(fit)
par(mfrow = c(1, 1))

# performance's modern alternative
check_model(fit)

# Individual residual types
resid(fit)              # raw
rstandard(fit)          # standardised
rstudent(fit)           # studentised
```

7.43.6 Output & Results

Four standard diagnostic plots from `plot(fit)`. `performance::check_model()` produces a richer multi-panel diagnostic with posterior predictive checks (for Bayesian fits), influential observations flagged, and explicit pass/fail-style annotations.

7.43.7 Interpretation

A reporting sentence: "Residual diagnostics for the linear model (mpg on weight and horsepower) showed: random scatter in residuals-vs-fitted (no missed non-linearity); approximately Normal residuals with mild heavy-tail behaviour at both extremes; constant variance across the fitted range; and three observations with moderate Cook's distance (D_i between 0.15 and 0.30) but none exceeding the conventional cutoff." Always describe what each diagnostic showed; reporting "diagnostics were satisfactory" without specifics is uninformative.

7.43.8 Practical Tips

- Always plot residuals; summary statistics do not catch the patterns that visual diagnostics reveal.

- Use studentised residuals (`rstudent`) for outlier checks; they correct for leverage and have a standard t reference distribution.
- Funnel-shaped residuals-vs-fitted indicate heteroscedasticity; remedies include log-transforming the response, weighted least squares, or robust HC3 standard errors.
- U-shape in residuals-vs-fitted indicates missed non-linearity; add polynomial terms, splines, or interactions.
- Heavy-tailed Q-Q plots motivate robust regression (`MASS::rlm`) or a heavier-tailed family (t regression, or robust ML in `lmrob`).
- Diagnostic plots are interpretive, not pass/fail; reasonable judgement is required and a small departure from ideal can be acceptable in large samples.

7.43.9 R Packages Used

Base R `plot.lm()` for the standard four-panel diagnostic; `car::residualPlots()` for component-plus-residual plots; `performance::check_model()` for an integrated tidy diagnostic; `DHARMa` for simulation-based residuals on GLMs and mixed models; `MASS::rlm()` for robust regression alternatives when residual diagnostics fail.

7.43.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.43.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score

- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.44 Ridge Regression

7.44.1 Introduction

Ridge regression adds an L2 penalty $\lambda \sum_j \beta_j^2$ to the residual sum of squares. It shrinks coefficients toward zero — but never exactly to zero — reducing variance at the cost of a small bias and producing stable fits even when predictors are highly correlated. The classical motivation was multicollinearity: when $X^\top X$ is near-singular, OLS estimates are extremely variable, and the ridge penalty regularises the inverse and yields a well-defined estimator. Modern uses extend to high-dimensional regression where $p > n$ makes OLS undefined.

7.44.2 Prerequisites

A working understanding of linear regression, the bias-variance trade-off, multicollinearity, and the role of cross-validation in tuning regularisation parameters.

7.44.3 Theory

The ridge estimator minimises

$$\|y - X\beta\|^2 + \lambda \|\beta\|_2^2,$$

with closed-form solution $\hat{\beta}_{\text{ridge}} = (X^\top X + \lambda I)^{-1} X^\top y$. Larger λ shrinks coefficients more aggressively; $\lambda = 0$ recovers OLS. The coefficient paths — coefficient values as λ varies on a log scale — show how each predictor contributes; coefficients shrink continuously rather than dropping out abruptly as in lasso.

Cross-validation (typically $k = 10$ folds) selects λ. Two common rules: λ_{min} minimises CV error, λ_{1se} takes the largest λ within one standard error of the minimum (more parsimonious shrinkage).

7.44.4 Assumptions

Standard linear-model assumptions on residuals; predictors should be standardised (`glmnet` does this internally and back-transforms). For statistical inference on shrunken coefficients, the bias from regularisation must be considered; bootstrap or debiased lasso/ridge methods provide adjusted intervals.

7.44.5 R Implementation

```r
library(glmnet)

X <- as.matrix(mtcars[, -1])
y <- mtcars$mpg

set.seed(2026)
cvfit <- cv.glmnet(X, y, alpha = 0)   # alpha = 0 -> ridge

plot(cvfit)
coef(cvfit, s = "lambda.min")
coef(cvfit, s = "lambda.1se")
```

7.44.6 Output & Results

`cv.glmnet()` with `alpha=0` fits ridge regression with cross-validated λ. The plot shows mean CV error against log λ with the two selection lines. `coef()` returns shrunken coefficients at the chosen λ; all are non-zero, in contrast to lasso.

7.44.7 Interpretation

A reporting sentence: "Ridge regression with CV-optimal $\lambda_{1se} = 0.5$ produced shrunken coefficient estimates that remained stable despite the strong correlations between displacement, weight, and horsepower; the cross-validated mean squared error was 7.2 mpg^2 versus 9.8 mpg^2 for OLS, reflecting the variance reduction." Always report the chosen λ and the cross-validated error.

7.44.8 Practical Tips

- Always standardise predictors before ridge; `glmnet` does this internally with `standardize=TRUE` (default).
- Use λ_{1se} for more shrinkage than λ_{min}; the parsimonious choice typically generalises better.
- For variable selection rather than pure shrinkage, switch to lasso (`alpha=1`) or elastic net (`0<alpha<1`); ridge keeps every predictor.
- For non-Gaussian outcomes, set `family="binomial"`, `"poisson"`, etc.; the ridge penalty extends naturally.
- Set the random seed before `cv.glmnet()` to ensure reproducible λ selection.
- Compare ridge against lasso and elastic net on the same cross-validation folds when the right level of sparsity is unclear; modern penalised regression workflows benchmark all three.

7.44.9 R Packages Used

`glmnet` for canonical ridge, lasso, and elastic-net implementations with cross-validated tuning; `MASS::lm.ridge()` for an alternative ridge implementation; `hd`

`i` for high-dimensional inference including debiased ridge; `caret` and `tidymodel s` for unified cross-validated comparison across penalised methods.

7.44.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.44.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.45 Robust Regression

7.45.1 Introduction

Robust regression estimators down-weight observations with large residuals to reduce the influence of outliers and heavy-tailed errors. Where ordinary least squares minimises the squared residual — making a single large residual contribute as much as many moderate ones — robust methods cap or smoothly bound the loss function so that extreme observations cannot dominate the fit. Two main families coexist: M-estimators (Huber, bisquare) and MM-estimators that combine a high-breakdown initial fit with a high-efficiency M-step.

7.45.2 Prerequisites

A working understanding of OLS, residual diagnostics, leverage and influence, and the breakdown-vs-efficiency trade-off in robust statistics.

7.45.3 Theory

M-estimators minimise $\sum \rho(r_i/s)$ where ρ is a robust loss function: quadratic for small residuals (where it acts like OLS) and bounded or transitioning to linear for large residuals (capping the influence of outliers). Huber's ρ transitions at 1.345 SDs; bisquare's ρ is exactly zero beyond a cutoff, eliminating large outliers entirely.

MM-estimators combine a high-breakdown initial fit (S-estimator with breakdown 50 %, the maximum possible) with a high-efficiency M-estimator using the S-estimate as starting point. The result has breakdown 50 % and 95 % efficiency at Normal errors — the modern default for robust linear regression.

7.45.4 Assumptions

Outliers in the response variable; for outliers in predictors (high-leverage points), the MM-estimator still resists but the breakdown safeguard is weaker. Errors symmetric or near-symmetric; for severely skewed errors, transformation may be preferable to robust regression.

7.45.5 R Implementation

```
library(MASS); library(robustbase)

# Simulated data with 10% outliers
set.seed(2026)
x <- rnorm(100)
y <- 2 + 1.5 * x + rnorm(100)
y[sample(100, 10)] <- y[sample(100, 10)] + rnorm(10, 0, 10)

fit_ols <- lm(y ~ x)
fit_rlm <- rlm(y ~ x)                      # Huber M-estimator
fit_mm  <- lmrob(y ~ x, method = "MM")     # MM-estimator

rbind(ols = coef(fit_ols), rlm = coef(fit_rlm), mm = coef(fit_mm))
```

7.45.6 Output & Results

OLS coefficients are pulled by the contamination; both robust estimators recover values close to the data-generating slope. The MM-estimator typically delivers tighter standard errors than the Huber M-estimator at the same level of robustness, reflecting its better efficiency.

7.45.7 Interpretation

A reporting sentence: "MM-regression gave slope 1.48 (SE 0.11), close to the true 1.5; OLS on the same 10 %-contaminated data produced 1.35 (SE 0.16), pulled toward zero by the outliers. Reporting both fits illustrates the robustness benefit and the cost of ignoring outliers." Pair OLS with a robust fit when outliers are suspected; large discrepancies signal the influential observations need investigation.

7.45.8 Practical Tips

- `robustbase::lmrob()` with `method="MM"` is the modern default for robust linear regression; superior to Huber `MASS::rlm()` in efficiency-breakdown trade-off.
- Always report both OLS and robust fits; large discrepancies signal outlier influence and motivate investigation of the contaminated points.
- Robust regression is a fix for outlier influence, not a substitute for understanding why outliers occurred — investigate first, robustify second.
- For outliers in predictor space (high-leverage points), the MM-estimator still resists but the safeguard is reduced; combine with leverage-aware diagnostics.
- Robust SEs (sandwich variance) address heteroscedasticity, not outliers; the two are distinct problems requiring distinct fixes.
- For GLMs, `robustbase::glmrob()` provides analogous robust estimators for binomial, Poisson, and gamma regression.

7.45.9 R Packages Used

`MASS::rlm()` for Huber M-estimators; `robustbase::lmrob()` for MM-estimators with high-breakdown initialisation; `robustbase::glmrob()` for robust GLMs; `quantreg` for quantile regression as a complementary robust alternative; `WRS2` for Wilcox's robust ANOVA and regression analogues.

7.45.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.45.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring

- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.46 ROC and AUC for Logistic Models

7.46.1 Introduction

The receiver operating characteristic (ROC) curve and its area-under-curve (AUC) summary quantify how well a logistic regression's predicted probabilities rank observed events above non-events. AUC ranges from 0.5 (chance) to 1.0 (perfect ranking) and is the conventional discrimination metric for binary classifiers in clinical and epidemiological work. It is invariant to the threshold chosen for class assignment, making it a property of the model's ranking ability rather than any specific cut-off.

7.46.2 Prerequisites

A working understanding of logistic regression, classification thresholds, sensitivity, and specificity.

7.46.3 Theory

For a binary outcome $Y \in \{0, 1\}$ and a continuous score (predicted probability) $\hat{p}$, the AUC equals the probability that a randomly chosen positive case has a higher predicted probability than a randomly chosen negative case:

$$\text{AUC} = \text{P}(\hat{p}_i > \hat{p}_j \mid Y_i = 1, Y_j = 0).$$

It is mathematically equivalent to the Mann-Whitney U statistic divided by the sample sizes. Rough conventional benchmarks: 0.5–0.7 poor, 0.7–0.8 acceptable, 0.8–0.9 excellent, above 0.9 outstanding (and worth scrutinising for over-fitting in small samples). DeLong's test compares two correlated AUCs from models fit on the same data.

7.46.4 Assumptions

A binary outcome and a continuous score (predicted probability or any monotone transformation thereof). AUC is invariant to monotone transformations of the score, so calibration is irrelevant to AUC; thresholding decisions, however, do depend on calibration.

7.46.5 R Implementation

```r
library(pROC)

d <- mtcars
d$am <- factor(d$am)
fit <- glm(am ~ wt + hp, data = d, family = binomial)
probs <- predict(fit, type = "response")

roc_obj <- roc(d$am, probs, quiet = TRUE)
auc(roc_obj); ci.auc(roc_obj)
plot(roc_obj, col = "#2A9D8F", lwd = 2)

# DeLong test: compare two models' AUCs
fit2 <- glm(am ~ wt, data = d, family = binomial)
roc2 <- roc(d$am, predict(fit2, type = "response"), quiet = TRUE)
roc.test(roc_obj, roc2)
```

7.46.6 Output & Results

`auc(roc_obj)` returns the point estimate, `ci.auc()` the bootstrap confidence interval. The plot displays the ROC curve with a diagonal reference for chance. `roc.test()` performs DeLong's test for the AUC difference between two correlated ROC curves.

7.46.7 Interpretation

A reporting sentence: "The full logistic regression had an in-sample AUC of 0.93 (95 % CI 0.84 to 1.00); a reduced model with weight only had AUC 0.89, and DeLong's test for the difference yielded $\Delta\text{AUC} = 0.04$ ($p = 0.12$), suggesting horsepower contributes little additional discrimination beyond weight." Report AUC with confidence interval and pair with at least one operating-point summary (sensitivity / specificity at a clinically chosen threshold).

7.46.8 Practical Tips

- Always report AUC with its 95 % confidence interval; the point estimate alone is uninterpretable for assessing precision.

- For honest internal validation, use bootstrap with the `0.632+` correction for optimism; the package `rms::validate` automates this.
- In severe class imbalance, precision-recall curves communicate operating-point trade-offs better than ROC; report both when positives are rare.
- AUC is independent of prevalence; calibration and predictive values (PPV, NPV) are not. Report calibration alongside AUC for any clinical prediction tool.
- For survival outcomes, use time-dependent ROC (`timeROC::timeROC`) and time-specific AUCs at clinically meaningful horizons.
- Compare nested models via DeLong's test (`pROC::roc.test`); for non-nested models, treat the comparison as exploratory and avoid hypothesis-test framing.

7.46.9 R Packages Used

`pROC` for ROC objects, AUC, confidence intervals, and DeLong's test; `ROCit` for an alternative API with bootstrap CIs and Hand's M-statistic; `yardstick` for tidy classifier metrics in `tidymodels`; `rms::validate` for bootstrap-based optimism correction.

7.46.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.46.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean

- Decision curves, NRI, IDI

7.47 Simple Linear Regression

7.47.1 Introduction

Simple linear regression models the mean of a continuous outcome as a linear function of a single continuous predictor. It is the workhorse of applied statistics: half of all reported analyses are linear regressions in some disguise, and the concepts it introduces (least squares estimation, residuals, fit-vs-data trade-off, R-squared, confidence and prediction bands) generalise to every more complex model.

7.47.2 Prerequisites

The reader should know what a mean, a variance, and a correlation are, and should be comfortable reading a scatter plot. No prior regression experience is required.

7.47.3 Theory

Given pairs (x_i, y_i) for $i = 1, \dots, n$, the simple linear regression model is

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \qquad \varepsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2).$$

The parameters β_0 (intercept) and β_1 (slope) are estimated by ordinary least squares – minimising the sum of squared residuals $\sum (y_i - \hat{y}_i)^2$. Closed-form solutions are

$$\hat{\beta}_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}, \qquad \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}.$$

The standard error of the slope is $SE(\hat{\beta}_1) = s / \sqrt{\sum (x_i - \bar{x})^2}$, where s is the residual standard error. Under the normality assumption, $\hat{\beta}_1$ is normally distributed around β_1, and $\hat{\beta}_1 / SE(\hat{\beta}_1)$ follows a t distribution with $n - 2$ degrees of freedom under the null $\beta_1 = 0$.

The coefficient of determination is $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$ – the proportion of variance in y explained by the linear relationship with x. For simple regression, R^2 equals the squared Pearson correlation.

7.47.4 Assumptions

The OLS inference machinery relies on four assumptions:

1. **Linearity**: the true relationship between x and the mean of y is linear.
2. **Independence**: the errors ε_i are independent of each other.
3. **Homoscedasticity**: the errors have constant variance σ^2 regardless of x.
4. **Normality of errors**: the errors are normally distributed – important for small-sample inference, less so for large samples.

7.47.5 R Implementation

```r
library(broom)
library(ggplot2)
library(performance)

set.seed(7)
n <- 80
x <- runif(n, 0, 10)
y <- 2 + 1.5 * x + rnorm(n, sd = 1.5)
df <- data.frame(x, y)

fit <- lm(y ~ x, data = df)
tidy(fit, conf.int = TRUE)
glance(fit)

ggplot(df, aes(x = x, y = y)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", colour = "steelblue") +
  theme_minimal(base_size = 12)

check_model(fit)
```

The `tidy()` function returns a one-row-per-term table with estimates, standard errors, t statistics, p-values, and confidence intervals. `glance()` returns a one-row model-level summary: R-squared, adjusted R-squared, residual standard error, F statistic, and log-likelihood. `check_model()` produces the standard diagnostic panel.

7.47.6 Output & Results

Typical coefficient output:

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	1.982	0.326	6.080	<0.001	1.333	2.631
x	1.512	0.056	27.155	<0.001	1.401	1.623

$R^2 = 0.904$, residual SE $= 1.47$, F(1, 78) $= 737.4$.

7.47.7 Interpretation

The slope estimate of 1.512 (95% CI 1.401 to 1.623) says that a one-unit increase in x is associated with a 1.51-unit increase in the conditional mean of y, holding all else equal (vacuous for a simple regression, but the phrasing generalises). The intercept of 1.982 is the predicted y when $x = 0$; this is only a scientifically meaningful quantity if $x = 0$ is within the range of the data.

For a manuscript: "A simple linear regression showed that y increased significantly with x, $b = 1.51$ (95% CI 1.40 to 1.62), $t(78) = 27.2$, $p < 0.001$. The model explained 90.4% of the variance in y."

7.47.8 Practical Tips

- Always plot the data and the fitted line; numbers alone can hide gross misfit.
- Residuals-vs-fitted and Q-Q plots are the two diagnostics you should reflexively inspect.
- An R^2 of 0.9 is stellar in physics, mediocre in psychology, and suspect in epidemiology – interpret it in context.
- The confidence interval is for the *mean* response at a given x; the prediction interval, which is wider, is for a *new individual* observation.
- Extrapolating beyond the range of the observed x is risky: the linear relationship that held for $x \in [0, 10]$ may be nonsense for $x = 100$.

7.47.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.47.10 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)

- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.48 Spline Regression

7.48.1 Introduction

Splines are piecewise polynomials joined smoothly at points called knots. They give flexible non-linear fits without the wild extrapolation behaviour of high-degree polynomials, which makes them the canonical tool for modelling non-linear continuous predictor effects in regression. Natural cubic splines and restricted cubic splines are the workhorses in clinical and epidemiological work; their constraint of linear behaviour beyond the boundary knots prevents the implausible curls that ordinary polynomials produce in the tails.

7.48.2 Prerequisites

A working understanding of polynomial regression, the limitations of high-degree polynomials at the extremes of the data, and the concept of basis functions in regression.

7.48.3 Theory

A natural cubic spline with k internal knots is a piecewise cubic polynomial that is continuous together with its first and second derivatives at each knot, and linear beyond the boundary knots. It uses $k+1$ degrees of freedom for the non-linear effect. Knot placement is typically at quantiles of X (3 knots: 10/50/90; 4 knots: 5/35/65/95; 5 knots: 5/27.5/50/72.5/95 percentiles). Three to five knots are adequate for most biomedical applications; more invites over-fitting.

Restricted cubic splines (RCS) impose linearity beyond the outer knots, preventing the polynomial from curling away in the tails — useful for clinical reporting where extrapolation into low-data regions is a real concern.

7.48.4 Assumptions

Standard regression assumptions on residuals; residuals vs. fitted should show no remaining structure after the spline is included; the chosen number and placement of knots are adequate for the underlying functional form.

7.48.5 R Implementation

```r
library(splines); library(rms)

set.seed(2026)
x <- runif(200, 0, 10)
y <- 2 + sin(x) * 2 + rnorm(200, 0, 0.5)

# Natural cubic spline via splines::ns
fit_ns <- lm(y ~ ns(x, df = 4))
summary(fit_ns)

# Visualise
plot(x, y, pch = 20)
xg <- seq(0, 10, length.out = 200)
lines(xg, predict(fit_ns, newdata = data.frame(x = xg)),
      col = "#2A9D8F", lwd = 2)

# rms::rcs for restricted cubic splines with plot
dd <- datadist(data.frame(x, y)); options(datadist = "dd")
fit_rcs <- ols(y ~ rcs(x, 4))
plot(Predict(fit_rcs, x = seq(0, 10, 0.2)))
```

7.48.6 Output & Results

`splines::ns()` and `rms::rcs()` produce basis matrices that enter `lm()` like any other predictor. The resulting fit object has the usual summary, coefficients (which are not individually interpretable), and predictions. Visualising the partial effect of X on Y via `predict()` or `rms::Predict()` is the primary inferential output.

7.48.7 Interpretation

A reporting sentence: "A natural cubic spline with 4 degrees of freedom captured the underlying sinusoidal pattern of y on x ($R^2 = 0.92$); a likelihood-ratio test against the linear-only model strongly rejected linearity ($\chi^2_3 = 287$, $p < 0.001$)." Always test for non-linearity formally and report the test result; a smooth curve in a plot is not enough.

7.48.8 Practical Tips

- Use 3–5 knots for most applications; more risks overfitting and hard-to-interpret partial effects.
- Restricted cubic splines (`rms::rcs`) constrain the tails to be linear, avoiding the wild extrapolation that natural cubic splines can produce when knots are sparse.

- Test for non-linearity with a likelihood-ratio test of the spline model against a linear-only specification; report the test result alongside any non-linear visualisation.
- Spline coefficients are not individually interpretable; communicate results via `Predict()` plots or `ggeffects::ggpredict()`.
- For multiple continuous predictors with potential non-linear effects, consider GAMs (`mgcv::gam`) which automate smoothness selection.
- For survival or logistic regression, the same `rcs()` syntax inside `cph()` or `lrm()` extends the spline machinery to the relevant likelihood.

7.48.9 R Packages Used

`splines` for `ns()` and `bs()` basis functions; `rms` for `rcs()`, `Predict()`, and `datadist`-aware regression workflows; `mgcv` for GAMs as the more general alternative; `ggeffects` for predicted-effect visualisation; `Hmisc::rcspline.eval()` for low-level RCS basis construction.

7.48.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.48.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.49 Stepwise Regression: Pitfalls

7.49.1 Introduction

Stepwise regression — forward, backward, or bidirectional automated selection of predictors based on p-values, AIC, or BIC — was the workhorse of model selection for decades and remains widely used despite extensive criticism. Frank Harrell's *Regression Modeling Strategies*, the TRIPOD guideline for clinical prediction models, and a long methodological literature have documented serious statistical pathologies that make stepwise reporting at best misleading and at worst actively wrong. Modern practice increasingly substitutes penalised regression (lasso, elastic net) and pre-specified predictor sets.

7.49.2 Prerequisites

A working understanding of multiple regression, model selection, and the bias-variance trade-off.

7.49.3 Theory

The well-documented problems with stepwise selection:

- **Selection bias**: R^2 is inflated and coefficients are biased away from zero because selection and testing are done on the same data.
- **Invalid p-values**: the final model's "significant" predictors include false positives at a rate far above the nominal α; the more candidate predictors, the worse this gets.
- **Instability**: small data perturbations (a single observation, a different random sample) change the selected predictor set substantially.
- **Reduced out-of-sample performance**: stepwise-selected models often predict *worse* than pre-specified or penalised alternatives.
- **Forward / backward / bidirectional give different answers**: the procedure is path-dependent.

Better alternatives include pre-specified predictor sets based on substantive theory, penalised regression with cross-validated tuning (lasso, elastic net), and cross-validated model comparison rather than naive selection-on-fit-statistics.

7.49.4 Assumptions

Standard regression framework; the issues are with the selection procedure, not the underlying model.

7.49.5 R Implementation

```
library(MASS)
```

```r
# Stepwise (for illustration; not recommended in practice)
full <- lm(mpg ~ ., data = mtcars)
step_model <- stepAIC(full, direction = "both", trace = FALSE)
summary(step_model)

# Better: penalised regression
library(glmnet)
X <- as.matrix(mtcars[, -1])
y <- mtcars$mpg
cvfit <- cv.glmnet(X, y, alpha = 1)
coef(cvfit, s = "lambda.min")
```

7.49.6 Output & Results

`stepAIC()` selects a subset based on AIC; reporting the resulting coefficients with naive standard errors and p-values produces invalid inference. Lasso with cross-validated λ provides a principled alternative with shrunken coefficients and a defensible variable-selection mechanism.

7.49.7 Interpretation

A reporting sentence: "We pre-specified the predictor set based on prior literature; stepwise selection was not used because of its known instability and inflated Type I error. For exploratory secondary analyses, we report cross-validated lasso results with the selected variables and CV mean squared error, not naive p-values." This phrasing places the responsible alternative in the methods section explicitly.

7.49.8 Practical Tips

- Avoid reporting naive t-statistics and p-values from stepwise-selected models; they are biased and miscalibrated.
- Use stepwise procedures only for exploratory purposes, never for confirmatory inference.
- Pre-specify the predictor set based on substantive theory; this is the strongest defence against selection-on-significance.
- For high-dimensional or correlated predictors, ridge, lasso, and elastic net handle the variable-selection problem with proper regularisation; cross-validate to choose the penalty.
- The TRIPOD reporting guideline for clinical prediction models specifically discourages stepwise selection; reviewers in clinical journals increasingly enforce this.
- If you must report stepwise results, do so honestly with caveats: report the selection procedure, the candidate predictor set, and a cross-validated estimate of out-of-sample performance.

7.49.9 R Packages Used

`MASS::stepAIC()` and base R `step()` for stepwise (with caveats); `glmnet` for canonical lasso and elastic-net alternatives; `caret` and `tidymodels` for cross-validated model comparison; `selectiveInference` for properly adjusted post-selection inference when selection is unavoidable.

7.49.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.49.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.50 Cox Regression as Regression

7.50.1 Introduction

Cox proportional-hazards regression models time-to-event outcomes with right censoring and is the de facto standard regression tool for survival analysis. This page gives a regression-modelling-focused summary; the Survival Analysis section contains a more detailed treatment with diagnostics, residuals, and stratification. The headline insight is that Cox is a regression in the same family as GLMs — coefficients, standard errors, confidence intervals, and likelihood-ratio

tests all behave familiarly — but the response is a `Surv` object encoding both event time and censoring status.

7.50.2 Prerequisites

A working understanding of survival analysis, the hazard function, censoring, and the partial-likelihood approach to inference.

7.50.3 Theory

The Cox model writes the hazard as

$$h(t \mid X) = h_0(t) \exp(X^\top \beta),$$

with the baseline hazard $h_0(t)$ left unspecified and the partial likelihood providing $\hat{\beta}$ without estimating it. Exponentiated coefficients are hazard ratios; a $\hat{\beta}_j = 0.02$ for age corresponds to a 2 % multiplicative increase in hazard per year of age, $\exp(0.02) \approx 1.02$.

The proportional-hazards assumption — that hazard ratios between any two covariate patterns are constant over time — is the model's defining restriction. When it fails, options include stratification, time-varying coefficients via `tt()` or `survSplit()`, or switching to AFT or RMST-based summaries.

7.50.4 Assumptions

Proportional hazards across covariate levels (test via Schoenfeld residuals), independent non-informative censoring, no unmodelled time-varying covariate effects, and adequate events per variable for stable estimation (rule of thumb: 10–15 events per regression coefficient).

7.50.5 R Implementation

```r
library(survival); library(broom)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
tidy(fit, conf.int = TRUE, exponentiate = TRUE)

# PH check
cox.zph(fit)
```

7.50.6 Output & Results

`tidy()` from `broom` returns a tibble of hazard ratios, confidence intervals, and Wald p-values. `summary(fit)` adds the likelihood-ratio test, score test, and Wald

test for the joint null. `cox.zph()` reports the proportional-hazards diagnostic for each covariate plus a global test.

7.50.7 Interpretation

A reporting sentence: "A multivariable Cox model adjusted for age, sex, and ECOG performance status estimated that each additional year of age increased the hazard of death by 2 % (HR 1.02, 95 % CI 1.00 to 1.04, $p = 0.04$); female sex was associated with a 40 % lower hazard (HR 0.60, 95 % CI 0.42 to 0.85, $p = 0.004$). The proportional-hazards assumption was supported (global Schoenfeld $p = 0.55$)." Always pair coefficient reporting with PH diagnostics.

7.50.8 Practical Tips

- Always run `cox.zph()` after fitting; reporting Cox HRs without PH diagnostics is no longer acceptable in clinical-trial publications.
- Use Efron's tie-handling method (`ties="efron"`) for tied event times; the default in older software is Breslow's, which is less accurate.
- Report HRs with 95 % CIs alongside p-values; a CI that includes 1.0 is more informative than a marginal p-value alone.
- For time-varying covariates, use counting-process format (`Surv(tstart,tstop,event)`) and cluster standard errors on subject id.
- For continuous covariates with non-linear effects, add restricted cubic splines (`rms::rcs(x,4)`) rather than dichotomising.
- See the Survival Analysis section's dedicated Cox tutorial for extended coverage of residuals, stratification, and influence diagnostics.

7.50.9 R Packages Used

`survival` for `coxph()`, `Surv()`, `cox.zph()`; `broom` for `tidy()` and `glance()` summaries; `survminer` for ggplot-style diagnostics; `rms::cph()` for an alternative interface with built-in spline and validation tools.

7.50.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.50.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring

- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.51 Tobit Regression

7.51.1 Introduction

Tobit regression handles outcomes that are observed at the boundary value for cases that fall outside a censoring threshold. Income top-coded at $200,000 (right-censored at the cap), laboratory values below the limit of detection (left-censored at the LOD), drug refill rates that cap at 100 %, all share the same structure: a real underlying value exists, but the observation records the censoring threshold rather than the true value. OLS on the observed outcome is biased toward zero in such cases; Tobit recovers the latent-model coefficients via the appropriate Normal likelihood.

7.51.2 Prerequisites

A working understanding of linear regression, the difference between censoring and truncation, and the Normal distribution as the latent error model.

7.51.3 Theory

For latent $Y_i^* = X_i^\top \beta + \varepsilon_i$ with $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$, the observed value is $Y_i = \max(L, Y_i^*)$ for left-censoring or $Y_i = \min(U, Y_i^*)$ for right-censoring. Maximum-likelihood estimation combines two contributions:

$$L(\beta, \sigma) = \prod_{i \notin \text{cens}} \phi\Big(\frac{Y_i - X_i^\top \beta}{\sigma}\Big) \cdot \prod_{i \in \text{cens}} \Phi\Big(\frac{L - X_i^\top \beta}{\sigma}\Big).$$

The first product is the standard Normal likelihood for non-censored observations; the second is the probability of censoring for boundary observations. The

Tobit model assumes the same coefficients govern both the threshold decision and the observed values; two-part models relax this when those processes differ.

7.51.4 Assumptions

Latent outcome Normally distributed; censoring threshold known and the same for all observations (or known per observation); standard linear-model assumptions on the latent regression.

7.51.5 R Implementation

```
library(AER)

set.seed(2026)
x <- rnorm(200)
y_star <- 2 + 1.5 * x + rnorm(200)
y <- pmax(y_star, 0)    # left-censored at 0

fit <- tobit(y ~ x, left = 0, right = Inf)
summary(fit)

# Compare with OLS on observed y
coef(lm(y ~ x))
```

7.51.6 Output & Results

`tobit()` returns coefficient estimates on the latent scale, recovering the data-generating parameters. OLS on the same observed (censored) data attenuates the slope toward zero, illustrating the bias from ignoring the censoring structure.

7.51.7 Interpretation

A reporting sentence: "Tobit regression with left-censoring at 0 estimated the latent-scale slope at 1.47 (95 % CI 1.19 to 1.75, $p < 0.001$), close to the true 1.5; OLS on the same data gave 0.98 (95 % CI 0.78 to 1.18), biased toward zero by the unaccounted censoring." Always describe the censoring rule and its threshold in methods.

7.51.8 Practical Tips

- Tobit assumes the same coefficients govern both the censoring and observed values; if these processes differ (some observations are structurally at the boundary, others are observed because of underlying behaviour), use a two-part / hurdle model instead.
- For right-censored time-to-event data, use survival regression (Cox, parametric AFT), not Tobit; the censoring mechanism differs.

- Check Normality of the latent residuals via quantile residuals (`fitdistrplus` for diagnostics); Tobit is sensitive to non-Normal latent errors.
- Tobit is common in economics (income, expenditure); biomedicine more often uses two-part models for detection-limit data because the "below LOD" decision is biologically distinct from the actual concentration.
- For mixed-effects censored outcomes, `survreg(...,dist="gaussian")` extended with random effects via `survival` workarounds, or `crch` for heteroscedastic censored regression.
- Predictions can target the censored observation (`type="response"`) or the latent value (`type="linear"`); choose based on what the question requires.

7.51.9 R Packages Used

`AER::tobit()` for the canonical Tobit implementation; `survival::survreg()` with `Surv(...,type="left")` as an alternative formulation; `VGAM::vglm()` with `family=tobit()` for richer specifications; `crch` for censored regression with heteroscedastic errors; `pscl::hurdle()` for hurdle alternatives when the threshold decision is distinct from the observed values.

7.51.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.51.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.52 Truncated Regression

7.52.1 Introduction

Truncated regression applies when observations outside a range are absent entirely from the dataset rather than being censored at the threshold. The distinction matters: censored data record a value at the threshold for observations that fall outside, while truncated data simply omit them. Examples: a study limited to hypertensive patients (no normotensive observations), an income survey of households above a threshold (no low-income data), a registry of high-grade tumours (no low-grade entries). Ordinary least squares on truncated data is biased because the conditional mean inside the truncation region is non-linear, even when the latent model is linear.

7.52.2 Prerequisites

A working understanding of linear regression, the difference between censoring and truncation, and the truncated-Normal distribution.

7.52.3 Theory

For latent $Y^* = X^\top \beta + \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, truncation at L from below means we observe Y only when $Y^* > L$. The conditional density on the truncated region is

$$f(y \mid x, Y > L) = \frac{\phi((y - x^\top \beta)/\sigma)/\sigma}{1 - \Phi((L - x^\top \beta)/\sigma)}.$$

The conditional mean inside the truncation region depends non-linearly on x via the truncation point, so OLS coefficients are attenuated. Maximum-likelihood estimation under the correct truncated-Normal density recovers the latent-model coefficients. The corresponding model with covariate-dependent truncation rules is the Heckman selection model (`sampleSelection::selection`).

7.52.4 Assumptions

A known truncation threshold (or a known truncation rule), Normal latent errors, correctly specified latent linear model. Heavy truncation makes inference fragile because few observations remain.

7.52.5 R Implementation

```
library(truncreg)

set.seed(2026)
x_all <- rnorm(300)
```

```
y_all <- 2 + 1.5 * x_all + rnorm(300)
keep  <- y_all > 0                # truncation: keep only y > 0
x <- x_all[keep]; y <- y_all[keep]

fit_trunc <- truncreg(y ~ x, point = 0, direction = "left")
summary(fit_trunc)

# Biased OLS
coef(lm(y ~ x))
```

7.52.6 Output & Results

`truncreg()` returns coefficient estimates that recover the latent-model parameters (close to the data-generating values 2 and 1.5). Naive OLS on the same truncated data attenuates the slope toward zero, illustrating the bias.

7.52.7 Interpretation

A reporting sentence: "Truncated regression accounting for the selection rule ($y > 0$) recovered the slope at 1.48 (95 % CI 1.20 to 1.76), close to the data-generating value of 1.5; naive OLS on the same truncated sample produced an attenuated slope of 0.91 (95 % CI 0.65 to 1.17), demonstrating the consequence of ignoring truncation." Always check whether observations were dropped before reaching the analysis dataset; truncation in the data-collection process is invisible to standard regression diagnostics.

7.52.8 Practical Tips

- Truncation is stronger than censoring: truncated values are unobserved, censored values are observed at the boundary. Use the right model for the data structure.
- For covariate-dependent truncation rules, Heckman two-step or `sampleSelection::selection()` jointly models the selection and outcome equations.
- Truncated regression is sensitive to the Normality assumption; check residuals on the truncated scale and consider semi-parametric alternatives if Normality fails.
- For heavily truncated samples (most of the latent distribution unobserved), inference becomes fragile; report the proportion of latent observations retained.
- Bayesian alternatives with informative priors stabilise truncated fits when the data alone are weak.
- Confirm the truncation point (often a study design parameter) before fitting; misstating L produces systematically biased estimates.

7.52.9 R Packages Used

`truncreg` for the canonical truncated-Normal regression; `sampleSelection` for Heckman two-step and ML-based sample-selection models; `crch` for censored / truncated regression with heteroscedastic errors; `gamlss` for truncated outcomes within a richer distributional-regression framework.

7.52.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.52.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.53 Zero-Inflated Models

7.53.1 Introduction

Zero-inflated models handle count data that contain more zeros than a Poisson or negative binomial distribution would predict — often called "structural zeros" because they arise from a process distinct from the count-generating process. The model is a mixture: one component decides whether each observation is a structural zero (always zero) or a sampling unit (could have been any non-negative count); the other component generates the count for the sampling units.

Insurance claims, healthcare-utilisation counts, ecological abundance with absent species — all share this structure.

7.53.2 Prerequisites

A working understanding of Poisson and negative-binomial regression, logistic regression, and the conceptual distinction between mixture models and two-part hurdle models.

7.53.3 Theory

The zero-inflated mixture density is

$$\mathrm{P}(Y = 0) = \pi + (1 - \pi) f(0), \qquad \mathrm{P}(Y = y) = (1 - \pi) f(y) \text{ for } y > 0,$$

where π is the structural-zero probability (logistic regression on covariates) and f is the count distribution (Poisson or NB). The two sub-models have their own linear predictors and can use different covariate sets; observed zeros come from both the structural-zero process *and* the count-process zero.

The contrast with hurdle models is conceptual: zero-inflated treats some zeros as structurally always-zero, while hurdle treats *all* zeros as arising from one Bernoulli decision and the count distribution is truncated to be strictly positive.

7.53.4 Assumptions

Count outcome, two distinct processes generating zeros and non-zero counts, independent observations, and the chosen count distribution (Poisson or NB) appropriate for the non-structural component.

7.53.5 R Implementation

```
library(pscl); library(glmmTMB)

set.seed(2026)
d <- data.frame(x = rnorm(300))
# Simulate with 40% structural zeros
pi_zero <- 0.4
d$y <- ifelse(rbinom(300, 1, pi_zero) == 1, 0,
              rpois(300, lambda = exp(0.5 + 0.6 * d$x)))

# Zero-inflated Poisson
fit_zip <- zeroinfl(y ~ x | x, data = d)
summary(fit_zip)

# Zero-inflated negative binomial
```

```r
fit_zinb <- zeroinfl(y ~ x | x, data = d, dist = "negbin")

AIC(fit_zip, fit_zinb)

# glmmTMB version
fit_tmb <- glmmTMB(y ~ x, ziformula = ~ x, data = d, family = poisson)
```

7.53.6 Output & Results

`summary(fit_zip)` returns two coefficient blocks: the count-component coefficients on the log-rate scale and the zero-inflation coefficients on the logit scale. `AIC()` compares ZIP against ZINB; substantially overdispersed counts favour ZINB strongly.

7.53.7 Interpretation

A reporting sentence: "A zero-inflated negative binomial model decomposed the observed zeros into structural and sampling sources: the count component estimated a rate ratio of 1.75 per unit increase in x (95 % CI 1.40 to 2.18, $p < 0.001$); the zero-inflation component estimated that a 1-unit increase in x reduced the odds of being a structural zero by 45 % (OR 0.55, 95 % CI 0.36 to 0.84, $p = 0.005$). ZINB was favoured over ZIP by AIC ($\Delta = 12$)." Always report both components and the framework rationale.

7.53.8 Practical Tips

- The Vuong test (`pscl::vuong`) compares zero-inflated to regular Poisson/NB but is often inconclusive; AIC and BIC are more reliable.
- Different covariates can drive the count and zero-inflation components; use `y~x_count|x_zero` to specify them separately.
- Hurdle models (`pscl::hurdle`) are an alternative when the scientific story is "did anything happen, and if so how much"; ZIP/ZINB fits "are some zeros structural while others are sampling".
- Overdispersed counts plus excess zeros typically warrant ZINB; pure overdispersion without structural zeros is just NB.
- Interpretation is cleaner in hurdle models when the two-stage scientific story is clear; ZIP/ZINB is preferable when zeros come from a genuine mixture.
- For random-effects extensions, `glmmTMB(...,ziformula=~...)` adds clustered or hierarchical structure to either component.

7.53.9 R Packages Used

`pscl::zeroinfl()` and `hurdle()` for canonical implementations; `glmmTMB` for mixed-effects zero-inflated and hurdle models with cleaner overdispersion han-

dling; `countreg` for additional count-model variants and rootogram diagnostics; `DHARMa` for residual diagnostics across zero-inflated families.

7.53.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

7.53.11 See also — labs in this chapter

- Correlation vs regression; Model-I/II regression
- Simple linear regression
- Multiple regression: confounding, interaction, centring
- Diagnostics: residuals, QQ, leverage, Cook's distance, VIF
- Robust regression, weighted LS, HC (sandwich) standard errors
- One-way ANOVA and contrasts (emmeans)
- Two-way / factorial ANOVA with interaction
- RCBD and repeated measures → mixed models
- GAMs with mgcv
- Non-linear regression with nls
- Logistic regression (binomial GLM)
- ANCOVA in RCTs (baseline adjustment)
- Ordinal and multinomial regression
- Poisson and negative-binomial regression
- Calibration, discrimination, ROC/AUC, Brier score
- Dichotomisation, change scores, regression to the mean
- Decision curves, NRI, IDI

7.54 Week 3, Session 2 — ANCOVA in RCTs (baseline adjustment)

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.54.1 Learning objectives

- Choose between change scores, post-only, and ANCOVA in an RCT analysis plan.
- Fit an ANCOVA model and read the treatment effect as an adjusted-mean difference.
- Show that ANCOVA is more efficient than the change-score approach when baseline and follow-up are correlated.

7.54.2 Prerequisites

Linear regression; one-way ANOVA.

7.54.3 Background

In a randomised trial with a continuous outcome measured at baseline and follow-up, three analyses are common: compare the post-intervention means, compare the change scores, or use analysis of covariance (ANCOVA) with baseline as a covariate. Randomisation ensures all three are unbiased in expectation, but they are not equally efficient.

ANCOVA weights the baseline by its observed correlation with the follow-up and is therefore always at least as efficient as the change-score approach, and much more efficient when that correlation is moderate. Change scores are equivalent to ANCOVA with the baseline slope constrained to 1; post-only is ANCOVA with the slope constrained to 0. When neither constraint matches the data, efficiency is lost.

Regression to the mean is the mechanical reason ANCOVA wins. Patients with unusually high baselines tend to have lower follow-ups whether they are treated or not; ANCOVA uses that pattern, change-score analysis pretends it does not exist.

7.54.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.54.5 1. Hypothesis

Simulate a trial with baseline and follow-up correlated at r = 0.6, then compare the three analyses.

7.54.6 2. Visualise

```
n <- 200
baseline <- rnorm(n, 140, 15)
arm <- rep(c("placebo", "active"), each = n / 2)
# true treatment effect on follow-up: -5
followup <- 0.6 * (baseline - 140) + 140 +
  ifelse(arm == "active", -5, 0) + rnorm(n, 0, 12)
dat <- tibble(baseline, followup, arm,
              change = followup - baseline) |>
```

```
  mutate(arm = factor(arm, levels = c("placebo", "active")))

ggplot(dat, aes(baseline, followup, colour = arm)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "Baseline", y = "Follow-up")
```

7.54.7 3. Assumptions

Randomisation; linear relationship between baseline and follow-up; equal slopes across arms (tested with an interaction term).

```
fit_int <- lm(followup ~ arm * baseline, data = dat)
tidy(fit_int) |> filter(term == "armactive:baseline")
```

If the interaction is small, pool the slopes in the ANCOVA.

7.54.8 4. Conduct

```
fit_post   <- lm(followup ~ arm, data = dat)
fit_change <- lm(change ~ arm, data = dat)
fit_ancova <- lm(followup ~ arm + baseline, data = dat)

bind_rows(
  tidy(fit_post, conf.int = TRUE) |> mutate(model = "post only"),
  tidy(fit_change, conf.int = TRUE) |> mutate(model = "change"),
  tidy(fit_ancova, conf.int = TRUE) |> mutate(model = "ANCOVA")
) |>
  filter(term == "armactive") |>
  dplyr::select(model, estimate, std.error, conf.low, conf.high)
```

The three estimates should all be around -5; the ANCOVA standard error should be the smallest.

7.54.9 5. Concluding statement

> In a simulated trial ($n =$ `nrow(dat)`), ANCOVA estimated the treatment effect at `round(coef(fit_ancova)["armactive"],2)` units (95% CI `round(confint(fit_ancova)["armactive",1],2)` to `round(confint(fit_ancova)["armactive",2],2)`) with a smaller standard error than either the post-only or change-score analyses. ANCOVA is the prespecified primary analysis in most continuous-outcome RCTs.

7.54.10 Common pitfalls

- Adjusting for post-baseline covariates (not covered here, but a classic trap).
- Using a change score when baseline and follow-up are strongly correlated.
- Dropping observations with missing baselines instead of imputing or pre-specifying a policy.

7.54.11 Further reading

- Senn S (2006), *Change from baseline and analysis of covariance…*
- Vickers AJ, Altman DG (2001), *Analysing controlled trials with…*
- ICH E9 (R1) Addendum on Estimands.

7.54.12 Session info

```
sessionInfo()
```

7.54.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.55 Week 3, Session 5 — Calibration, discrimination, ROC/AUC, Brier score

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.55.1 Learning objectives

- Separate discrimination (AUC) from calibration and from overall performance (Brier score).
- Draw an ROC curve with `pROC` and a calibration plot by binning predicted probabilities.
- Understand that a high AUC does not imply well-calibrated probabilities.

7.55.2 Prerequisites

Logistic regression.

7.55.3 Background

A risk prediction model should do three things well: separate events from non-events (discrimination), produce predicted probabilities that match observed

frequencies (calibration), and balance both in a single summary of overall performance (the Brier score). A model can have high discrimination and poor calibration — for instance, if the predicted probabilities are systematically too extreme — and a model with perfect calibration can still be useless if it cannot separate cases from non-cases.

ROC analysis walks the decision threshold across all possible values and plots sensitivity against 1 − specificity. The area under the curve (AUC) is the probability that a randomly chosen case receives a higher predicted probability than a randomly chosen non-case. The Brier score is the mean squared error between predicted probabilities and the 0/1 outcome; a perfect model has a Brier of 0 and the no-skill model has Brier equal to the variance of the outcome.

Calibration plots group predicted probabilities into bins and plot the observed event rate in each bin against the mean predicted probability. A 45° reference line is perfect calibration. Smoothed (LOESS) calibration curves are an alternative in larger samples.

7.55.4 Setup

```
library(tidyverse)
library(broom)
library(MASS)
library(pROC)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.55.5 1. Hypothesis

Using `MASS::Pima.tr` (train) and `MASS::Pima.te` (test), fit a logistic regression and evaluate its discrimination, calibration, and Brier score.

7.55.6 2. Visualise

```
train <- as_tibble(Pima.tr) |> mutate(y = as.integer(type == "Yes"))
test  <- as_tibble(Pima.te) |> mutate(y = as.integer(type == "Yes"))

fit <- glm(y ~ glu + bmi + age + ped, data = train, family = binomial)
test$p <- predict(fit, test, type = "response")

ggplot(test, aes(p, fill = factor(y))) +
  geom_histogram(position = "identity", alpha = 0.5, bins = 30) +
  labs(x = "Predicted probability", y = "Count", fill = "Diabetes")
```

7.55.7 3. Assumptions

The usual logistic-regression assumptions for the training fit; evaluation is on held-out data.

7.55.8 4. Conduct

Discrimination:

```
roc_obj <- roc(test$y, test$p, quiet = TRUE)
auc(roc_obj)
ci.auc(roc_obj)

plot(roc_obj, main = sprintf("ROC — AUC = %.2f", auc(roc_obj)))
```

Calibration (decile binning):

```
cal <- test |>
  mutate(bin = ntile(p, 10)) |>
  group_by(bin) |>
  summarise(predicted = mean(p), observed = mean(y), n = n())

ggplot(cal, aes(predicted, observed)) +
  geom_abline(linetype = 2, colour = "grey50") +
  geom_point(size = 3) +
  geom_line() +
  coord_equal() +
  labs(x = "Predicted probability (bin mean)",
       y = "Observed event rate")
```

Brier score:

```
brier <- mean((test$p - test$y)^2)
brier
```

7.55.9 5. Concluding statement

> On the held-out test set ($n =$ `nrow(test)`), the four-predictor logistic model achieved AUC = `round(as.numeric(auc(roc_obj)),2)` (95% CI: `round(ci.auc(roc_obj)[1],2)` to `round(ci.auc(roc_obj)[3],2)`) and a Brier score of `round(brier,3)`. Calibration by decile was close to the 45° line.

7.55.10 Common pitfalls

- Reporting AUC on the training set without cross-validation.
- Using accuracy at a 0.5 threshold when the prevalence is not 0.5.
- Treating a single deciles-calibration plot as a formal test.

7.55.11 Further reading

- Steyerberg EW. *Clinical Prediction Models*.
- Harrell FE. *Regression Modeling Strategies*, ch. 10.
- Van Calster B et al. (2019), *Calibration: the Achilles heel…*

7.55.12 Session info

```
sessionInfo()
```

7.55.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.56 Week 1, Session 1 — Correlation vs regression; Model-I/II regression

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.56.1 Learning objectives

- Distinguish correlation from regression and say when each is appropriate.
- Fit Model-I (OLS) and Model-II (major-axis / SMA) lines and interpret the difference.
- Report an association with a slope, interval, and p-value rather than with a correlation alone.

7.56.2 Prerequisites

Hypothesis testing and basic R.

7.56.3 Background

Correlation and regression answer related but different questions. Correlation asks *how tightly do two variables track each other?* and is symmetric in X and Y. Regression asks *how does the mean of Y change with X?* and treats the two axes asymmetrically: X is assumed to be measured without error and Y is modelled. This asymmetry matters for prediction (which needs a regression line) and for comparing slopes across groups (which also needs regression), but it becomes problematic when both variables carry measurement error and neither can be declared the predictor.

Model-II regression — of which the standardised major axis (SMA) and reduced major axis are the common flavours — was developed for exactly this case. When both X and Y are measured with error, ordinary least squares (OLS) is biased toward zero, because it attributes all residual variation to Y. Model-II methods acknowledge error in both directions and give a less biased estimate of the true functional relationship. The price is a weaker theoretical grounding for inference and the need to report the method used.

Model-II regression has a respectable history in allometry and in method-comparison studies, where the instruments on both axes are imperfect. In a clinical lab context you will often see Deming regression for comparing two assays; SMA and Deming are close cousins under reasonable assumptions. The choice should follow the science: *is X a fixed, controlled input, or a noisy measurement of something?*

7.56.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.56.5 1. Hypothesis

We will simulate two noisy measurements of the same underlying quantity — say, body size measured by two instruments — and ask how OLS and SMA differ.

Null hypothesis: the two measurements are uncorrelated. Working alternative: they are positively associated, with a slope near 1 if the instruments agree.

7.56.6 2. Visualise

```
n <- 200
true_size <- rnorm(n, mean = 50, sd = 10)
x <- true_size + rnorm(n, 0, 3)
y <- true_size + rnorm(n, 0, 3)
dat <- tibble(x, y)

ggplot(dat, aes(x, y)) +
  geom_point(alpha = 0.6, colour = "grey30") +
  labs(x = "Instrument 1", y = "Instrument 2")
```

A cloud of points that climbs from lower-left to upper-right. Both instruments carry error, so neither deserves to be called the truth.

7.56.7 3. Assumptions

OLS assumes X is measured without error and residuals in Y are approximately normal with constant variance. SMA assumes errors in both variables are comparable in scale. Both assume linearity.

```
fit_ols <- lm(y ~ x, data = dat)
par(mfrow = c(1, 2))
plot(fit_ols, which = c(1, 2))
par(mfrow = c(1, 1))
```

Residuals look patternless; the QQ plot is close to the diagonal. The OLS diagnostics are fine; they just do not capture the fact that X is also noisy.

7.56.8 4. Conduct

```
cor.test(dat$x, dat$y)

fit_ols <- lm(y ~ x, data = dat)
tidy(fit_ols, conf.int = TRUE)
```

Now a hand-coded SMA slope. The SMA slope is the ratio of standard deviations, signed by the correlation:

```
sma_slope <- function(x, y) sign(cor(x, y)) * sd(y) / sd(x)
sma_intercept <- function(x, y) mean(y) - sma_slope(x, y) * mean(x)

b_sma <- sma_slope(dat$x, dat$y)
a_sma <- sma_intercept(dat$x, dat$y)
b_ols <- coef(fit_ols)[2]

c(ols_slope = unname(b_ols), sma_slope = b_sma)
```

```
ggplot(dat, aes(x, y)) +
  geom_point(alpha = 0.5, colour = "grey40") +
  geom_abline(intercept = coef(fit_ols)[1], slope = b_ols,
              colour = "steelblue", linewidth = 0.9) +
  geom_abline(intercept = a_sma, slope = b_sma,
              colour = "firebrick", linewidth = 0.9, linetype = 2) +
  labs(x = "Instrument 1", y = "Instrument 2",
       title = "OLS (blue) vs SMA (red)")
```

The SMA slope is closer to 1. OLS is attenuated because it treats X as fixed when it is not.

7.56.9 5. Concluding statement

In simulated data ($n = 200$), instrument 2 increased with instrument 1 (Pearson $r =$ `round(cor(dat$x,dat$y),2)`; OLS slope `round(unname(b_ols),2)`; SMA slope `round(b_sma,2)`). Because both instruments carry measurement error, we report the SMA slope as the functional relationship and note the OLS slope for comparison.

7.56.10 Common pitfalls

- Reporting a correlation when a slope is what the reader needs.
- Using OLS to compare two noisy measurements and concluding the slope differs from 1 when the attenuation is an artefact.
- Forgetting that r has no units and therefore cannot substitute for an effect size on the measurement scale.

7.56.11 Further reading

- Warton DI et al. (2006), *Bivariate line-fitting methods for allometry.*
- Legendre P, Legendre L. *Numerical Ecology*, ch. 10.
- Altman DG, Bland JM (1983), *Measurement in Medicine.*

7.56.12 Session info

```
sessionInfo()
```

7.56.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.57 Week 4, Session 4 — Decision curves, NRI, IDI

Workflow lab: Goal → Approach → Execution → Check → Report.

7.57.1 Learning objectives

- Compute net benefit and draw a decision-curve plot for two competing risk models.
- Compute category-free NRI and IDI and read what each adds beyond ROC/AUC.
- Decide which of the three to report in which setting.

7.57.2 Prerequisites

Calibration and discrimination, plus a working logistic-regression fit.

7.57.3 Background

A prediction model is useful when its decisions, weighted by clinical consequence, beat any alternative policy. Decision-curve analysis makes that comparison explicit: across a range of threshold probabilities, it plots *net benefit* — true positives minus false positives, weighted by the threshold — against the "treat all" and "treat none" default policies. A model whose decision curve dominates the defaults across a clinically plausible range of thresholds is providing decision value.

The net reclassification improvement (NRI) and integrated discrimination improvement (IDI) are category-free alternatives to comparing AUCs. NRI summarises how well the new model moves events upwards and non-events downwards in risk; IDI summarises the mean change in predicted probabilities. Both are noisy in small samples and both are prone to optimistic interpretation, so report them alongside, not instead of, the decision curve.

7.57.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.57.5 1. Goal

Compare a baseline logistic model with a model that adds a new marker, using decision curves, NRI, and IDI.

7.57.6 2. Approach

```
n <- 800
x1 <- rnorm(n)
x2 <- rnorm(n) + 0.3 * x1
lp <- -1 + 0.9 * x1 + 0.6 * x2
y  <- rbinom(n, 1, plogis(lp))
df <- tibble(y = y, x1 = x1, x2 = x2)

base_fit <- glm(y ~ x1,       data = df, family = binomial)
new_fit  <- glm(y ~ x1 + x2,  data = df, family = binomial)
df <- df |>
  mutate(p_base = predict(base_fit, type = "response"),
```

```
        p_new  = predict(new_fit,  type = "response"))
```

7.57.7 3. Execution — decision curve (hand-coded)

```
net_benefit <- function(y, p, thr) {
  n <- length(y)
  pred_pos <- p >= thr
  tp <- sum(pred_pos &  y == 1)
  fp <- sum(pred_pos &  y == 0)
  tp / n - fp / n * (thr / (1 - thr))
}

thresholds <- seq(0.05, 0.6, by = 0.01)
dc <- tibble(
  threshold    = thresholds,
  base_model   = sapply(thresholds, \(t) net_benefit(df$y, df$p_base,
  ↪  t)),
  new_model    = sapply(thresholds, \(t) net_benefit(df$y, df$p_new,  t)),
  treat_all    = sapply(thresholds, \(t)
                        mean(df$y) - (1 - mean(df$y)) * t / (1 - t)),
  treat_none   = 0
) |>
  pivot_longer(-threshold, names_to = "strategy", values_to =
  ↪  "net_benefit")
```

```
ggplot(dc, aes(threshold, net_benefit, colour = strategy)) +
  geom_line(linewidth = 0.8) +
  scale_y_continuous(limits = c(-0.05, max(dc$net_benefit))) +
  labs(x = "Threshold probability", y = "Net benefit",
       colour = NULL)
```

7.57.8 4. Check — NRI and IDI

```
nri <- function(y, p_old, p_new) {
  up   <- mean((p_new > p_old)[y == 1]) - mean((p_new < p_old)[y == 1])
  down <- mean((p_new < p_old)[y == 0]) - mean((p_new > p_old)[y == 0])
  c(events = up, non_events = down, overall = up + down)
}

idi <- function(y, p_old, p_new) {
  ev <- mean((p_new - p_old)[y == 1])
  ne <- mean((p_old - p_new)[y == 0])
  c(events = ev, non_events = ne, overall = ev + ne)
}
```

```
nri(df$y, df$p_base, df$p_new)
idi(df$y, df$p_base, df$p_new)
```

7.57.9 5. Report

> Adding the new marker to a baseline logistic model yielded higher net benefit than the baseline across clinically plausible threshold probabilities. The category-free NRI was positive for events and non-events; the IDI was also positive overall. Absolute magnitudes should be interpreted alongside the decision curve, not in isolation.

7.57.10 Common pitfalls

- Reporting NRI without an interval or decision-curve context.
- Choosing the threshold range post-hoc to maximise apparent gain.
- Forgetting that decision curves assume the cost/benefit ratio implied by the threshold is clinically reasonable.

7.57.11 Further reading

- Vickers AJ, Elkin EB. (2006). *Decision curve analysis: a novel method for evaluating prediction models.* Medical Decision Making.
- Pencina MJ et al. (2008). *Evaluating the added predictive ability of a new marker.*

7.57.12 Session info

```
sessionInfo()
```

7.57.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.58 Week 1, Session 4 — Diagnostics: residuals, QQ, leverage, Cook's distance, VIF

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.58.1 Learning objectives

- Read the four default `plot(lm)` diagnostics and say what each is for.

- Identify leverage and influence separately and combine them using Cook's distance.
- Detect collinearity with VIF and decide when to act on it.

7.58.2 Prerequisites

Sessions 2 and 3 of this week.

7.58.3 Background

Diagnostics are the step in a regression workflow that protects the report from the dataset. A coefficient is only as trustworthy as the assumptions behind it. The four classical plots — residuals vs fitted, QQ, scale-location, and residuals vs leverage — answer four questions: is the mean right, are the errors roughly normal, is the variance constant, and is any single point running the show?

Leverage measures how unusual a point's predictor values are; influence measures how much the fit changes when that point is removed. Cook's distance combines the two. A point can have high leverage without high influence if it sits on the regression line, and it can have high influence without spectacular residuals if its predictors are extreme.

The `performance::check_model()` function wraps many of these ideas in a single call and is increasingly the way teams read diagnostics on screen. `car::vif()` gives the variance inflation factors; a rule of thumb is that values above 5 deserve attention and above 10 demand it, but rules of thumb are no substitute for thinking about which predictors are truly redundant.

7.58.4 Setup

```
library(tidyverse)
library(broom)
library(car)
library(performance)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.58.5 1. Hypothesis

Simulate a regression with mild heteroscedasticity, a high-leverage point, and two correlated predictors. Ask which diagnostics flag which problem.

7.58.6 2. Visualise

```
n <- 150
x1 <- rnorm(n, 0, 1)
```

```
x2 <- x1 + rnorm(n, 0, 0.3)      # x1 and x2 correlated
x3 <- rnorm(n, 0, 1)
y  <- 1 + 2 * x1 - 1.5 * x2 + 0.5 * x3 + rnorm(n, 0, 1 + 0.4 * abs(x1))
# add one influential point
x1[1] <- 5; x2[1] <- 5; y[1] <- 0
dat <- tibble(y, x1, x2, x3)

ggplot(dat, aes(x1, y)) +
  geom_point(alpha = 0.6) +
  labs(x = "x1", y = "y")
```

7.58.7 3. Assumptions

```
fit <- lm(y ~ x1 + x2 + x3, data = dat)
par(mfrow = c(2, 2))
plot(fit)
par(mfrow = c(1, 1))
```

The residuals-vs-fitted plot should show no trend. The QQ plot should track the line. Scale-location should be flat. Residuals-vs-leverage should have no points outside the Cook's-distance contours.

7.58.8 4. Conduct

Influence diagnostics:

```
infl <- augment(fit) |> mutate(row = row_number())
infl |> arrange(desc(.cooksd)) |> head(5) |>
  select(row, .fitted, .resid, .hat, .cooksd)
```

VIF:

```
vif(fit)
```

Performance summary:

```
check_model(fit, check = c("vif", "qq", "outliers", "linearity"))
```

VIF for x1 and x2 should be high; x3 should be fine. The first row (the injected leverage point) should dominate Cook's distance.

7.58.9 5. Concluding statement

> Regression diagnostics identified one high-influence observation (Cook's $D =$ `round(max(infl$.cooksd),2)`) and collinearity between x1 and x2 (VIF $=$ `round(max(vif(fit)),1)`). Refitting without the

influential point or after combining the collinear predictors would be the next step before reporting coefficients.

7.58.10 Common pitfalls

- Treating a single Cook's distance number as a verdict. Always look at the plot.
- Solving a VIF problem by dropping one variable at a time when the underlying issue is that two predictors measure the same thing.
- Declaring the assumptions met because the numbers pass. Plot first.

7.58.11 Further reading

- Fox J, Weisberg S. *An R Companion to Applied Regression*, ch. 8.
- Belsley DA, Kuh E, Welsch RE. *Regression Diagnostics*.
- Lüdecke D et al. (2021), *performance: An R Package for Assessment…*

7.58.12 Session info

```
sessionInfo()
```

7.58.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.59 Week 4, Session 1 — Dichotomisation, change scores, regression to the mean

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.59.1 Learning objectives

- Quantify the power loss of a median split on a continuous predictor.
- Demonstrate regression to the mean in a simulated pre-post design.
- Distinguish apparent improvement due to RTM from a true treatment effect.

7.59.2 Prerequisites

Basic regression; ANCOVA.

7.59.3 Background

Dichotomising a continuous predictor — *high vs low* — loses information in proportion to the variability thrown away. A well-known rule of thumb is that the median split reduces power in a comparison equivalent to throwing away roughly a third of the sample. The only time a dichotomy is defensible is when the clinical decision it feeds into is itself binary, and even then the continuous underlying variable should enter the analysis.

Regression to the mean is a statistical fact about correlated pairs of measurements: participants who score unusually high on a first measurement will, on average, score less extremely on a second measurement of the same quantity, even with no intervention. In a pre-post study this produces apparent *improvement* in the high baseline group and apparent *worsening* in the low baseline group. ANCOVA handles this automatically; a simple comparison of pre and post in the extreme group does not.

RTM is not the same as measurement error, but measurement error is one source of RTM. Any non-perfect correlation between measurements produces it. This is why baseline stratification by a continuous variable must be prespecified — selecting on a single noisy baseline guarantees RTM.

7.59.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.59.5 1. Hypothesis

Two simulations:

1. A continuous predictor truly associated with a continuous outcome; compare the p-value when treated as continuous vs median-split.
2. A pre-post dataset with no intervention; show apparent improvement in the high-baseline group.

7.59.6 2. Visualise

```
n <- 200
x <- rnorm(n)
y <- 0.3 * x + rnorm(n, 0, 1)
dat1 <- tibble(x, y, x_bin = factor(ifelse(x > median(x), "high",
 ↪  "low")))
```

```
ggplot(dat1, aes(x, y)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", colour = "steelblue") +
  labs(x = "x (continuous)", y = "y")
```

7.59.7 3. Assumptions

Independence, approximate normality; standard regression assumptions.

7.59.8 4. Conduct

Power loss from dichotomisation:

```
tidy(lm(y ~ x, data = dat1))     # continuous
tidy(lm(y ~ x_bin, data = dat1)) # median split
```

The p-value for x_bin should be larger than for x; the standard error is larger and the estimate is attenuated.

Regression to the mean:

```
n2 <- 1000
pre  <- rnorm(n2, 100, 15)
post <- 0.7 * (pre - 100) + 100 + rnorm(n2, 0, 11)
dat2 <- tibble(pre, post) |>
  mutate(high_baseline = pre > quantile(pre, 0.8))

dat2 |>
  group_by(high_baseline) |>
  summarise(pre_mean = mean(pre), post_mean = mean(post),
            change = mean(post - pre))
```

The "high baseline" group shows a large negative mean change; the rest of the sample shows a small positive change. No intervention occurred.

```
ggplot(dat2, aes(pre, post)) +
  geom_point(alpha = 0.3) +
  geom_abline(slope = 1, intercept = 0, linetype = 2, colour = "grey50") +
  geom_smooth(method = "lm", colour = "firebrick") +
  labs(x = "Baseline", y = "Follow-up")
```

The fitted line is flatter than the identity line; that flattening is RTM.

7.59.9 5. Concluding statement

> Dichotomising the continuous predictor inflated its p-value from `signif(tidy(lm(y~x,data=dat1))$p.value[2],2)` to `signif(tidy(lm(y~x`

`_bin,data=dat1))$p.value[2],2)` and widened its standard error. In the pre-post simulation with no intervention, the top-baseline quintile showed a mean decline of `round(dat2|>filter(high_baseline)|>summarise(mean(post-pre))|>pull(),1)` units, entirely attributable to regression to the mean.

7.59.10 Common pitfalls

- Dichotomising to avoid thinking about the slope.
- Comparing pre-post within a high-baseline subgroup and calling the decline an effect.
- Using a paired *t*-test on change in a selected subgroup.

7.59.11 Further reading

- MacCallum RC et al. (2002), *On the practice of dichotomization…*
- Altman DG, Royston P (2006), *The cost of dichotomising continuous…*
- Senn S (2011), *Francis Galton and regression to the mean.*

7.59.12 Session info

```
sessionInfo()
```

7.59.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.60 Week 2, Session 4 — GAMs with mgcv

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.60.1 Learning objectives

- Fit a generalised additive model with `mgcv::gam()` and a single smooth term.
- Read effective degrees of freedom as a data-driven measure of flexibility.
- Visualise a smooth with `gratia::draw()` and report its shape in plain language.

7.60.2 Prerequisites

Linear and multiple regression.

7.60.3 Background

Generalised additive models replace the linear term $\beta \cdot \mathrm{X}$ in a regression with a smooth function f(X) estimated from the data. The smooth is regularised so that wiggliness is penalised; the penalty strength is chosen by restricted maximum likelihood or generalised cross-validation. The effective degrees of freedom (edf) of the smooth quantifies how much of that flexibility was used: an edf near 1 means the smooth is essentially a line; an edf of 5 means something meaningfully curved.

GAMs are particularly useful when the *shape* of the relationship is the scientific question. They are also useful as diagnostics for linear models, because plotting a smooth against a predictor can reveal a curve that the linear term would miss.

The `mgcv` package is the reference implementation. `gratia` is a newer graphics layer on top of `mgcv` that produces tidy, ggplot-based plots of smooths, partial effects, and derivatives.

7.60.4 Setup

```
library(tidyverse)
library(mgcv)
library(gratia)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.60.5 1. Hypothesis

Simulate a sine-wave-plus-noise relationship and ask whether a GAM recovers it while a linear model misses it.

7.60.6 2. Visualise

```
n <- 300
x <- runif(n, 0, 4 * pi)
y <- sin(x) + rnorm(n, 0, 0.4)
dat <- tibble(x, y)

ggplot(dat, aes(x, y)) +
  geom_point(alpha = 0.5, colour = "grey30") +
  labs(x = "x", y = "y")
```

7.60.7 3. Assumptions

GAMs assume independent observations and a correctly specified response distribution. The smooth flexibility is chosen automatically; the user chooses the basis and, if relevant, the upper limit on flexibility via `k`.

```
fit_lm  <- lm(y ~ x, data = dat)
fit_gam <- gam(y ~ s(x, bs = "cr", k = 20), data = dat, method = "REML")

plot(fit_gam, residuals = TRUE, pch = 1, shade = TRUE)
gam.check(fit_gam)
```

`gam.check` reports `k-index` and effective df — an indication of whether `k` is large enough.

7.60.8 4. Conduct

```
summary(fit_gam)
AIC(fit_lm, fit_gam)
```

```
draw(fit_gam)
```

7.60.9 5. Concluding statement

> A GAM with a single smooth of x captured the non-linear relationship (edf = `round(summary(fit_gam)$s.table[,"edf"],1)`; approximate $p < 0.001$). A linear model gave a near-zero slope because the underlying function is a full sine wave. The GAM is preferred when the shape, not a scalar slope, is the scientific object.

7.60.10 Common pitfalls

- Setting `k` too small and constraining the smooth.
- Comparing GAMs with AIC as if it were a likelihood-ratio test.
- Reporting only the smooth without showing its plot.

7.60.11 Further reading

- Wood SN. *Generalized Additive Models: An Introduction with R.*
- Pedersen EJ et al. (2019), *Hierarchical generalized additive models.*
- Simpson GL. *gratia* package vignette.

7.60.12 Session info

```
sessionInfo()
```

7.60.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.61 Week 3, Session 1 — Logistic regression (binomial GLM)

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.61.1 Learning objectives

- Fit a logistic regression with `glm(family=binomial)` and translate coefficients into odds ratios.
- Construct a confusion matrix at a chosen probability threshold.
- Report the model with an odds ratio, an interval, and a brief note on discrimination.

7.61.2 Prerequisites

Linear regression material.

7.61.3 Background

Logistic regression models the log-odds of a binary outcome as a linear combination of predictors. The exponent of a coefficient is the odds ratio for a one-unit increase in the corresponding predictor. Because the relationship is linear on the log-odds scale and not the probability scale, a slope of 0.5 log-odds is a larger change in probability near 0.5 than near 0.95.

The model is fit by maximum likelihood with iteratively reweighted least squares. Standard errors are computed from the observed information, and Wald intervals on the log-odds scale are exponentiated to give intervals on the odds ratio. Profile-likelihood intervals are more trustworthy when the sample is small or the predictor is strongly associated with the outcome.

A confusion matrix at a single threshold tells only part of the story; the thresholds used in the clinic may not be the one that maximises accuracy. A later lab covers ROC curves and calibration, which give a threshold-free view of the same model.

7.61.4 Setup

```
library(tidyverse)
library(broom)
```

```
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.61.5 1. Hypothesis

Using `MASS::Pima.tr`, can plasma glucose (`glu`) predict diabetes status (`type`)?

7.61.6 2. Visualise

```
pima <- as_tibble(Pima.tr) |>
  mutate(type = factor(type, levels = c("No", "Yes")))

ggplot(pima, aes(type, glu, fill = type)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = "Diabetes", y = "Plasma glucose") +
  theme(legend.position = "none")
```

7.61.7 3. Assumptions

Binary outcome, independent observations, log-odds linear in the predictors (or handled explicitly with a smooth/transformation).

```
fit <- glm(type ~ glu, data = pima, family = binomial)
# residuals are less informative in GLMs; check for linearity on the
 ↪  logit scale
pima |>
  mutate(p = fitted(fit),
         logit = qlogis(p)) |>
  ggplot(aes(glu, logit)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "loess", se = FALSE, colour = "steelblue") +
  labs(x = "Plasma glucose", y = "Fitted logit")
```

7.61.8 4. Conduct

```
tidy(fit, conf.int = TRUE, exponentiate = TRUE)
glance(fit)
```

```
pred_prob <- predict(fit, type = "response")
pred_class <- ifelse(pred_prob > 0.5, "Yes", "No")
table(pred_class, truth = pima$type)
```

7.61.9 5. Concluding statement

> In `nrow(pima)` women, each 10 mg/dL increase in plasma glucose was associated with an odds ratio for diabetes of `round(exp(10*coef(fit)["glu"]),2)` (95% CI from profile: `round(exp(10*confint(fit)["glu",1]),2)` to `round(exp(10*confint(fit)["glu",2]),2)`). At a 0.5 threshold the classifier had `round(100*mean(pred_class==pima$type),1)`% accuracy; a threshold-free assessment appears in a later lab.

7.61.10 Common pitfalls

- Reporting the coefficient on the log-odds scale without exponentiating.
- Confusing probability and odds.
- Evaluating the model with accuracy at one threshold and nothing else.

7.61.11 Further reading

- Hosmer DW, Lemeshow S. *Applied Logistic Regression*.
- Harrell FE. *Regression Modeling Strategies*, ch. 10.
- Gelman A, Hill J. *Data Analysis Using Regression and Multilevel…*.

7.61.12 Session info

```
sessionInfo()
```

7.61.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.62 Week 1, Session 3 — Multiple regression: confounding, interaction, centring

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.62.1 Learning objectives

- Distinguish confounding from mediation and from effect modification.
- Read a multiple regression coefficient as an *adjusted* effect.
- Centre predictors before fitting interactions and explain why.

7.62.2 Prerequisites

Sessions 1 and 2 of this week.

7.62.3 Background

Multiple regression is the standard way to estimate the effect of one predictor while holding others constant. The word *holding* is doing a lot of work: the adjusted coefficient is the effect conditional on the other variables in the model, not the effect one would observe in a new study that intervened on them. Confounding, mediation, and collider bias all pivot on which covariates are in the model. The statistics textbook cannot answer that question; the subject-matter science must.

Interactions complicate this picture. When two predictors interact, the effect of one depends on the level of the other, and the main-effect coefficient becomes the effect when the interacting variable is zero. Centring continuous predictors before fitting an interaction restores a readable interpretation: the main effect becomes the effect at the sample mean of the other variable, which is usually what the reader wants.

Collinearity and scale are often discussed together. Centring does not reduce *real* collinearity between X and Z, but it does reduce the algebraic collinearity between X and $X \cdot Z$ that otherwise makes the coefficients hard to interpret and inflates their standard errors.

7.62.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.62.5 1. Hypothesis

Simulate a scenario in which a confounder masks the effect of interest, then add an interaction to show effect modification.

7.62.6 2. Visualise

```
n <- 300
age <- rnorm(n, 55, 10)
# smoker more common among older people in this simulation (confounding)
smoker <- rbinom(n, 1, plogis(-3 + 0.05 * age))
# outcome depends on both; smoker effect is stronger at older ages
 ↪  (interaction)
```

```
y <- 120 + 0.6 * age + 5 * smoker + 0.2 * smoker * (age - mean(age)) +
  rnorm(n, 0, 8)
dat <- tibble(age, smoker = factor(smoker, labels = c("no", "yes")), y)

ggplot(dat, aes(age, y, colour = smoker)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "Age (years)", y = "Outcome", colour = "Smoker")
```

Both lines climb with age; the smoker line sits higher and may slope more steeply.

7.62.7 3. Assumptions

The usual linear-model assumptions, plus an implicit assumption that all relevant confounders are in the model.

```
fit_full <- lm(y ~ age * smoker, data = dat)
par(mfrow = c(2, 2))
plot(fit_full)
par(mfrow = c(1, 1))
```

7.62.8 4. Conduct

Crude smoker effect, then adjusted for age, then with interaction:

```
crude    <- lm(y ~ smoker, data = dat)
adjusted <- lm(y ~ smoker + age, data = dat)
inter    <- lm(y ~ smoker * age, data = dat)

bind_rows(
  tidy(crude, conf.int = TRUE) |> mutate(model = "crude"),
  tidy(adjusted, conf.int = TRUE) |> mutate(model = "adjusted"),
  tidy(inter, conf.int = TRUE) |> mutate(model = "interaction")
) |>
  filter(term != "(Intercept)") |>
  select(model, term, estimate, conf.low, conf.high, p.value)
```

Now with centred age, which makes the `smokeryes` coefficient the smoker effect *at mean age*:

```
dat_c <- dat |> mutate(age_c = age - mean(age))
inter_c <- lm(y ~ smoker * age_c, data = dat_c)
tidy(inter_c, conf.int = TRUE)
```

7.62.9 5. Concluding statement

> After adjusting for age, the estimated smoker-effect at the sample mean age was `round(coef(inter_c)["smokeryes"],1)` units (95% CI: `round(confint(inter_c)["smokeryes",1],1)` to `round(confint(inter_c)["smokeryes",2],1)`), with evidence of effect modification by age (interaction coefficient `round(coef(inter_c)["smokeryes:age_c"],2)`).

7.62.10 Common pitfalls

- Reading an adjusted coefficient as a population-level causal effect without thinking about what the adjustment set represents.
- Fitting an interaction without centring and then trying to interpret the main effects.
- Dropping covariates that look non-significant one at a time.

7.62.11 Further reading

- Greenland S (1998), *Confounding and collapsibility in causal inference*.
- VanderWeele TJ. *Explanation in Causal Inference*.
- Fox J. *Applied Regression Analysis and Generalized Linear Models*.

7.62.12 Session info

```
sessionInfo()
```

7.62.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.63 Week 2, Session 5 — Non-linear regression with nls

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.63.1 Learning objectives

- Fit a parametric non-linear model with `nls()` and report the three parameters.
- Choose starting values that let the optimiser converge.
- Distinguish a GAM's flexible curve from an `nls` model's mechanistic curve.

7.63.2 Prerequisites

Sessions 2 and 4 of this week.

7.63.3 Background

Where a GAM says *fit a smooth and see what shape it takes*, `nls` says *this curve has a known parametric form and I want the parameters.* Michaelis–Menten kinetics, three-parameter logistic dose-response, and exponential decay are all examples of mechanistic models where the parameters have physical meanings: asymptote, half-maximal response, rate, and so on.

Fitting non-linear models is slightly more fragile than fitting linear ones because the optimiser must start somewhere and can get lost. Sensible starting values come from the shape of the data: the asymptote from the highest fitted values, the inflection from where the curve flattens, and so on. `SSlogis` and related self-starting functions estimate these automatically for common parameterisations.

`nls` gives asymptotic standard errors by default. For small samples or poor parameter identifiability, bootstrap or profile-likelihood intervals (`confint(fit)` uses profiles) are more honest.

7.63.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.63.5 1. Hypothesis

Simulate dose-response data and fit a three-parameter logistic.

7.63.6 2. Visualise

```
n <- 60
dose <- rep(c(0.1, 0.3, 1, 3, 10, 30), each = 10)
# true params: asym = 100, xmid = log(3), scal = 0.7 (log-scale)
true_resp <- 100 / (1 + exp(-(log(dose) - log(3)) / 0.7))
resp <- true_resp + rnorm(n, 0, 5)
dat <- tibble(dose, resp)

ggplot(dat, aes(dose, resp)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  labs(x = "Dose (log scale)", y = "Response")
```

7.63.7 3. Assumptions

Correct model form, independent normal errors with constant variance on the response scale, and informative starting values.

```
fit <- nls(resp ~ SSlogis(log(dose), Asym, xmid, scal), data = dat)
resid_plot <- tibble(fitted = fitted(fit), resid = resid(fit))
ggplot(resid_plot, aes(fitted, resid)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = 2, colour = "grey50") +
  labs(x = "Fitted", y = "Residual")
```

7.63.8 4. Conduct

```
summary(fit)
confint(fit)

pred <- tibble(dose = 10 ^ seq(-1, 1.5, length.out = 100))
pred$resp <- predict(fit, newdata = pred)

ggplot(dat, aes(dose, resp)) +
  geom_point(alpha = 0.7) +
  geom_line(data = pred, colour = "steelblue", linewidth = 0.8) +
  scale_x_log10() +
  labs(x = "Dose (log scale)", y = "Response")
```

7.63.9 5. Concluding statement

> A three-parameter logistic fitted on the log-dose scale returned an asymptote of `round(coef(fit)["Asym"],1)` units and a mid-dose (EC50) at `round(exp(coef(fit)["xmid"]),2)`. Confidence intervals were obtained via the likelihood profile (see `confint`).

7.63.10 Common pitfalls

- Starting values that put the optimiser in a local minimum.
- Reporting symmetric SE-based intervals when the likelihood is asymmetric; prefer `confint`.
- Over-interpreting parameters when the data do not span the asymptote.

7.63.11 Further reading

- Bates DM, Watts DG. *Nonlinear Regression Analysis and Its Applications*.
- Ritz C, Streibig JC. *Nonlinear Regression with R*.
- Pinheiro JC, Bates DM. *Mixed-Effects Models in S and S-PLUS*, ch. 8.

7.63.12 Session info

```
sessionInfo()
```

7.63.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.64 Week 2, Session 1 — One-way ANOVA and contrasts (emmeans)

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.64.1 Learning objectives

- Fit a one-way ANOVA with `aov()` and interpret the omnibus F-test.
- Use `emmeans` to obtain estimated marginal means and pairwise contrasts with controlled family-wise error.
- Read a Tukey HSD output and translate it into a sentence.

7.64.2 Prerequisites

t-tests; linear regression basics.

7.64.3 Background

Analysis of variance is linear regression with a categorical predictor and a tradition. The omnibus F-test asks whether any group means differ; the follow-up contrasts ask which ones. The correct order is omnibus first, then targeted contrasts, preferably with a multiple testing correction that respects the family of tests you intended before seeing the data.

The `emmeans` package has made the post-hoc machinery substantially cleaner. `emmeans(fit,~group)` returns the estimated marginal means on the response scale; `pairs()` produces the pairwise differences with Tukey-adjusted p-values by default.

A common mistake is to run pairwise t-tests without correction and report the one with the smallest p-value. A less common but more insidious mistake is to run the omnibus F, see it is significant, and stop — reporting a single p-value without naming which groups differ and by how much.

7.64.4 Setup

```
library(tidyverse)
library(broom)
library(emmeans)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.64.5 1. Hypothesis

Using `ToothGrowth`, ask whether guinea-pig tooth length depends on dose of vitamin C.

Null: mean tooth length is equal across the three doses. Alternative: at least one dose differs.

7.64.6 2. Visualise

```
tg <- ToothGrowth |> mutate(dose = factor(dose))
ggplot(tg, aes(dose, len, fill = dose)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = "Dose (mg/day)", y = "Tooth length") +
  theme(legend.position = "none")
```

7.64.7 3. Assumptions

Approximate normality within groups and equal variances.

```
fit <- aov(len ~ dose, data = tg)
par(mfrow = c(2, 2))
plot(fit)
par(mfrow = c(1, 1))
bartlett.test(len ~ dose, data = tg)
```

7.64.8 4. Conduct

```
summary(fit)

emm <- emmeans(fit, ~ dose)
emm
pairs(emm, adjust = "tukey")
```

7.64.9 5. Concluding statement

> Tooth length increased with dose of vitamin C (one-way ANOVA $F =$ `round(summary(fit)[[1]]$"Fvalue"[1],2)`; $p =$ `signif(summary(fit)[[1]]$"Pr(>F)"[1],2)`). Tukey-adjusted contrasts indicated that the 2 mg/day dose gave longer teeth than both lower doses (all adjusted $p < 0.05$).

7.64.10 Common pitfalls

- Stopping at a significant F without reporting which contrasts drive it.
- Reporting unadjusted pairwise p-values after the omnibus test.
- Treating a non-significant Bartlett test as proof of equal variances.

7.64.11 Further reading

- Lenth RV. *emmeans* package vignette.
- Oehlert GW. *A First Course in Design and Analysis of Experiments*.
- Maxwell SE, Delaney HD. *Designing Experiments and Analyzing Data*.

7.64.12 Session info

```
sessionInfo()
```

7.64.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.65 Week 3, Session 3 — Ordinal and multinomial regression

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.65.1 Learning objectives

- Fit a proportional-odds model with `MASS::polr()` and report a cumulative odds ratio.
- Fit a multinomial logistic regression with `nnet::multinom()` for an unordered outcome.
- Check the proportional-odds assumption and decide what to do when it fails.

7.65.2 Prerequisites

Binary logistic regression.

7.65.3 Background

Ordinal outcomes — pain scales, tumour stages, Likert items — carry more information than a dichotomised version and usually less than a continuous version. The proportional-odds model (`polr`) assumes the effect of each predictor is the same across all cumulative splits of the response; the single coefficient is the log cumulative odds ratio. Multinomial logistic regression makes no such constraint and fits a separate logit for each non-reference category.

The proportional-odds assumption is strong. A Brant test (in the `brant` package) or simply comparing the `polr` fit to a multinomial fit on deviance can detect violations. When it fails, options include a partial proportional-odds model, the multinomial model, or a continuation-ratio model.

Multinomial models are notoriously hungry for data: each non-reference category requires a full set of coefficients. Reporting should list all coefficient tables and a global test (likelihood ratio) rather than a single p-value.

7.65.4 Setup

```
library(tidyverse)
library(broom)
library(MASS)
library(nnet)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.65.5 1. Hypothesis

Simulate an ordinal outcome (symptom severity in three levels) driven by a single continuous exposure, then contrast the ordinal and multinomial analyses.

7.65.6 2. Visualise

```
n <- 400
x <- rnorm(n)
eta <- 1.2 * x
# cumulative cutpoints
probs <- cbind(
  plogis(-0.5 - eta),
  plogis(1.0 - eta) - plogis(-0.5 - eta),
  1 - plogis(1.0 - eta)
```

```
)
y <- apply(probs, 1, function(p) sample(1:3, 1, prob = p))
dat <- tibble(x, y = factor(y, levels = 1:3,
                            labels = c("mild", "moderate", "severe"),
                            ordered = TRUE))

ggplot(dat, aes(y, x, fill = y)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = "Severity", y = "Exposure x") +
  theme(legend.position = "none")
```

7.65.7 3. Assumptions

Proportional-odds (same β across cumulative splits) for `polr`; independent observations for both models.

```
# quick graphical check: fit logistic at each cumulative split and
↪  compare slopes
splits <- tibble(cut = c("mild vs rest", "mild+mod vs severe")) |>
  mutate(coef = c(
    coef(glm(I(as.numeric(y) >= 2) ~ x, data = dat, family =
    ↪  binomial))[2],
    coef(glm(I(as.numeric(y) >= 3) ~ x, data = dat, family = binomial))[2]
  ))
splits
```

If the two slopes are close, proportional odds is plausible.

7.65.8 4. Conduct

```
fit_polr <- polr(y ~ x, data = dat, Hess = TRUE)
summary(fit_polr)
# approximate p-values
coef_summary <- coef(summary(fit_polr))
p <- pnorm(abs(coef_summary[, "t value"]), lower.tail = FALSE) * 2
cbind(coef_summary, "p value" = round(p, 3))
exp(coef(fit_polr))  # cumulative odds ratio
```

Multinomial:

```
fit_multi <- multinom(y ~ x, data = dat, trace = FALSE)
summary(fit_multi)
z <- summary(fit_multi)$coefficients / summary(fit_multi)$standard.errors
p_multi <- (1 - pnorm(abs(z), 0, 1)) * 2
round(p_multi, 3)
```

7.65.9 5. Concluding statement

The proportional-odds model returned a cumulative odds ratio per unit of x of `round(exp(coef(fit_polr))[1],2)`, meaning larger x shifted symptom severity upward. The multinomial model returned two logits (moderate vs mild and severe vs mild) with effects in the same direction; deviance-based comparison did not reject the proportional-odds simplification.

7.65.10 Common pitfalls

- Reporting only the odds ratio and ignoring whether proportional-odds holds.
- Using a multinomial model with a tiny reference category.
- Treating an ordered outcome as continuous and fitting a linear model.

7.65.11 Further reading

- Agresti A. *Analysis of Ordinal Categorical Data*.
- Harrell FE. *Regression Modeling Strategies*, ch. 13.
- Venables WN, Ripley BD. *Modern Applied Statistics with S*, ch. 7.

7.65.12 Session info

```
sessionInfo()
```

7.65.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.66 Week 3, Session 4 — Poisson and negative-binomial regression

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.66.1 Learning objectives

- Fit a Poisson GLM for counts with `glm(family=poisson)` and use an offset for exposure.
- Diagnose overdispersion and switch to negative-binomial regression with `MASS::glm.nb` when needed.
- Report incidence rate ratios with intervals.

7.66.2 Prerequisites

Logistic regression.

7.66.3 Background

Counts with a known exposure are the natural habitat of the Poisson GLM: number of events per person-year, per square metre, per trap night. The canonical link is the log, so the exponentiated coefficient is an incidence rate ratio. Exposure enters the model as an offset — a term with a fixed coefficient of 1 on the log scale.

The Poisson model assumes variance equals the mean. Most real count data violate this, with variance growing faster than the mean. Overdispersion inflates Type-I error when ignored. Checking the ratio of residual deviance to its degrees of freedom gives a quick read; when it is substantially above 1, the negative-binomial GLM (or a quasi-Poisson variance) is a better fit.

Poisson regression without an offset is a common trap. If the denominators vary, a coefficient on an exposure variable is confounded with exposure; the offset separates the two and restores the interpretation of the log-rate ratio.

7.66.4 Setup

```
library(tidyverse)
library(broom)
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.66.5 1. Hypothesis

Simulate counts of events with a dispersion parameter larger than 1 (so negative-binomial is the correct model), plus a known exposure.

7.66.6 2. Visualise

```
n <- 300
x <- rnorm(n)
exposure <- runif(n, 0.5, 2)
mu <- exposure * exp(0.5 + 0.4 * x)
# negative-binomial counts with theta = 2 (overdispersion)
y <- rnegbin(n, mu = mu, theta = 2)
dat <- tibble(x, exposure, y)

ggplot(dat, aes(x, y)) +
```

```
  geom_point(alpha = 0.6) +
  labs(x = "x", y = "Count")
```

7.66.7 3. Assumptions

Independent counts; correct mean structure; for Poisson, variance = mean.

```
fit_p <- glm(y ~ x + offset(log(exposure)), data = dat, family = poisson)
# dispersion
dev_over_df <- deviance(fit_p) / df.residual(fit_p)
dev_over_df
```

Dispersion » 1 signals negative-binomial.

7.66.8 4. Conduct

```
fit_nb <- glm.nb(y ~ x + offset(log(exposure)), data = dat)

bind_rows(
  tidy(fit_p, conf.int = TRUE, exponentiate = TRUE) |> mutate(model =
  ↪  "Poisson"),
  tidy(fit_nb, conf.int = TRUE, exponentiate = TRUE) |> mutate(model =
  ↪  "NB")
) |>
  filter(term == "x") |>
  dplyr::select(model, estimate, conf.low, conf.high, std.error)
```

The estimates are similar but the NB standard error is larger and honest.

7.66.9 5. Concluding statement

> Event counts increased with x (NB IRR per unit x = `round(exp(coef(fit_nb))["x"],2)`; 95% CI `round(exp(confint(fit_nb))["x",1],2)` to `round(exp(confint(fit_nb))["x",2],2)`). Poisson residual deviance indicated overdispersion; we therefore report the negative-binomial fit as primary.

7.66.10 Common pitfalls

- Forgetting the offset when denominators vary.
- Using Poisson when overdispersion is present.
- Reporting the rate ratio without an interval.

7.66.11 Further reading

- McCullagh P, Nelder JA. *Generalized Linear Models*, ch. 6.

- Hilbe JM. *Negative Binomial Regression*.
- Zeileis A, Kleiber C, Jackman S (2008), *Regression models for count data*.

7.66.12 Session info

```
sessionInfo()
```

7.66.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.67 Week 2, Session 3 — RCBD and repeated measures → mixed models

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.67.1 Learning objectives

- Fit a randomised complete block design (RCBD) with `aov()` and read the block term.
- Move from repeated-measures ANOVA to a mixed model with `lme4::lmer()`.
- Interpret random intercepts as between-subject variance and residuals as within-subject variance.

7.67.2 Prerequisites

Sessions 1–2 of this week.

7.67.3 Background

Blocks and repeated measures share a single idea: some units are more similar to themselves than to the rest of the data. In an RCBD, the block absorbs nuisance variation (plots in a field, litter in an experiment) so the treatment effect can be estimated against a smaller error. In repeated-measures designs, subjects serve as their own blocks because each is measured more than once. In both cases, ignoring the grouping inflates the standard error of the treatment effect and sometimes the Type I error of the test.

Classical RCBD ANOVA treats the block as a fixed factor with a line in the ANOVA table. Modern practice is usually to treat the block (or subject) as

a random effect in a mixed model, which separates between-block and within-block variance explicitly and produces standard errors that account for the correlation.

The `sleep` dataset from base R is a ready-made within-subject design: each of ten subjects received two drugs and reported extra hours of sleep. A paired t-test answers the main question; a mixed model with subject as a random intercept is the same thing with more machinery and more honest diagnostics.

7.67.4 Setup

```
library(tidyverse)
library(broom)
library(broom.mixed)
library(lme4)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.67.5 1. Hypothesis

Using the `sleep` dataset, does drug group affect extra hours of sleep after accounting for the within-subject pairing?

7.67.6 2. Visualise

```
ggplot(sleep, aes(group, extra, group = ID)) +
  geom_line(colour = "grey70") +
  geom_point(aes(colour = group), size = 3) +
  labs(x = "Drug", y = "Extra sleep (hours)", colour = "Drug")
```

Lines track each subject across the two drugs. Most rise from left to right, suggesting drug 2 adds more sleep than drug 1.

7.67.7 3. Assumptions

```
fit_rcbd <- aov(extra ~ group + Error(ID), data = sleep)
summary(fit_rcbd)
```

Residual diagnostics for the mixed version:

```
fit_lmm <- lmer(extra ~ group + (1 | ID), data = sleep)
qqnorm(resid(fit_lmm)); qqline(resid(fit_lmm))
plot(fitted(fit_lmm), resid(fit_lmm),
     xlab = "Fitted", ylab = "Residuals"); abline(h = 0, lty = 2)
```

7.67.8 4. Conduct

```
summary(fit_lmm)
tidy(fit_lmm, conf.int = TRUE, effects = "fixed")
```

The random intercept variance captures between-subject differences in baseline sleep; the residual variance captures within-subject noise.

7.67.9 5. Concluding statement

> In ten subjects each given both drugs, drug 2 was associated with `round(fixef(fit_lmm)[2],2)` additional hours of extra sleep relative to drug 1 (95% CI from the mixed model: `round(confint(fit_lmm,method="Wald")[4,1],2)` to `round(confint(fit_lmm,method="Wald")[4,2],2)`). The model included a random intercept for subject to account for the pairing.

7.67.10 Common pitfalls

- Ignoring the pairing and fitting a two-sample t-test.
- Fitting a random slope that the data cannot identify (tiny cluster sizes).
- Reporting only fixed-effect estimates and omitting the variance components.

7.67.11 Further reading

- Pinheiro JC, Bates DM. *Mixed-Effects Models in S and S-PLUS*.
- Bates D et al. (2015), *Fitting linear mixed-effects models using lme4*.
- Zuur AF et al. *Mixed Effects Models and Extensions in Ecology*.

7.67.12 Session info

```
sessionInfo()
```

7.67.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.68 Week 1, Session 5 — Robust regression, weighted LS, HC (sandwich) standard errors

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.68.1 Learning objectives

- Fit a robust regression with `MASS::rlm()` and compare coefficients with OLS.
- Use weighted least squares when variance is known to vary with a predictor.
- Replace OLS standard errors with heteroscedasticity-consistent (sandwich) ones using `sandwich::vcovHC` and `lmtest::coeftest`.

7.68.2 Prerequisites

Sessions 2–4 of this week.

7.68.3 Background

Classical OLS is optimal under a narrow set of assumptions. When residuals have heavy tails, a few observations can dominate the fit; M-estimators such as those in `rlm` downweight extreme residuals and return coefficients that reflect the bulk of the data. When variance is known to depend on a predictor, weighted least squares restores efficiency by weighting each observation by the inverse of its variance. When the variance pattern is unknown but you want valid *inference*, sandwich (HC) standard errors correct the standard errors without changing the point estimates.

The three tools solve different problems. Robust regression changes the coefficient. Weighted LS changes both the coefficient and the standard error if you have the correct weights. HC standard errors keep the OLS coefficient and replace only the standard errors. Choose the one that matches your threat.

The most common use of HC standard errors in practice is in cross-sectional regression where you suspect heteroscedasticity but have no good model for it. HC3 is a sensible default in small samples; HC0 is classical but liberal.

7.68.4 Setup

```
library(tidyverse)
library(broom)
library(MASS)        # rlm; load before tidyverse dplyr::select is not
 ↪  needed here
```

```
library(sandwich)
library(lmtest)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.68.5 1. Hypothesis

Fit an OLS, a robust, and a weighted regression to the same data and compare. Then repeat the OLS with sandwich standard errors.

7.68.6 2. Visualise

```
n <- 200
x <- runif(n, 0, 10)
eps <- rnorm(n, 0, 0.3 + 0.2 * x)     # heteroscedastic
# a few extreme outliers
outl <- sample(seq_len(n), 8)
eps[outl] <- eps[outl] + rnorm(8, 0, 6)
y <- 2 + 0.7 * x + eps
dat <- tibble(x, y)

ggplot(dat, aes(x, y)) +
  geom_point(alpha = 0.6) +
  labs(x = "x", y = "y")
```

7.68.7 3. Assumptions

```
fit_ols <- lm(y ~ x, data = dat)
par(mfrow = c(2, 2))
plot(fit_ols)
par(mfrow = c(1, 1))
```

Fan-shaped residuals and heavy QQ tails — textbook cue for robust or sandwich treatment.

7.68.8 4. Conduct

OLS, robust, and weighted fits:

```
fit_rlm <- MASS::rlm(y ~ x, data = dat)
# variance grows roughly with x; weight by 1 / (fitted variance proxy)
w <- 1 / (0.3 + 0.2 * dat$x)^2
fit_wls <- lm(y ~ x, data = dat, weights = w)
```

```
bind_rows(
  tidy(fit_ols) |> mutate(model = "OLS"),
  tibble(term = c("(Intercept)", "x"),
         estimate = coef(fit_rlm),
         std.error = sqrt(diag(vcov(fit_rlm))),
         model = "robust"),
  tidy(fit_wls) |> mutate(model = "WLS")
) |>
  filter(term == "x") |>
  dplyr::select(model, estimate, std.error)
```

Sandwich SEs on the OLS fit:

```
coeftest(fit_ols, vcov. = vcovHC(fit_ols, type = "HC3"))
```

7.68.9 5. Concluding statement

OLS, robust, and weighted fits gave similar slope estimates (approximately `round(coef(fit_ols)[2],2)`, `round(coef(fit_rlm)[2], 2)`, and `round(coef(fit_wls)[2],2)` respectively). Classical OLS standard errors were biased by heteroscedasticity; HC3 sandwich standard errors were preferred for inference.

7.68.10 Common pitfalls

- Using `rlm` and then testing coefficients with classical OLS code that does not know the residuals were reweighted.
- Choosing weights from the residuals of an earlier fit without acknowledging the circularity.
- Quoting HC standard errors without saying which HC flavour was used.

7.68.11 Further reading

- Huber PJ, Ronchetti EM. *Robust Statistics*, 2nd ed.
- MacKinnon JG, White H (1985), *Some heteroscedasticity-consistent…*
- Zeileis A (2004), *Econometric computing with HC and HAC errors.*

7.68.12 Session info

```
sessionInfo()
```

7.68.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.69 Week 1, Session 2 — Simple linear regression

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.69.1 Learning objectives

- Fit a simple linear regression with `lm()` and read the output.
- Produce a tidy coefficient table with intervals using `broom`.
- Draw the fitted line on the scatterplot and express the slope in the units of the variables.

7.69.2 Prerequisites

Correlation vs regression; basic comfort with ggplot2.

7.69.3 Background

Simple linear regression models the mean of a response Y as a linear function of a single predictor X. The model is $Y = \beta_0 + \beta_1 X + \varepsilon$ with the error term ε assumed independent, zero-mean, and of constant variance. The slope β_1 is the expected change in Y for a one-unit change in X; the intercept β_0 is the expected Y at $X = 0$, which is sometimes meaningful and sometimes only a device for anchoring the line.

Although the formulas are old, the habits they require are modern: always plot first, always report an interval, and always read the slope back in the units of the variables. A regression coefficient is only useful if the reader can imagine the units on the axis.

The default `summary()` printout from `lm()` is dense. A clean way to read a fit is to use `broom::tidy()` for coefficients and `broom::glance()` for global quantities such as R^2 and residual standard error, and then plot the line on the data to sanity-check.

7.69.4 Setup

```
library(tidyverse)
library(broom)
library(palmerpenguins)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.69.5 1. Hypothesis

Among Adelie penguins, does bill length predict body mass?

Null: slope of body mass on bill length is zero. Alternative: slope is non-zero.

7.69.6 2. Visualise

```
ad <- penguins |>
  filter(species == "Adelie") |>
  drop_na(bill_length_mm, body_mass_g)

ggplot(ad, aes(bill_length_mm, body_mass_g)) +
  geom_point(alpha = 0.6, colour = "grey30") +
  geom_smooth(method = "lm", se = TRUE, colour = "steelblue") +
  labs(x = "Bill length (mm)", y = "Body mass (g)")
```

The cloud climbs gently from left to right. The smoothed line is an honest guess at the conditional mean.

7.69.7 3. Assumptions

Linearity, independence, homoscedasticity, and approximate normality of residuals.

```
fit <- lm(body_mass_g ~ bill_length_mm, data = ad)
par(mfrow = c(2, 2))
plot(fit)
par(mfrow = c(1, 1))
```

Residuals vs fitted is patternless; QQ is close to straight. No single point dominates.

7.69.8 4. Conduct

```
tidy(fit, conf.int = TRUE)
glance(fit) |> select(r.squared, adj.r.squared, sigma, p.value)
```

```
slope <- coef(fit)[2]
ci <- confint(fit)[2, ]
```

7.69.9 5. Concluding statement

> Among Adelie penguins ($n =$ `nrow(ad)`), each additional mm of bill length was associated with an increase of `round(slope,0)` g in body mass (95% CI: `round(ci[1],0)` to `round(ci[2],0)` g; $p =$ `signif(gl`

ance(fit)$p.value,2)). Bill length explained round(100*glance(fit)$r.squared,1)% of the variance in body mass.

7.69.10 Common pitfalls

- Extrapolating beyond the range of X.
- Interpreting the intercept when X = 0 is nonsensical.
- Quoting R^2 without reporting the slope and its interval.

7.69.11 Further reading

- Faraway JJ. *Linear Models with R*, ch. 2.
- Kutner MH et al. *Applied Linear Statistical Models*, ch. 1–3.
- Weisberg S. *Applied Linear Regression*.

7.69.12 Session info

```
sessionInfo()
```

7.69.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

7.70 Week 2, Session 2 — Two-way / factorial ANOVA with interaction

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

7.70.1 Learning objectives

- Fit a two-factor ANOVA and distinguish main effects from interactions.
- Visualise an interaction with `emmip()` and read the picture before the table.
- Report a factorial analysis in the correct order: interaction first, then conditional main effects.

7.70.2 Prerequisites

One-way ANOVA.

7.70.3 Background

A factorial design crosses two (or more) categorical factors. The two-way ANOVA decomposes variability into main effects and the interaction between

factors. If the interaction is negligible, main effects can be reported as averages across levels of the other factor; if not, the main-effect table is misleading and conditional (simple) effects must be reported instead.

The canonical small dataset is `ToothGrowth`, which crosses dose of vitamin C with delivery method (orange juice vs ascorbic acid). It is a complete factorial and nicely balanced, which means the classification of variance into Type-I, Type-II, and Type-III sums of squares does not matter. With unbalanced designs it does, and Type-III via `car::Anova` with contrasts set to `contr.sum` is the usual safe choice.

Plotting the interaction is almost always the right first step. An interaction plot that shows parallel lines agrees with an insignificant interaction term; crossed lines signal an interaction; lines with different slopes signal a quantitative interaction.

7.70.4 Setup

```
library(tidyverse)
library(broom)
library(car)
library(emmeans)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

7.70.5 1. Hypothesis

Does delivery method alter the effect of dose on tooth length?

Null: no interaction between supp and dose. Alternative: the dose effect differs by supp.

7.70.6 2. Visualise

```
tg <- ToothGrowth |> mutate(dose = factor(dose))

ggplot(tg, aes(dose, len, colour = supp, group = supp)) +
  stat_summary(fun = mean, geom = "point", size = 3) +
  stat_summary(fun = mean, geom = "line") +
  stat_summary(fun.data = mean_cl_normal, geom = "errorbar", width = 0.1)
  ↪  +
  labs(x = "Dose (mg/day)", y = "Tooth length", colour = "Supp")
```

7.70.7 3. Assumptions

```
fit <- aov(len ~ supp * dose, data = tg)
par(mfrow = c(2, 2))
plot(fit)
par(mfrow = c(1, 1))
```

7.70.8 4. Conduct

```
# Type-II sums of squares for a balanced design
Anova(lm(len ~ supp * dose, data = tg), type = 2)

emm <- emmeans(fit, ~ supp | dose)
pairs(emm)

emmip(fit, supp ~ dose, CIs = TRUE)
```

7.70.9 5. Concluding statement

> Tooth length responded to dose (large main effect) and to supplement (smaller main effect), with evidence of an interaction (see ANOVA table). The difference between orange juice and ascorbic acid was largest at the lowest dose and negligible at the highest.

7.70.10 Common pitfalls

- Reporting main effects without first checking the interaction.
- Using Type-III sums of squares with default treatment contrasts in unbalanced designs; the output depends on the contrast coding.
- Ignoring unequal cell sizes when reading an ANOVA table.

7.70.11 Further reading

- Fox J, Weisberg S. *An R Companion to Applied Regression*, ch. 5.
- Cochran WG, Cox GM. *Experimental Designs*.
- Wilkinson L, Rogers CE (1973), *Symbolic description of factorial…*

7.70.12 Session info

```
sessionInfo()
```

7.70.13 See also — chapter index

- Methods in this chapter

- Find your method (Chapter 0)

Chapter 8

Multivariate Methods

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-10

Methods for high-dimensional data without an outcome variable: principal components, factor analysis, canonical correlation, linear discriminant analysis, hierarchical and partition clustering, UMAP, and t-SNE. Both the variance-explained and the visual-projection use cases are covered.

This chapter contains **30 method pages** and **3 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

8.1 Method pages

Method	Source slug
Bray-Curtis Dissimilarity	`bray-curtis`
Canonical Correlation Analysis	`canonical-correlation`
Cluster Validation: Silhouette	`cluster-validation-silhouette`
Confirmatory Factor Analysis	`factor-analysis-confirmatory`
Correspondence Analysis	`correspondence-analysis`
DBSCAN Clustering	`dbscan-clustering`
Distance and Dissimilarity Measures	`distance-measures`
Exploratory Factor Analysis	`factor-analysis-exploratory`
Factor Rotation	`factor-rotation`
Gaussian Mixture Models	`gaussian-mixture-models`
Hierarchical Clustering	`hierarchical-clustering`
Independent Component Analysis	`ica-independent-component-analysis`
Jaccard and Dice Dissimilarity	`jaccard-dissimilarity`
k-Means Clustering	`kmeans-clustering`
Kernel PCA	`kernel-pca`

Method	Source slug
Linear Discriminant Analysis	`linear-discriminant-analysis`
Mahalanobis Distance	`mahalanobis-distance`
MANOVA	`manova`
Metric Multidimensional Scaling	`mds-metric`
Multiple Correspondence Analysis	`multiple-correspondence-analysis`
Non-Metric Multidimensional Scaling	`mds-nonmetric`
Partial Least Squares Regression	`pls-regression`
Partitioning Around Medoids	`pam-kmedoids`
PLS Discriminant Analysis	`pls-da`
Principal Component Analysis	`principal-component-analysis`
Quadratic Discriminant Analysis	`quadratic-discriminant-analysis`
Structural Equation Models	`structural-equation-models`
The Elbow Method	`elbow-method`
The Gap Statistic	`gap-statistic`
UMAP and t-SNE: Overview	`umap-tsne-overview`

8.2 Labs

Lab
PCA, factor analysis, CCA, LDA
Clustering: k-means, hierarchical, model-based
UMAP and t-SNE

8.3 Bray-Curtis Dissimilarity

8.3.1 Introduction

Bray-Curtis dissimilarity is the default index for comparing ecological communities described by species-abundance data. It ranges from 0 when two samples share the same composition and abundances to 1 when they share no species at all, and unlike binary indices it makes use of the abundance information rather than collapsing it to presence/absence. Microbial ecology, plant community work, and any compositional dataset where rare and abundant species both contribute meaningful information typically reach for Bray-Curtis as the first dissimilarity to compute.

8.3.2 Prerequisites

A working knowledge of distance and dissimilarity matrices, abundance data structures, and ordination methods (NMDS, PCoA) that consume dissimilarity matrices.

8.3.3 Theory

For two samples i and j with abundances x_{ik} across species k:

$$\mathrm{BC}_{ij} = \frac{\sum_k |x_{ik} - x_{jk}|}{\sum_k (x_{ik} + x_{jk})}.$$

It is a dissimilarity, not a strict metric — the triangle inequality can fail, so Bray-Curtis matrices are best visualised with non-metric MDS (NMDS) rather than principal coordinates. The binary form (presence-absence input) reduces to the Sørensen dissimilarity. Bray-Curtis is sensitive to dominant species; a log or Hellinger transformation of the abundance matrix before computing the index down-weights dominance and lets rare species contribute more.

8.3.4 Assumptions

Count or abundance data, all entries non-negative, comparable sampling effort across samples (or normalisation to relative abundance to remove the effort difference).

8.3.5 R Implementation

```r
library(vegan)

data(dune)
d_bc <- vegdist(dune, method = "bray")
as.matrix(d_bc)[1:3, 1:3]

# Log-transform to down-weight dominant species
d_bc_log <- vegdist(log1p(dune), method = "bray")

# NMDS visualisation
nmds <- metaMDS(dune, distance = "bray", k = 2, trymax = 50)
plot(nmds, type = "t")
```

8.3.6 Output & Results

`vegdist()` returns a `dist` object containing the lower-triangular Bray-Curtis matrix. `metaMDS()` runs NMDS on the resulting matrix and returns coordinates, stress, and convergence diagnostics. Stress values below 0.1 indicate excellent representation; 0.1–0.2 acceptable; above 0.2 poor.

8.3.7 Interpretation

A reporting sentence: “Bray-Curtis dissimilarity averaged 0.55 across all pairs (range 0.32 to 0.84), indicating substantial community turnover; NMDS in two

dimensions converged at stress 0.07, and PERMANOVA (`vegan::adonis2`) detected a significant management-type effect ($F = 3.6$, $p = 0.001$, $R^2 = 0.21$)." Always pair Bray-Curtis ordination with a formal group test such as PERMANOVA or ANOSIM.

8.3.8 Practical Tips

- Log-transform abundance (`log1p()`) or use Hellinger distance (`vegdist(...,method="hellinger")`) to reduce dominance by abundant species.
- Bray-Curtis is the default distance in `vegan::metaMDS()`; passing the data matrix directly is equivalent to `vegdist(data,"bray")` followed by `metaMDS()` on the dist.
- For phylogenetic-aware distances on microbial communities, use UniFrac (`GUniFrac::GUniFrac`) instead; it accounts for evolutionary relationships among taxa.
- Pair NMDS with PERMANOVA (`adonis2`) for group comparison and PERMDISP (`betadisper`) to check homogeneity of multivariate dispersion; significant PERMDISP can confound PERMANOVA conclusions.
- `vegdist()` supports many alternatives — Jaccard, Manhattan, Euclidean, Canberra, Gower, Hellinger — change `method` to switch.
- For very large community matrices, sparse implementations in `parallelDist` scale Bray-Curtis computation across cores without code changes.

8.3.9 R Packages Used

`vegan` for `vegdist()`, `metaMDS()`, `adonis2()`, and `betadisper()`; `GUniFrac` for phylogenetic UniFrac variants on microbiome data; `parallelDist` for fast multicore distance computation; `phyloseq` for microbiome workflows that integrate Bray-Curtis with sequence and tree data.

8.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.3.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.4 Canonical Correlation Analysis

8.4.1 Introduction

Canonical correlation analysis (CCA) finds pairs of linear combinations — one from each of two variable sets — that are maximally correlated. Where multiple regression has one outcome and many predictors, CCA generalises to many outcomes and many predictors simultaneously: the first canonical pair (U_1, V_1) is the linear combination of the predictor side and the linear combination of the outcome side whose correlation is highest. Successive pairs are constrained to be uncorrelated with previous ones, mirroring the orthogonality of principal components.

8.4.2 Prerequisites

A working understanding of multiple regression, principal components analysis, and the geometry of correlation in multivariate spaces.

8.4.3 Theory

Given two sets of variables X (p columns) and Y (q columns), CCA finds pairs of canonical variates $U_k = Xa_k$ and $V_k = Yb_k$ maximising $\text{cor}(U_k, V_k)$ subject to $\text{Var}(U_k) = \text{Var}(V_k) = 1$ and orthogonality with previous pairs. There are $\min(p, q)$ canonical pairs, indexed by canonical correlations $\rho_1 \geq \rho_2 \geq \ldots$ that are non-increasing.

Inference under multivariate Normality uses Wilks' lambda or Pillai's trace to test whether canonical correlations beyond a given index are zero. Canonical loadings (correlations between original variables and canonical variates) help interpret the structure of each canonical pair.

8.4.4 Assumptions

Multivariate Normality (for the asymptotic significance tests), linear associations between variable sets, and adequate sample size relative to $p + q$. CCA is sensitive to outliers; one or two extreme observations can dominate the canonical structure.

8.4.5 R Implementation

```
library(CCA)

# Two sets: first two and last two iris variables (artificial split)
X <- scale(iris[, 1:2])
Y <- scale(iris[, 3:4])
```

```
cc_res <- cc(X, Y)
cc_res$cor      # canonical correlations
cc_res$xcoef    # canonical coefficients for X
cc_res$ycoef    # canonical coefficients for Y

# Significance via Wilks' lambda
wilks <- (1 - cc_res$cor^2)
wilks
```

8.4.6 Output & Results

`cc()` returns canonical correlations (sorted in decreasing order), canonical coefficients (weights for each variable set), and the canonical variate scores. Wilks' lambda statistic and its associated F-approximation test the joint significance of canonical correlations from a given index onwards.

8.4.7 Interpretation

A reporting sentence: "Canonical correlation analysis on the iris measurements found a first canonical correlation of $\rho_1 = 0.94$ ($p < 0.001$, Wilks' $\lambda = 0.077$); the canonical loadings showed petal dimensions dominating the Y-side combination, indicating that overall flower size is the primary axis of association between the sepal and petal measurements." Report canonical loadings (correlations) rather than raw weights for interpretation; the latter are scale-dependent.

8.4.8 Practical Tips

- Interpret only the first few canonical correlations; later pairs typically capture little signal.
- For high-dimensional settings ($p + q$ approaching or exceeding n), use regularised CCA (`CCA::rcc`); standard CCA breaks down with rank-deficient covariance matrices.
- Partial CCA (`CCA::pcc` or `CCP::p.asym`) controls for covariates that should be removed before testing the canonical structure.
- For predictive use, partial least squares (`pls::plsr` or `mixOmics::spls`) is often preferable; PLS optimises predictive covariance rather than mutual correlation.
- Canonical loadings (correlations between variables and canonical variates) interpret cleaner than raw weights because they are scale-independent.
- Always pre-screen outliers; CCA is highly sensitive to extreme observations, and a robust CCA variant (`rrcov::CcaClassic` vs `CcaRobust`) is worth comparing.

8.4.9 R Packages Used

`CCA` for `cc()`, `rcc()`, and visualisation helpers; `CCP` for asymptotic significance tests (Wilks' lambda, Pillai, Hotelling-Lawley); `vegan::CCorA()` for ecology-flavoured CCA; `mixOmics::cca()` for sparse and regularised variants integrated with omics workflows.

8.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.4.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.5 Cluster Validation: Silhouette

8.5.1 Introduction

The silhouette is the most widely used internal validation metric for clustering. For each observation, it compares the mean distance to others in its own cluster against the mean distance to the nearest competing cluster, returning a value between -1 and $+1$. Values near 1 indicate observations comfortably inside their assigned cluster; values near 0 indicate ambiguous assignments on a cluster boundary; negative values indicate observations that would fit better in a different cluster. Averaging silhouettes across observations gives a single number summarising cluster quality without needing labels or external references.

8.5.2 Prerequisites

A working understanding of distance metrics, clustering algorithms (k-means, hierarchical, PAM), and the distinction between internal and external validation criteria.

8.5.3 Theory

For observation i in cluster C_k, define:

- a_i = mean distance from i to all other observations in C_k.

- b_i = minimum, over clusters $C_l \neq C_k$, of the mean distance from i to observations in C_l.

The silhouette width is

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}, \qquad s_i \in [-1, 1].$$

The cluster silhouette is the mean s_i across observations in that cluster; the overall silhouette is the mean across all observations. Conventional thresholds: > 0.7 indicates a strong cluster structure, 0.5–0.7 reasonable, 0.25–0.5 weak, < 0.25 no substantial structure.

8.5.4 Assumptions

A meaningful distance metric; unambiguous cluster assignments. Silhouette assumes convex clusters and can be misleading for density-defined or non-convex shapes (rings, half-moons), where DBSCAN-style validation is more appropriate.

8.5.5 R Implementation

```r
library(cluster); library(factoextra)

X <- scale(iris[, 1:4])

# k-means with silhouette
km <- kmeans(X, centers = 3, nstart = 25)
sil <- silhouette(km$cluster, dist(X))
summary(sil)

fviz_silhouette(sil)

# Choose k by silhouette
fviz_nbclust(X, kmeans, method = "silhouette")
```

8.5.6 Output & Results

`silhouette()` returns per-observation s_i, the assigned cluster, and the nearest-competing cluster. `summary()` aggregates by cluster and overall. `fviz_silhouette()` produces the canonical horizontal-bar silhouette plot. `fviz_nbclust()` automates k selection by maximising overall silhouette.

8.5.7 Interpretation

A reporting sentence: "k-means with $k = 3$ on the standardised iris data produced an overall silhouette width of 0.51 (cluster widths 0.79, 0.39, 0.45), no observations had negative silhouette, and 12 observations near the versicolor / virginica boundary had values below 0.20." Always report cluster-specific silhouettes, not just the overall mean — a poorly-fitting cluster can be hidden inside a high overall average.

8.5.8 Practical Tips

- Use silhouette to choose k alongside the elbow method and gap statistic; concordance across all three is the strongest argument for a chosen partition.
- Negative silhouettes flag misassigned observations; investigate whether they represent genuine intermediate cases, data errors, or an inadequate k.
- Silhouette is misleading for non-convex clusters (rings, manifolds); for density-based methods (DBSCAN, OPTICS), prefer density-aware validation indices.
- Plot silhouette per cluster and compare cluster-level means; a single low-silhouette cluster pulls the overall mean down without a per-cluster breakdown making it visible.
- Alternative internal indices (Davies-Bouldin, Calinski-Harabasz, Dunn) capture different aspects of cluster quality; `NbClust` computes 30+ and reports the consensus optimum.
- Silhouette computation is $O(n^2)$ in distance evaluations; for very large n, use sub-sample-based silhouette via `cluster::silhouette()` on a random subset.

8.5.9 R Packages Used

`cluster` for `silhouette()` and the underlying clustering algorithms; `factoextra` for `fviz_silhouette()` and `fviz_nbclust()` ggplot-style visualisation; `NbClust` for 30+ alternative internal indices in one call; `clusterCrit` for additional intrinsic and extrinsic validation indices.

8.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.5.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.6 Correspondence Analysis

8.6.1 Introduction

Correspondence analysis (CA) is the categorical analogue of principal components analysis. It takes a two-way contingency table and produces a low-dimensional map in which rows and columns become points whose proximity reflects association beyond what would be expected under independence. CA decomposes the chi-squared statistic into row and column contributions across a small number of dimensions, much as PCA decomposes covariance variance, and the resulting biplot is one of the most intuitive ways to visualise the association structure of a contingency table.

8.6.2 Prerequisites

A working understanding of contingency tables, the chi-squared statistic for independence, principal components analysis, and singular value decomposition.

8.6.3 Theory

For an $r \times c$ contingency table with cell counts n_{ij} and margins $n_{i\cdot}$, $n_{\cdot j}$, total n:

1. Compute expected counts under independence $e_{ij} = n_{i\cdot} n_{\cdot j} / n$.
2. Form the matrix of standardised residuals $z_{ij} = (n_{ij} - e_{ij}) / \sqrt{e_{ij}}$.
3. Apply singular value decomposition $Z = U D V^{\top}$.
4. Row coordinates and column coordinates come from U and V scaled by D and the row/column masses; the squared singular values sum to the total inertia χ^2 / n.

The biplot places row points close to column points with positive association; points close to the origin are near the average profile and contribute little to the structure. Multiple correspondence analysis (MCA) generalises CA to more than two categorical variables.

8.6.4 Assumptions

Count data in a rectangular contingency table with no zero marginal sums (all rows and columns have at least one observation), and a meaningful chi-squared interpretation of association.

8.6.5 R Implementation

```
library(FactoMineR); library(factoextra)

data(smoke)
ca_res <- CA(smoke, graph = FALSE)
fviz_ca_biplot(ca_res)
summary(ca_res)
```

8.6.6 Output & Results

`CA()` returns row and column coordinates, contributions to each axis, total inertia, and inertia per axis (the categorical analogue of variance explained). `fviz_ca_biplot()` displays both row and column points on the first two principal axes, with axis labels showing the proportion of inertia captured.

8.6.7 Interpretation

A reporting sentence: "Correspondence analysis of the smoke dataset (5 rank levels × 4 smoking categories) decomposed total inertia of 0.085 into three principal axes; the first axis (88 % of inertia) placed senior employees close to the 'heavy smoker' category and junior managers near 'none', suggesting an age-rank-smoking association beyond chance ($\chi^2 = 16.4$, $p = 0.17$ — the visual association is strong but the statistical test is underpowered for this small table)." Always quote the proportion of inertia displayed and complement with the chi-squared test.

8.6.8 Practical Tips

- Report the proportion of inertia retained in the displayed dimensions; if the first two axes capture less than 70 %, much of the structure is not visible in the biplot.
- For more than two categorical variables, switch to multiple correspondence analysis (MCA, `FactoMineR::MCA`); supplementary continuous covariates can be projected on top.
- Remove or merge zero-count rows or columns before fitting; they break the chi-squared transformation and produce undefined coordinates.
- The symmetric biplot scales rows and columns comparably; the row-principal or column-principal alternatives are useful when one dimension is the focus but distort the joint geometry.
- Pair CA with a chi-squared test on the original table; the inertia decomposition shows where the association concentrates, while the test quantifies whether it exceeds chance.
- For supplementary categories that should not influence the axes (e.g. demographic markers on the original survey items), pass them through the

`quali.sup` argument; they are projected after fitting but do not affect axis estimation.

8.6.9 R Packages Used

`FactoMineR` for `CA()`, `MCA()`, and the broader factor-analysis family; `factoextra` for `fviz_ca_biplot()`, `fviz_ca_row()`, `fviz_ca_col()` ggplot-style visualisation; `ca` for an alternative implementation with classic biplot styling; `ade4::dudi.coa()` for ecology-flavoured CA.

8.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.6.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.7 DBSCAN Clustering

8.7.1 Introduction

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is the most influential density-based clustering algorithm. Unlike k-means, it does not require specifying the number of clusters in advance, finds clusters of arbitrary shape (rings, half-moons, elongated bands), and explicitly flags low-density points as noise rather than forcing them into a cluster. These properties make it the right choice for spatial data, anomaly detection alongside clustering, and any context where the intuitive notion of "cluster" is "a dense region surrounded by low-density gaps".

8.7.2 Prerequisites

A working understanding of distance metrics, density-based reasoning, and the limitations of partitioning algorithms (k-means, PAM) for non-convex cluster shapes.

8.7.3 Theory

DBSCAN has two parameters: ϵ (eps), the neighbourhood radius, and minPts, the minimum number of points (including the centre) required for a point to be a *core* point.

- A **core point** has at least minPts points within distance ϵ.
- A **border point** is within ϵ of a core point but is not itself a core point.
- A **noise point** is neither.

Clusters are formed as maximal connected sets of core points (with their associated border points), with noise points left unassigned. The algorithm is order-invariant for core-point structure but border-point assignments can depend on processing order. Choosing ϵ is the practical challenge; the standard heuristic is the k-distance plot, which shows the distance to the k-th nearest neighbour for each point sorted in ascending order — a knee in the curve marks a sensible ϵ.

8.7.4 Assumptions

A meaningful distance metric; clusters have sufficient density separation to be distinguished from noise; standardised input so distances are comparable across variables.

8.7.5 R Implementation

```r
library(dbscan)

X <- scale(iris[, 1:4])

# Choose eps from k-distance plot
kNNdistplot(X, k = 5)
abline(h = 0.7, col = "red", lty = 2)

# DBSCAN
db <- dbscan(X, eps = 0.7, minPts = 5)
db$cluster

table(db$cluster, iris$Species)
```

8.7.6 Output & Results

`dbscan()` returns a vector of cluster labels with `0` indicating noise. Core points, border points, and noise are tagged separately in the result object. The k-distance plot guides ϵ selection; the table comparison against species labels shows how DBSCAN's discovered structure aligns with known classes.

8.7.7 Interpretation

A reporting sentence: "DBSCAN with $\epsilon = 0.7$ and minPts $= 5$ identified three clusters and flagged 10 observations as noise; the clusters reproduced the iris species structure with 88 % agreement, with the noise points concentrated at the versicolor / virginica boundary." Always state ϵ, minPts, and the number of noise points; clusters identified are sensitive to all three.

8.7.8 Practical Tips

- Use the k-distance plot (`kNNdistplot()`) to select ϵ; a clear knee in the sorted distance curve marks a sensible threshold.
- DBSCAN is sensitive to ϵ, especially with clusters of widely different densities; use OPTICS (`dbscan::optics()`) or HDBSCAN (`dbscan::hdbscan()`) when density varies across clusters.
- Standardise variables before clustering; distance-based methods misbehave when scales differ across columns.
- Treat noise points explicitly in downstream analysis; do not force them into the nearest cluster, as that defeats DBSCAN's main advantage over k-means.
- For high-dimensional data, distances concentrate (the "curse of dimensionality") and DBSCAN loses discriminating power; reduce dimensionality first via PCA or UMAP.
- HDBSCAN is generally more robust than DBSCAN — it avoids the ϵ choice and handles varying densities — and is increasingly preferred for new applications.

8.7.9 R Packages Used

`dbscan` for `dbscan()`, `optics()`, `hdbscan()`, and the k-distance heuristic; `fpc::dbscan()` for an alternative implementation with similar API; `factoextra::fviz_cluster()` for ggplot-style cluster visualisation; `cluster::clusterboot()` for stability-based validation of the resulting clusters.

8.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.7.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based

- UMAP and t-SNE

8.8 Distance and Dissimilarity Measures

8.8.1 Introduction

The choice of distance or dissimilarity is the foundation of most multivariate methods: clustering algorithms (k-means, hierarchical, PAM, DBSCAN) consume a distance, multidimensional scaling and PCoA produce coordinates that approximate one, and k-nearest neighbours computes them at every prediction. Different metrics capture different notions of proximity, and the right choice depends on the data type — continuous vs. categorical, abundances vs. presence/absence, magnitudes vs. directions — and on the inferential question. Mismatched metrics produce results that look correct but answer the wrong question.

8.8.2 Prerequisites

A working understanding of vectors, the difference between distance and dissimilarity (metric vs. semi-metric), and the most common multivariate methods that consume distance matrices.

8.8.3 Theory

Standard continuous distances between p-dimensional vectors:

- **Euclidean**: $\sqrt{\sum_i (x_i - y_i)^2}$ — straight-line distance, the default for most uses.
- **Manhattan (L_1)**: $\sum_i |x_i - y_i|$ — robust to outliers, appropriate for grid-like geometry.
- **Minkowski (L_p)**: $(\sum_i |x_i - y_i|^p)^{1/p}$ — generalises Euclidean ($p = 2$) and Manhattan ($p = 1$).
- **Canberra**: $\sum_i |x_i - y_i| / (|x_i| + |y_i|)$ — sensitive to small values; useful for compositional data.
- **Cosine**: $1 - (x \cdot y)/(\|x\|\|y\|)$ — measures direction, not magnitude; standard in text and gene-expression contexts.
- **Mahalanobis**: scale- and correlation-aware, weights the deviation by the inverse covariance matrix.

For categorical or mixed data: Gower distance handles mixed types via per-variable scaling; Jaccard and Bray-Curtis suit presence-absence and abundance data respectively.

8.8.4 Assumptions

Metrics satisfy the triangle inequality and symmetry; dissimilarities may relax these. Some downstream methods (classical MDS, k-means) implicitly assume

metric distances; others (NMDS, hierarchical clustering with non-Euclidean distances) handle dissimilarities gracefully.

8.8.5 R Implementation

```r
library(cluster)

x <- rbind(c(1, 2, 3), c(2, 4, 5), c(5, 0, 1))

dist(x, method = "euclidean")
dist(x, method = "manhattan")
dist(x, method = "minkowski", p = 3)
dist(x, method = "canberra")

# Gower for mixed data
d <- data.frame(num = 1:3, cat = c("A", "B", "A"))
daisy(d, metric = "gower")
```

8.8.6 Output & Results

`dist()` returns a `dist` object containing the lower-triangular pairwise distance matrix. Different methods produce numerically different matrices that emphasise different features of the geometry. `cluster::daisy()` extends this to mixed numeric-categorical data via Gower distance.

8.8.7 Interpretation

A reporting sentence: "Euclidean distances of 3.0 between rows 1 and 2 and 5.7 between rows 1 and 3 reflect continuous-valued separation; the Gower distance on mixed data scaled each variable to $[0, 1]$ before combining and produced a unitless dissimilarity in the same range." Always state the metric used; clustering or ordination output is uninterpretable without it.

8.8.8 Practical Tips

- Standardise variables (`scale()`) before Euclidean distance whenever scales differ across columns; otherwise large-variance variables dominate the distance and bias every downstream method.
- For mixed numeric and categorical data, Gower is the standard; it scales each variable to $[0, 1]$ via type-specific rules and combines them as a weighted average.
- For binary presence/absence, prefer Jaccard or Sørensen-Dice (which ignore joint absences) over Hamming (which counts all matches).
- For sparse high-dimensional data (text, gene expression), cosine distance is often more meaningful than Euclidean because magnitudes vary widely while direction encodes the signal.

- Mahalanobis is essential when predictor correlations matter; it underlies the discriminant function in LDA and Hotelling's T^2.
- Always match the metric to the data type and downstream method; Euclidean on mixed scales or on presence/absence data is a frequent mistake that invalidates clustering results.

8.8.9 R Packages Used

Base R `dist()` for the standard continuous metrics; `cluster::daisy()` for Gower distance; `vegan::vegdist()` for ecological distances (Bray-Curtis, Jaccard, Hellinger, Canberra); `philentropy::distance()` for an extensive catalogue of distance and divergence functions; `proxy::dist()` and `parallelDist::parDist()` for multi-core distance computation on large matrices.

8.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.8.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.9 The Elbow Method

8.9.1 Introduction

The elbow method is the simplest heuristic for choosing the number of clusters in k-means or any partitioning algorithm. The within-cluster sum of squares (WSS) decreases monotonically with k, but the rate of decrease slows once the natural cluster structure has been captured. Plotting WSS against k reveals a "bend" — the elbow — at the point where additional clusters bring diminishing returns. The method is unavoidably subjective but widely used because it is fast, intuitive, and works well when the data have well-separated clusters.

8.9.2 Prerequisites

A working understanding of k-means clustering, the within-cluster sum of squares as a quality criterion, and the relationship between k and within-cluster compactness.

8.9.3 Theory

For partition $C = \{C_1, \ldots, C_k\}$ of the data, $\text{WSS}(k) = \sum_{j=1}^{k} \sum_{x \in C_j} \|x - \bar{x}_j\|^2$ is the total within-cluster sum of squared distances to centroids. As k grows, WSS necessarily decreases — at $k = n$ it reaches zero. A genuine cluster structure produces a sharp bend at the true number of clusters, after which adding more k subdivides homogeneous regions and yields only modest reductions. The bend is identified visually or, more reproducibly, by algorithms such as Kneedle that compute a normalised distance from each point on the curve to a chord between the endpoints.

8.9.4 Assumptions

A genuine cluster structure exists in the data; if no clusters are present, no clear elbow appears. Variables are standardised so distances along each axis are comparable.

8.9.5 R Implementation

```r
library(factoextra)

X <- scale(iris[, 1:4])

# Elbow plot
fviz_nbclust(X, kmeans, method = "wss", k.max = 10)

# Manual: compute WSS across k
wss <- sapply(1:10, function(k)
  kmeans(X, centers = k, nstart = 25)$tot.withinss)
plot(1:10, wss, type = "b",
     xlab = "Number of clusters k",
     ylab = "Within-cluster sum of squares")
```

8.9.6 Output & Results

A monotone decreasing WSS curve. For the standardised iris data, the bend appears at $k = 3$, consistent with the three species. `fviz_nbclust()` automates the standard elbow plot with `ggplot2` styling and works with k-means, PAM, and CLARA via the `FUNcluster` argument.

8.9.7 Interpretation

A reporting sentence: "The elbow plot showed a clear bend at $k = 3$ (within-cluster sum of squares dropped from 681 at $k = 1$ to 140 at $k = 3$ and only marginally further to 96 at $k = 10$); we chose $k = 3$, consistent with the silhouette method's optimum and with the known species structure." Always couple

the elbow visual with at least one quantitative criterion (silhouette, gap statistic).

8.9.8 Practical Tips

- Pair the elbow with the silhouette method (`fviz_nbclust(...,method="silhouette")`) and the gap statistic (`method="gap_stat"`); concordance across all three increases confidence in the chosen k.
- If no clear elbow appears, the data may not have a natural cluster structure; do not force a k where the WSS curve is smooth.
- Standardise variables before clustering; variables with larger variance dominate WSS without standardisation and distort the elbow.
- Use `nstart=25` or higher when computing WSS; multiple random starts avoid local minima that produce noisy WSS curves.
- The Kneedle algorithm (R package `kneedle`) automates elbow detection by finding the point of maximum curvature; useful for batch processing many datasets.
- Elbow alone is insufficient justification for a chosen k in publications; report it alongside silhouette and gap-statistic results.

8.9.9 R Packages Used

`factoextra` for `fviz_nbclust()` ggplot-style elbow, silhouette, and gap-statistic plots; `cluster` for the underlying clustering algorithms; `NbClust` for 30+ cluster-validation indices in one call; `kneedle` for automated elbow detection.

8.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.9.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.10 Confirmatory Factor Analysis

8.10.1 Introduction

Confirmatory factor analysis (CFA) tests a pre-specified factor structure against data: which items load on which factors, which loadings are constrained to zero,

and how factors correlate with each other. It is the measurement-model portion of structural equation modelling and the standard tool for validating questionnaire structures, latent-trait models in psychometrics, and multi-indicator latent constructs in clinical and behavioural research. Unlike exploratory factor analysis, CFA does not estimate which items belong to which factor — that decision is in the model — and instead estimates only the strength of the predetermined relationships and tests model-data fit.

8.10.2 Prerequisites

A working understanding of exploratory factor analysis, regression, and the `lavaan` model-syntax conventions for specifying latent variables.

8.10.3 Theory

CFA specifies a complete pattern matrix in advance: each item either loads freely on its assigned factor or is constrained to zero on every other factor. Fit is assessed via:

- **Chi-squared test**: a non-significant χ^2 formally accepts the model, but the test rejects too easily at large n.
- **Incremental fit indices**: CFI and TLI compare to a baseline null model; values above 0.95 indicate good fit, 0.90–0.95 acceptable.
- **Absolute fit**: RMSEA measures discrepancy per degree of freedom; below 0.06 is good, 0.06–0.08 acceptable. The accompanying 90 % CI quantifies precision.
- **SRMR**: standardised root-mean-square residual; below 0.08 indicates good fit.

A well-fitting model satisfies multiple criteria simultaneously — no single index is sufficient.

8.10.4 Assumptions

A correctly specified factor structure, multivariate Normal continuous indicators (or robust / WLSMV estimators for non-Normal or ordinal data), and sufficient sample size (rule of thumb: at least 10 observations per estimated parameter, or $n \geq 200$).

8.10.5 R Implementation

```
library(lavaan); library(semTools)

# Three-factor model on HolzingerSwineford1939 dataset
model <- '
  visual  =~ x1 + x2 + x3
  textual =~ x4 + x5 + x6
```

```
  speed   =~ x7 + x8 + x9
'
fit <- cfa(model, data = HolzingerSwineford1939)
summary(fit, fit.measures = TRUE, standardized = TRUE)

# Modification indices for potential improvements
head(modificationIndices(fit, sort = TRUE), 10)
```

8.10.6 Output & Results

`summary(fit,fit.measures=TRUE,standardized=TRUE)` reports unstandardised and standardised loadings, factor correlations, residual variances, and the full panel of fit indices (chi-squared, CFI, TLI, RMSEA with CI, SRMR). `modificationIndices()` lists potential parameter additions ordered by expected chi-squared improvement.

8.10.7 Interpretation

A reporting sentence: "The three-factor CFA fit the data adequately ($\chi^2(24) = 85.3$, $p < 0.001$; CFI 0.93; RMSEA 0.092 [90 % CI 0.071 to 0.114]; SRMR 0.065); all standardised loadings exceeded 0.50 and were statistically significant ($p < 0.001$). RMSEA exceeded the conventional 0.06 threshold, suggesting modest local misspecification — modification indices indicated a residual correlation between x_7 and x_9 that would improve fit if substantively justified." Always report multiple fit indices; no single one is decisive.

8.10.8 Practical Tips

- Do not follow modification indices blindly; each unconstrained parameter moves the analysis from confirmatory to exploratory and inflates Type I error if not pre-specified.
- For ordinal Likert items, use diagonally-weighted least squares (`cfa(...,estimator="DWLS")` or `"WLSMV"`); ordinary ML on Likert items violates Normality assumptions.
- For non-Normal continuous data, use robust ML (`estimator="MLR"`); standard errors and chi-squared are corrected for non-Normality.
- Measurement-invariance testing across groups (configural, metric, scalar invariance) is a standard follow-up before comparing latent means; use `semTools::measurementInvariance()`.
- Report chi-squared with degrees of freedom and the full set of fit indices; never report a single index alone.
- Standardised loadings (factor-loading correlations) communicate effect size more clearly than unstandardised; always include them.

8.10.9 R Packages Used

`lavaan` for `cfa()` and `sem()` with multiple estimators; `semTools` for measurement-invariance testing, reliability coefficients, and modification heuristics; `lavaanPlot` and `semPlot` for path-diagram visualisation; `blavaan` for Bayesian CFA when small samples or complex priors require it.

8.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.10.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.11 Exploratory Factor Analysis

8.11.1 Introduction

Exploratory factor analysis (EFA) searches for a small number of latent factors that explain the correlations among a larger set of observed variables. It is the cornerstone of psychometric scale development, the standard tool for validating questionnaire structures, and a useful exploratory technique in any context where many correlated indicators are believed to reflect a smaller number of latent constructs. The output — rotated factor loadings together with factor correlations — directly informs item-selection, scale-construction, and CFA model-specification decisions.

8.11.2 Prerequisites

A working understanding of correlation matrices, principal components analysis, and the conceptual difference between observed indicators and latent constructs.

8.11.3 Theory

The common factor model is

$$X = \Lambda F + U,$$

where X is the observed item matrix, Λ is the loading matrix, F are the common factors, and U is unique (specific) variance. Unlike PCA, which lumps common and specific variance together, EFA explicitly partitions the variance.

The standard EFA workflow has five steps:

1. **Check factorability**: Kaiser-Meyer-Olkin (KMO > 0.6 acceptable, > 0.8 good) and Bartlett's sphericity test confirm that the correlation matrix has enough structure to factor.
2. **Choose number of factors**: parallel analysis is the modern gold standard; the obsolete Kaiser eigenvalue-greater-than-1 rule and the subjective scree plot are still seen but should be supplemented.
3. **Extract**: maximum likelihood (ML) for inferential purposes, principal-axis factoring (PAF) for robustness, principal-components for quick exploration.
4. **Rotate**: varimax (orthogonal) or oblimin / promax (oblique) for interpretable loadings.
5. **Interpret**: examine rotated loadings, factor correlations, and item communalities.

8.11.4 Assumptions

Multivariate Normal continuous data for ML extraction; linearity of item-factor relationships; sufficient sample size (rule of thumb: at least 5–10 cases per item, with $n \geq 200$ as a floor).

8.11.5 R Implementation

```r
library(psych); library(GPArotation)

d <- bfi[, 1:25]   # 25-item big-five inventory

# Factorability
KMO(d)
cortest.bartlett(cor(d, use = "pairwise"), n = nrow(d))

# Number of factors
fa.parallel(d, fa = "fa")

# EFA with 5 factors, oblique rotation
efa <- fa(d, nfactors = 5, rotate = "oblimin", fm = "ml")
print(efa, cut = 0.3)
```

8.11.6 Output & Results

`KMO()` returns overall and per-item factorability indices; `cortest.bartlett()` returns the sphericity test. `fa.parallel()` overlays observed eigenvalues on simulated null eigenvalues to determine the number of factors. `fa()` returns rotated loadings, communalities, factor correlations (under oblique rotation), and explained variance per factor.

8.11.7 Interpretation

A reporting sentence: "EFA on 25 personality items (n = 2,800; KMO = 0.85, Bartlett's $\chi^2 = 18{,}245$, $p < 0.001$) supported a five-factor solution by parallel analysis; oblimin-rotated loadings aligned with the Big Five structure (extraversion, neuroticism, conscientiousness, agreeableness, openness), explaining 45 % of common variance, and factor correlations ranged from 0.15 to 0.42." Always report KMO, Bartlett, the number-of-factors method, the extraction method, and the rotation.

8.11.8 Practical Tips

- Use parallel analysis (`fa.parallel`) for the number-of-factors decision; avoid the obsolete "eigenvalue > 1" Kaiser rule which over-extracts in most settings.
- Prefer oblique rotation (oblimin, promax) by default; uncorrelated factors are rare in real psychological or biological data.
- Watch for Heywood cases — loadings above 1 or communalities above 1 — which indicate over-extraction or model misspecification.
- For ordinal Likert items, use polychoric correlations (`fa(...,cor="poly")`); Pearson correlations on Likert items underestimate the true factor structure.
- For confirmatory tests of the resulting structure, run a separate CFA (`lavaan::cfa`) on a held-out sample; never confirm on the data used to extract.
- Report item communalities alongside loadings; communalities below 0.20 indicate items that contribute little to any factor and may need removal.

8.11.9 R Packages Used

`psych` for `fa()`, `KMO()`, `fa.parallel()`, and the comprehensive EFA workflow; `GPArotation` for the underlying rotation algorithms; `EFAtools` for parallel-analysis-aware extraction with bootstrap confidence intervals; `lavaan` for confirmatory factor analysis as the natural follow-up; `MBESS` for additional reliability and structural-equation utilities.

8.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.11.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.12 Factor Rotation

8.12.1 Introduction

After factor extraction, rotation transforms the loading matrix to make interpretation easier without changing model fit. The mathematical reason is that any rotation of the factor solution produces the same predicted covariance matrix; rotation merely chooses a particular representation among the infinitely many that fit equally well. The choice of rotation method — orthogonal versus oblique, varimax versus promax versus oblimin — substantially affects which interpretation a reader will reach, so disclosing it is part of any responsible EFA report.

8.12.2 Prerequisites

A working understanding of exploratory factor analysis, factor loadings, and the distinction between orthogonal and oblique factor structures.

8.12.3 Theory

Orthogonal rotations keep factors uncorrelated; the rotated loading matrix retains the property that factors do not share variance.

- **Varimax**: maximises the variance of squared loadings within each column, producing "simple structure" where each variable loads strongly on one factor and weakly on others.
- **Quartimax**: maximises variance per row; tends to produce a single general factor.

Oblique rotations allow factors to correlate; this is more realistic for most psychological, biological, and social phenomena where underlying constructs are not artificially independent.

- **Oblimin**: direct minimisation of cross-factor loadings under controlled obliquity.
- **Promax**: applies a power transformation to a varimax solution to produce an oblique structure.

- **Geomin**: alternative oblique rotation popular in IRT contexts.

Oblique rotations produce both a *pattern* matrix (regression-like loadings of variables on factors) and a *structure* matrix (correlations between variables and factors); the two coincide only under orthogonal rotation.

8.12.4 Assumptions

Rotation leaves fit unchanged; only interpretation changes. The chosen rotation respects the substantive theory about whether factors should correlate.

8.12.5 R Implementation

```r
library(psych); library(GPArotation)

d <- bfi[, 1:25]

# Varimax (orthogonal)
efa_v <- fa(d, nfactors = 5, rotate = "varimax", fm = "ml")
print(efa_v, cut = 0.3)

# Oblimin (oblique)
efa_ob <- fa(d, nfactors = 5, rotate = "oblimin", fm = "ml")
print(efa_ob, cut = 0.3)

# Factor correlations after oblique rotation
efa_ob$Phi
```

8.12.6 Output & Results

`fa()` returns a rotated loading matrix; under oblique rotation, `$Phi` contains the factor correlation matrix. Setting `cut=0.3` in `print()` suppresses small loadings for cleaner display. Comparing varimax and oblimin loadings on the same data shows how dramatically the interpretation can shift.

8.12.7 Interpretation

A reporting sentence: “Oblimin-rotated factors showed moderate correlations (range 0.15 to 0.42), justifying oblique rotation; the resulting pattern matrix gave clear simple structure with loadings above 0.40 aligning with the hypothesised Big Five domains and cross-loadings predominantly below 0.20.” Always report factor correlations under oblique rotation; they justify the choice and constrain interpretation.

8.12.8 Practical Tips

- Oblique rotation (oblimin or promax) is the default in modern psychometrics; orthogonal rotation should be justified by genuine reason to expect uncorrelated factors.
- Report factor correlations from the Phi matrix; if all are below 0.20, an orthogonal solution may be acceptable.
- Varimax produces simpler-looking loading structures but can be misleading when factors truly correlate; the apparent simplicity may hide systematic cross-loadings.
- Quartimax tends to produce a strong general factor; useful when a hierarchical structure is expected (Spearman's g, hierarchical personality models).
- Rotation is a numerical optimisation; start from multiple random initialisations to confirm the solution is stable.
- `psych::fa.diagram()` visualises the rotated structure as a path diagram, often easier to communicate than a loadings table.

8.12.9 R Packages Used

`psych` for `fa()` and the broad EFA / CFA workflow with rotation options; `GPArotation` for the underlying rotation algorithms (oblimin, promax, geomin, simplimax); `lavaan` for confirmatory factor analysis as the natural follow-up; `EFAtools` for parallel-analysis-aware EFA workflows.

8.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.12.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.13 The Gap Statistic

8.13.1 Introduction

The gap statistic, introduced by Tibshirani, Walther, and Hastie in 2001, is a principled way to choose the number of clusters k in any partitioning method. Where the elbow method relies on visual judgement and the silhouette method

on per-observation neighbour comparisons, the gap statistic compares the within-cluster dispersion of the data to what would be expected under a null reference distribution with no cluster structure. The point at which the data deviate most from the null is the optimal k, with bootstrap-based standard errors that quantify the certainty of the choice.

8.13.2 Prerequisites

A working understanding of clustering algorithms, within-cluster sum of squares, and bootstrap resampling for standard error estimation.

8.13.3 Theory

For each candidate k, define W_k as the log of the within-cluster dispersion (sum of squared distances to centroids) on the actual data after running k-means or another partitioning algorithm. Generate B datasets from a null reference — uniformly distributed over the data's bounding box, or over its principal-components box — and compute the average within-cluster dispersion $\mathbb{E}[W_k^*]$ under the null. The gap statistic is

$$\text{Gap}(k) = \mathbb{E}[W_k^*] - W_k.$$

The optimal k is the smallest value satisfying $\text{Gap}(k) \geq \text{Gap}(k+1) - s_{k+1}$, where s_{k+1} is the standard deviation of the null reference at $k+1$ (the so-called "first-SE-max" criterion). This rule rewards parsimonious choices: it picks the smallest k that is not significantly worse than larger values.

8.13.4 Assumptions

A meaningful distance metric on the data; the null reference distribution is sensible (uniform over the bounding box is reasonable for low-to-moderate dimensions; PC-aligned bounding boxes are better in high dimensions).

8.13.5 R Implementation

```
library(cluster); library(factoextra)

X <- scale(iris[, 1:4])

set.seed(2026)
gap <- clusGap(X, FUN = kmeans, nstart = 25, K.max = 10, B = 100)
print(gap, method = "firstSEmax")

fviz_gap_stat(gap)
```

8.13.6 Output & Results

`clusGap()` returns a matrix with W_k, $\mathbb{E}[W_k^*]$, the gap value, and its bootstrap standard error for each k. `print(...,method="firstSEmax")` applies Tibshirani's selection rule to identify the optimal k. `fviz_gap_stat()` plots the gap curve with error bars.

8.13.7 Interpretation

A reporting sentence: "The gap statistic, computed with $B = 100$ bootstrap replicates against a uniform reference, identified $k = 3$ as the optimal number of clusters using the firstSEmax criterion ($\text{Gap}(3) = 0.59$, $\text{Gap}(4) - s_4 = 0.55$); this matches the underlying species structure in the iris data." Report the criterion explicitly (firstSEmax, Tibs2001SEmax, globalmax); they sometimes disagree.

8.13.8 Practical Tips

- Use $B = 100$ to 500 bootstrap replicates; fewer can produce unstable optima, more is rarely worth the runtime cost.
- The default uniform-bounding-box reference works well in low dimensions; in high dimensions, switch to a PC-aligned reference (`spaceH0="scaledPCA"`).
- Combine the gap statistic with elbow and silhouette methods; concordance across all three is the strongest argument for a chosen k.
- Gap performs best on well-separated clusters; for overlapping or hierarchical structures it can favour very large k or fail to find a clear optimum.
- For very large datasets, subsample before applying the gap statistic; the bootstrap step otherwise dominates runtime.
- Selection criteria differ: `firstSEmax` (Tibshirani's parsimonious choice), `globalmax` (the absolute maximum, often larger), and `Tibs2001SEmax` (the original 2001 rule). Quote the criterion in any report.

8.13.9 R Packages Used

`cluster` for `clusGap()` and the underlying clustering algorithms; `factoextra` for `fviz_gap_stat()` and `fviz_nbclust()` ggplot-style visualisation; `NbClust` for 30+ alternative cluster-validation indices in one call.

8.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.13.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.14 Gaussian Mixture Models

8.14.1 Introduction

Gaussian mixture models (GMMs) represent the data as a weighted sum of several multivariate Normal distributions, each component characterising a latent cluster. Unlike k-means, which is essentially a hard-assignment heuristic, GMMs are fully probabilistic generative models: each observation has a posterior probability of belonging to each cluster, the clusters can have different shapes (covariance structures), and model-selection criteria (BIC) handle the choice of cluster count without ad-hoc heuristics. The `mclust` package enumerates a small library of covariance structures and selects the best automatically.

8.14.2 Prerequisites

A working understanding of the multivariate Normal distribution, the EM algorithm for latent-variable models, and the BIC as a model-selection criterion.

8.14.3 Theory

The mixture density is

$$p(x) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(x \mid \mu_k, \Sigma_k),$$

with mixing proportions $\pi_k \geq 0$ summing to 1. Maximum-likelihood estimation alternates an E-step (compute posterior responsibilities $\gamma_{ik} = \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)/p(x_i)$) and an M-step (update π_k, μ_k, Σ_k from the responsibilities). `mclust` parameterises the covariance via 14 structures from spherical-equal-volume to fully-unconstrained-different-shape and orientation, indexed by three letters: volume, shape, orientation, each Equal or Varying. BIC selects the best combination of structure and component count.

8.14.4 Assumptions

Multivariate Normal mixture components, independent observations, and a sample size large enough to estimate component covariances stably (the most flexible models require $\Omega(Kp^2)$ observations for K clusters in p dimensions).

8.14.5 R Implementation

```r
library(mclust)

X <- iris[, 1:4]
fit <- Mclust(X)
summary(fit)

# Soft assignments
head(fit$z)

# Plot
plot(fit, what = "classification")

# BIC across models and K
plot(fit, what = "BIC")
```

8.14.6 Output & Results

`Mclust()` returns the best model code (e.g. "VEV"), the number of components, BIC values across the model space, the fitted parameters, and the matrix of soft assignment probabilities `fit$z`. `plot(fit,what="BIC")` shows BIC against component count for each covariance structure, making the selection rationale visible.

8.14.7 Interpretation

A reporting sentence: "`mclust` selected a VEV (varying volume, equal shape, varying orientation) model with three components (BIC $= -560.4$, ΔBIC to the second-best model 18.2); the three components closely match the iris species (overall accuracy 97 %), and the soft assignments quantify uncertainty for the seven observations classified differently from their species labels." Always report the chosen covariance structure and BIC value.

8.14.8 Practical Tips

- Use BIC for model selection across K and covariance structure jointly; comparing K at fixed structure or vice versa misses the joint optimum.
- Soft assignments (`fit$z`) are useful for downstream uncertainty propagation; rather than committing to one cluster label, propagate the posterior probabilities into subsequent analyses.
- For high-dimensional data ($p > n$), regularised mixture models (`HDclassif`, `pgmm`) or variational EM are needed; standard `mclust` is unstable when Σ_k has more parameters than observations.
- Outliers can distort the Normal components; consider t-mixture models (`teigen::teigen`) for heavier-tailed components, or trimmed EM (`tclust`)

for robustness.
- Log-likelihood can become unbounded for unconstrained covariances when a component covers a single observation; `mclust` adds a small prior to prevent this, but watch for warnings.
- For very large n, mini-batch EM and online variants scale to millions of observations, but the `mclust` defaults handle up to a few thousand comfortably.

8.14.9 R Packages Used

`mclust` for `Mclust()`, BIC selection across the 14-model library, and visualisation; `flexmix` for finite mixtures of regressions and other generalised mixture models; `teigen` for t-mixture models robust to outliers; `tclust` for trimmed mixture models with explicit contamination handling.

8.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.14.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.15 Hierarchical Clustering

8.15.1 Introduction

Hierarchical clustering builds a nested family of partitions: at the bottom every observation is its own cluster, at the top all observations form a single cluster, and intermediate levels correspond to all the partitions you might consider. Agglomerative methods (the practical default) merge the two closest clusters at each step until everything is connected; divisive methods reverse the process. The output is a dendrogram — a tree that can be cut at any height to produce a partition with the corresponding number of clusters. The dendrogram is the headline visual that distinguishes hierarchical clustering from partitioning methods.

8.15.2 Prerequisites

A working understanding of distance metrics, the difference between distance between observations and distance between clusters, and the dendrogram as a graphical representation of the merging sequence.

8.15.3 Theory

Linkage criteria define the distance between two clusters A and B:

- **Single linkage**: $\min_{a \in A, b \in B} d(a, b)$. Prone to chaining: it joins clusters by their nearest pair, producing long stringy structures.
- **Complete linkage**: $\max_{a \in A, b \in B} d(a, b)$. Produces compact, roughly equal-size clusters.
- **Average linkage**: average of pairwise distances; intermediate behaviour.
- **Ward's linkage**: at each merge, choose the pair that minimises the increase in within-cluster sum of squares. Produces compact spherical clusters and is the most popular default.

The dendrogram is cut at a user-chosen height (or at k clusters via `cutree()`); the height of the cut is itself a diagnostic — short merges below the cut indicate tight clusters, tall merges above suggest meaningful separation.

8.15.4 Assumptions

A meaningful distance metric on the data and roughly independent observations. The chosen linkage criterion matches the expected cluster geometry; Ward's assumes Euclidean distance.

8.15.5 R Implementation

```r
library(factoextra)

X <- scale(iris[, 1:4])
d <- dist(X)

hc_ward <- hclust(d, method = "ward.D2")
fviz_dend(hc_ward, k = 3, rect = TRUE, palette = "Set2")

# Compare linkages
par(mfrow = c(2, 2))
for (m in c("single", "complete", "average", "ward.D2")) {
  plot(hclust(d, method = m), main = m, hang = -1)
}

# Cluster assignments at k = 3
cutree(hc_ward, k = 3)
```

8.15.6 Output & Results

`hclust()` returns the merging sequence and per-merge heights as an `hclust` object. `fviz_dend()` plots the dendrogram with cluster boxes and a colour palette. `cutree()` extracts cluster assignments at a chosen k. The four-panel comparison shows how dramatically linkage choice changes the resulting tree.

8.15.7 Interpretation

A reporting sentence: "Ward-linkage agglomerative clustering on standardised iris data, cut at $k = 3$, produced three clusters matching the species labels with 95 % agreement; single linkage produced a chained 1-iris-vs-149 partition that is uninformative for this dataset, illustrating the linkage choice's importance." Always disclose the linkage method and the distance metric used.

8.15.8 Practical Tips

- Ward's linkage (`ward.D2`) is the most reliable default for spherical clusters of similar size; it agrees with k-means in spirit but produces a tree.
- Single linkage often produces stringy chains; reach for it only when chain-like structures are substantively meaningful (e.g. minimum spanning trees in network analysis).
- Visualise the dendrogram before choosing k; the height of merges above the cut suggests meaningful versus arbitrary splits.
- For very large datasets, hierarchical clustering is $O(n^2)$ in memory and time; switch to k-means or DBSCAN above tens of thousands of observations.
- For mixed numeric-categorical data, compute Gower distance with `cluster::daisy()` and feed it to `hclust()`; Ward's then needs `method="ward.D"` (not D2).
- Consider `cluster::agnes()` over `hclust()` for an agglomerative coefficient that quantifies the strength of the clustering structure (close to 1 = strong structure).

8.15.9 R Packages Used

Base R `hclust()` and `cutree()`; `cluster::agnes()` and `diana()` for agglomerative and divisive variants with structure indices; `factoextra` for `fviz_dend()` and `fviz_cluster()` ggplot-style visualisation; `dendextend` for advanced dendrogram manipulation including tanglegrams comparing two trees.

8.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.15.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.16 Independent Component Analysis

8.16.1 Introduction

Independent component analysis (ICA) is the canonical approach for recovering statistically independent latent signals from observed linear mixtures. The classic application is the "cocktail-party problem" — separating individual conversations from microphone recordings of overlapping speakers — but the technique generalises to any setting where multiple independent sources mix linearly into a smaller set of observed channels. EEG, fMRI, gene-expression data, and audio source separation all use ICA as a primary decomposition method.

8.16.2 Prerequisites

A working understanding of principal components analysis, statistical independence (a stronger condition than uncorrelatedness), and the concept of a linear mixing model.

8.16.3 Theory

The ICA model assumes observations X are generated as $X = AS$, where A is an unknown $n \times n$ mixing matrix and S contains the independent latent sources. The goal is to estimate A (or its inverse, the unmixing matrix W) so that $\hat{S} = WX$ recovers the sources up to permutation and scale ambiguities.

Two principles drive ICA algorithms:

- **Non-Gaussianity maximisation**: by the central limit theorem, sums of independent variables become more Gaussian; an unmixing direction that maximises non-Gaussianity (kurtosis, negentropy) recovers an independent source.
- **Statistical independence maximisation**: minimise mutual information between recovered components.

FastICA is the most widely used algorithm and uses fixed-point iteration on kurtosis or negentropy. Infomax is the alternative principle implemented in domain-specific tools.

The key contrast with PCA: PCA produces *uncorrelated* components by maximising variance; ICA produces *independent* components by maximising non-Gaussianity. The two coincide only for Gaussian data, where ICA is undefined.

8.16.4 Assumptions

Sources are statistically independent and non-Gaussian (at most one Gaussian source can be recovered); mixing is linear and instantaneous; the number of sources equals the number of observed channels (or equals the chosen number of components).

8.16.5 R Implementation

```r
library(fastICA)

set.seed(2026)
# Simulate two sources: sine and sawtooth
t <- seq(0, 8, length.out = 1000)
s1 <- sin(2 * pi * t)
s2 <- ((2 * t) %% 2) - 1      # sawtooth

# Mix
A <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
S <- cbind(s1, s2)
X <- S %*% t(A)

# Run ICA
ic <- fastICA(X, n.comp = 2)

par(mfrow = c(3, 1))
matplot(S, type = "l", main = "True sources")
matplot(X, type = "l", main = "Mixed signals")
matplot(ic$S, type = "l", main = "Recovered ICA components")
par(mfrow = c(1, 1))
```

8.16.6 Output & Results

`fastICA()` returns the recovered sources `$S`, the estimated mixing matrix `$A`, and the unmixing matrix `$W`. The three-panel plot shows the original sine and sawtooth sources, the mixed signals as observed, and the ICA-recovered components — visually almost identical to the originals up to sign and permutation.

8.16.7 Interpretation

A reporting sentence: "FastICA on two channels of mixed sine and sawtooth signals recovered both sources cleanly (visual inspection), demonstrating blind

source separation under the linear-mixing model; recovered components were rescaled to unit variance and permuted, reflecting the inherent ICA ambiguities." Mention the algorithm and pre-processing (whitening, mean-centring) in the methods section.

8.16.8 Practical Tips

- Pre-whiten the data (PCA-based decorrelation followed by unit-variance scaling) before FastICA; the `fastICA` package does this automatically.
- ICA is essentially a rotation of whitened data — order and sign of components are arbitrary; do not interpret component indices as meaningful.
- At most one Gaussian source can be recovered; if all sources are Gaussian, ICA cannot separate them.
- For fMRI and EEG, specialised implementations (MELODIC in FSL, runica in EEGLAB) handle the domain-specific challenges of large channel counts and physiological artefacts.
- For non-square mixing problems (more sources than channels), compressive ICA and multi-modal extensions exist but require specialised tools.
- ICA is unsupervised; evaluating the result requires interpreting the recovered signals substantively, often through expert judgement.

8.16.9 R Packages Used

`fastICA` for the standard FastICA implementation; `ica` for additional ICA variants including extended Infomax; `mlbench` for synthetic mixing examples; `JADE` for joint approximate diagonalisation of eigen-matrices, an alternative algebraic approach to ICA.

8.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.16.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.17 Jaccard and Dice Dissimilarity

8.17.1 Introduction

Jaccard and Dice dissimilarities are the standard distance measures for binary (presence/absence) data: ecological community composition, disease symptom panels, gene set membership, document term frequencies after thresholding. Both indices share a key property: they ignore joint absences and focus only on agreements and disagreements among presences. This is exactly what is needed when an absence carries little information — a species absent from two sites might be absent because the habitat is unsuitable, or simply not detected — but is wrong when absences are themselves diagnostic.

8.17.2 Prerequisites

A working understanding of binary data, set-based similarity reasoning, and the conventional 2×2 contingency table for two binary vectors.

8.17.3 Theory

For two binary vectors x, y with counts a (both present), b (only in x), c (only in y), d (both absent):

- **Jaccard index**: $J = a/(a + b + c)$, the ratio of the intersection to the union of the present-set. **Jaccard dissimilarity** is $1 - J$.
- **Dice / Sørensen coefficient**: $D = 2a/(2a + b + c)$, with **Dice dissimilarity** $1 - D$. Dice doubles the weight on joint presences relative to mismatches.
- **Sørensen** and **Dice** are equivalent for binary data.

Both ignore d (joint absences), making them appropriate when absences are uninformative. Indices that include d — simple matching, Hamming distance — are preferable when joint absences carry meaning (e.g., binary medical diagnoses where "neither has condition X" is informative).

8.17.4 Assumptions

Binary data (or count data thresholded to binary), and joint absences are not meaningful for the inferential question. The same data on different absence-meaning assumptions can yield very different distances.

8.17.5 R Implementation

```
library(vegan)

# Species abundance data (reduce to presence/absence)
data(dune)
```

```r
dune_pa <- (dune > 0) * 1
head(dune_pa)

# Jaccard dissimilarity
d_jac <- vegdist(dune_pa, method = "jaccard", binary = TRUE)
as.matrix(d_jac)[1:3, 1:3]

# Sorensen (= Dice)
d_sor <- vegdist(dune_pa, method = "bray", binary = TRUE)
```

8.17.6 Output & Results

Two pairwise dissimilarity matrices on the dune presence/absence data, one for Jaccard and one for Sørensen-Dice. Jaccard distances are systematically larger than Dice for the same data because Dice double-counts joint presences in its denominator.

8.17.7 Interpretation

A reporting sentence: "Jaccard dissimilarity averaged 0.40 across all pairs of sites (range 0.16 to 0.65), indicating that 40 % of species presences are not shared between any randomly chosen pair; the corresponding Sørensen-Dice dissimilarity averaged 0.25, reflecting Dice's higher weight on joint presences." Always state which index was used; the same data produce different conclusions under Jaccard versus Dice.

8.17.8 Practical Tips

- Use Jaccard or Dice when joint absences are biologically or ecologically uninformative (rare species, sparse term-document data, gene-set membership).
- For abundance rather than presence-absence data, Bray-Curtis is the standard; Bray-Curtis on binarised data reduces to Sørensen-Dice.
- For microbiome data with phylogenetic information, UniFrac (`GUniFrac::GUniFrac`) is preferable because it weights distances by evolutionary relationships, not just shared OTUs.
- For binary medical data where joint absences are meaningful (neither patient has the condition), use simple matching (`dist(x,method="binary")` for Hamming-style) rather than Jaccard.
- For very large sparse binary matrices (text, single-cell sparse counts), use the sparse-matrix-aware implementations in `proxy` or `parallelDist` to avoid memory blow-up.
- Pair dissimilarity matrices with NMDS or PCoA for visualisation; both ordinations consume Jaccard or Dice matrices directly.

8.17.9 R Packages Used

`vegan` for `vegdist()` with `method="jaccard"` or `"bray"` and the broader ecology workflow; `proxy` for general dissimilarity functions including binary metrics; `philentropy` for distance and similarity coefficients across many families; `GUniFrac` for phylogenetic UniFrac alternatives in microbiome analysis.

8.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.17.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.18 Kernel PCA

8.18.1 Introduction

Kernel principal component analysis performs PCA in a high-dimensional feature space implicitly defined by a kernel function, capturing non-linear structure that linear PCA misses. The classic example — two concentric circles — illustrates the point: linear PCA fails to separate them because no straight axis distinguishes the two classes, but kernel PCA with a radial-basis-function kernel finds the radial coordinate in an implicit infinite-dimensional space and separates the rings cleanly. The technique generalises PCA to data lying on curved manifolds while preserving its eigen-decomposition mathematical structure.

8.18.2 Prerequisites

A working understanding of principal components analysis, the kernel trick, Gram matrices, and the difference between linear and non-linear feature spaces.

8.18.3 Theory

A kernel function $K(x, y) = \langle \phi(x), \phi(y) \rangle$ defines an inner product in a feature space without ever computing ϕ explicitly (the "kernel trick"). Standard choices:

- **RBF / Gaussian**: $K(x, y) = \exp(-\gamma \|x - y\|^2)$, infinite-dimensional feature space.

- **Polynomial**: $K(x, y) = (x^\top y + c)^d$, polynomial feature space of degree d.
- **Sigmoid**: $K(x, y) = \tanh(\alpha x^\top y + c)$, related to neural-net activations.

Construct the $n \times n$ Gram matrix K, centre it in feature space, and compute its top eigenvectors. The projection of an observation onto a kernel principal component is a weighted sum of kernel evaluations against all training observations — making kernel PCA $O(n^2)$ in space and time, which becomes prohibitive for $n \gg 10{,}000$.

8.18.4 Assumptions

The kernel encodes a similarity meaningful for the inferential question; the Gram matrix is positive semi-definite (true for valid Mercer kernels); sufficient observations for stable eigendecomposition.

8.18.5 R Implementation

```r
library(kernlab)

# Data on two concentric circles (linear PCA fails)
set.seed(2026)
theta <- runif(200, 0, 2 * pi)
r <- c(rep(1, 100), rep(3, 100))
X <- cbind(r * cos(theta), r * sin(theta)) + matrix(rnorm(400, 0, 0.1),
 ↪  200)
class <- factor(r)

# Linear PCA
pca <- prcomp(X)
plot(pca$x[, 1:2], col = class, main = "Linear PCA")

# Kernel PCA with RBF
kpca <- kpca(X, kernel = "rbfdot", kpar = list(sigma = 0.1), features = 2)
plot(rotated(kpca), col = class, main = "Kernel PCA (RBF)")
```

8.18.6 Output & Results

Two side-by-side scatterplots: linear PCA leaves the two concentric rings overlapping in PC space, while kernel PCA with RBF kernel and $\sigma = 0.1$ separates them cleanly along the first kernel principal component.

8.18.7 Interpretation

A reporting sentence: "Kernel PCA with an RBF kernel ($\sigma = 0.1$) separated the two concentric rings (n = 100 each) that linear PCA could not, demonstrating that the underlying class structure is non-linear in the original feature space;

the first kernel principal component captured the radial coordinate that distinguishes the classes." When kernel PCA succeeds where linear PCA fails, name the geometric feature being captured.

8.18.8 Practical Tips

- Kernel choice and hyperparameters matter; RBF with median-heuristic bandwidth is a reasonable default, then tune by cross-validation on a downstream supervised task.
- Kernel PCA components are not directly interpretable in the original feature space; they are weighted sums of kernel evaluations against training points.
- Computation scales as $O(n^2)$ space and $O(n^3)$ time for the eigendecomposition; for $n > 10{,}000$, use Nyström approximation (`kernlab::kha`, `RSpectra` for partial eigendecomposition).
- Modern alternatives for non-linear visualisation — UMAP, t-SNE, autoencoders — often outperform kernel PCA for cluster separation in two dimensions.
- Kernel PCA is most useful as a feature engineering step for downstream supervised learning when p is very small; in $p \gg n$ regimes, sparse linear methods (PLS, lasso) are typically more efficient.
- For preserving local geometry, prefer Laplacian eigenmaps or diffusion maps; kernel PCA preserves variance in the implicit feature space, not local distances.

8.18.9 R Packages Used

`kernlab` for `kpca()` and a wide range of kernels; `RSpectra` for partial eigendecomposition of large Gram matrices; `uwot` (UMAP) and `Rtsne` for modern non-linear embedding alternatives; `dimRed` for a unified interface across kernel PCA, t-SNE, UMAP, and Isomap.

8.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.18.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.19 k-Means Clustering

8.19.1 Introduction

k-means is the workhorse of partitioning cluster analysis: assign each of n observations to one of k clusters, with the cluster centres as multivariate means and the objective being to minimise the total within-cluster sum of squared distances. It scales to large datasets, is conceptually simple, and produces interpretable centres in the original feature space. Its limitations — sensitivity to outliers, dependence on Euclidean geometry, the need to pre-specify k, and the assumption of approximately spherical equal-size clusters — are equally well known and motivate the alternatives (PAM, DBSCAN, Gaussian mixtures) in this category.

8.19.2 Prerequisites

A working understanding of distance metrics, multivariate means, and the trade-off between within-cluster compactness and between-cluster separation.

8.19.3 Theory

Lloyd's algorithm alternates two steps until convergence:

1. **Initialise** k centres, either randomly or via the k-means++ heuristic which spreads initial centres apart for better convergence.
2. **Assign** each observation to the cluster of its nearest centre.
3. **Update** each centre to the mean of its assigned observations.
4. **Repeat** until cluster assignments no longer change.

The objective is

$$\text{WSS} = \sum_{j=1}^{k} \sum_{x \in C_j} \|x - \bar{x}_j\|^2,$$

monotonically decreasing across iterations. The algorithm converges to a local optimum, so multiple random starts (`nstart>=25`) are needed to find a good solution. Choice of k uses the elbow method, silhouette analysis, gap statistic, or composite tools like `NbClust`.

8.19.4 Assumptions

Continuous variables on comparable scales (standardise first); roughly spherical clusters of similar size; Euclidean distance is meaningful. Outliers can drag means and produce unbalanced clusters; PAM is preferable when outliers are present.

8.19.5 R Implementation

```r
library(factoextra)

X <- scale(iris[, 1:4])

# 3 clusters
set.seed(2026)
km <- kmeans(X, centers = 3, nstart = 25)
km$size
km$centers

fviz_cluster(km, data = X, palette = "Set2", ggtheme = theme_minimal())

# Elbow and silhouette
fviz_nbclust(X, kmeans, method = "wss")
fviz_nbclust(X, kmeans, method = "silhouette")
```

8.19.6 Output & Results

`kmeans()` returns cluster assignments, cluster centres in the standardised feature space, total within-cluster sum of squares, between-cluster sum of squares, and convergence diagnostics. `fviz_cluster()` plots the clusters on the first two principal components with cluster ellipses; `fviz_nbclust()` automates elbow and silhouette diagnostics.

8.19.7 Interpretation

A reporting sentence: "k-means with $k = 3$ and 25 random starts partitioned the standardised iris data with total within-cluster sum of squares 140; the resulting clusters closely matched the three species (overall accuracy 89 %, with versicolor and virginica contributing the misclassifications)." Always state k, the number of starts, and a quality metric (silhouette or WSS).

8.19.8 Practical Tips

- Always standardise predictors before k-means; variables with larger variance dominate Euclidean distance and bias the partition.
- Use `nstart=25` or higher to avoid poor local optima; the algorithm is non-convex and a single random start can converge to a sub-optimal solution.
- For non-spherical clusters, switch to hierarchical clustering, DBSCAN (density-based), or Gaussian mixture models (model-based with elliptical clusters).
- For categorical data, k-means does not apply directly; use k-modes (`klaR::kmodes`) or PAM with Gower distance.

- Cross-validate the chosen k via stability measures (`fpc::clusterboot`); clusters that recur under bootstrapping are more trustworthy than those that change with each resample.
- Outliers can drag means; for outlier-contaminated data, prefer PAM (`cluster::pam`), trimmed k-means (`tclust::tkmeans`), or robust mixture models.

8.19.9 R Packages Used

Base R `kmeans()` for the standard algorithm; `factoextra` for `fviz_cluster()`, `fviz_nbclust()`, and ggplot-style diagnostics; `cluster` for related partitioning algorithms; `klaR::kmodes()` for categorical data; `fpc::clusterboot()` for stability-based cluster validation.

8.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.19.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.20 Linear Discriminant Analysis

8.20.1 Introduction

Linear discriminant analysis (LDA) is the canonical Gaussian classifier for two or more classes. Under the assumption that within-class distributions are multivariate Normal with a common covariance matrix, the optimal Bayes classifier is linear in the predictors, and the corresponding posterior probabilities are simple closed-form functions of class means and the pooled covariance. Fisher's complementary derivation gives a projection onto $K-1$ axes that maximally separates the K class centroids; this is the most useful low-dimensional visualisation for class structure.

8.20.2 Prerequisites

A working understanding of multivariate Normal distributions, classification metrics, and the Bayes-optimal classifier under known generative assumptions.

8.20.3 Theory

Assume class k has prior π_k, mean μ_k, and common covariance Σ. The Bayes-optimal log-discriminant is

$$\delta_k(x) = x^\top \Sigma^{-1} \mu_k - \tfrac{1}{2} \mu_k^\top \Sigma^{-1} \mu_k + \log \pi_k.$$

Assigning x to the class with the largest δ_k gives the LDA classifier. The decision boundary between any two classes is linear in x, which is what makes the model interpretable: classification depends on a small number of linear combinations of the predictors.

Fisher's LDA, conceptually identical, finds the directions in which the between-class variance is large relative to the within-class variance; projecting onto the top $K - 1$ directions gives the canonical low-dimensional visualisation of class structure.

8.20.4 Assumptions

Multivariate Normal within each class, equal class covariance matrices (otherwise QDA is preferred), and independent observations. Outliers can pull class means and inflate the pooled covariance, distorting the linear boundary.

8.20.5 R Implementation

```r
library(MASS)

set.seed(2026)
train_idx <- sample(nrow(iris), 100)
train <- iris[train_idx, ]; test  <- iris[-train_idx, ]

fit_lda <- lda(Species ~ ., data = train)
pred <- predict(fit_lda, test)
table(pred$class, test$Species)

# Project onto LDA axes
plot(fit_lda, col = as.integer(train$Species))
```

8.20.6 Output & Results

`lda()` returns class priors, class means, scaling matrix (coefficients of linear discriminants), and the proportion of trace each axis explains. `predict()` provides class labels and posterior probabilities. The classifier projects test observations onto the LDA axes and assigns each to the nearest class centroid in the projected space.

8.20.7 Interpretation

A reporting sentence: "Linear discriminant analysis on the iris training set classified the held-out observations with 98 % accuracy (49/50 correct); the first discriminant axis accounted for 99 % of between-group variance, indicating that a single linear combination of the four measurements largely separates the three species." Compare LDA to QDA when class covariances differ; otherwise LDA is the more efficient choice.

8.20.8 Practical Tips

- For class-specific covariance differences, switch to quadratic discriminant analysis (QDA); Box's M test (`biotools::boxM`) provides a formal check, but it is sensitive to non-Normality.
- LDA is sensitive to outliers; screen for them or use robust variants such as `MASS::lda(...,method="mve")` or `rrcov::Linda()`.
- Prior class probabilities affect the decision boundary; specify them via `prior=c(...)` when training-set proportions differ from population proportions.
- For high-dimensional data with p approaching or exceeding n, regularise via `klaR::rda()` (regularised discriminant analysis) or sparse LDA (`MASS::lda` with PCA pre-step, or `sparseLDA`).
- Multiclass alternatives include multinomial logistic regression (`nnet::multinom`) and random forests; LDA remains competitive in low-dimensional problems with Gaussian-like classes.
- Visualise the LDA projection with the first two discriminants; a 2D plot with class colours quickly shows whether classes are linearly separable.

8.20.9 R Packages Used

`MASS` for `lda()` and `qda()`; `klaR` for `rda()` (regularised) and `partimat()` for partition diagnostics; `caret` for cross-validated comparison across discriminant variants; `rrcov` for robust LDA; `sparseLDA` for high-dimensional sparse extensions.

8.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.20.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based

- UMAP and t-SNE

8.21 Mahalanobis Distance

8.21.1 Introduction

Mahalanobis distance is the multivariate generalisation of "how many standard deviations from the mean" that respects scale and correlation between variables. Where Euclidean distance treats each axis equally and ignores covariance, Mahalanobis pre-multiplies the deviation by the inverse covariance matrix, so directions with high variance contribute less and correlated directions are decorrelated before measurement. It is the foundation of multivariate outlier detection, the discriminant function in LDA and QDA, and Hotelling's T^2 statistic.

8.21.2 Prerequisites

A working understanding of covariance matrices, multivariate Normal distributions, and the chi-squared distribution.

8.21.3 Theory

For a point $x \in \mathbb{R}^p$, mean vector μ, and covariance Σ:

$$D_M(x, \mu, \Sigma) = \sqrt{(x-\mu)^\top \Sigma^{-1} (x-\mu)}.$$

Under multivariate Normality, D_M^2 follows a chi-squared distribution with p degrees of freedom. This is the basis for probabilistic outlier flagging: points with D_M^2 exceeding the $1-\alpha$ chi-squared quantile are unlikely under the null. Substituting the sample mean and sample covariance gives the standard estimator, but the sample covariance is itself influenced by outliers — leading to "masking" where extreme points hide behind each other. Robust estimators (Minimum Covariance Determinant, MCD; Minimum Volume Ellipsoid, MVE) trim the most-influential observations before estimating Σ.

8.21.4 Assumptions

Multivariate Normality for the chi-squared reference distribution; positive-definite Σ (or its pseudo-inverse if rank-deficient); sufficient sample size relative to p for stable covariance estimation.

8.21.5 R Implementation

```
library(robustbase)

X <- iris[, 1:4]
```

```r
mu <- colMeans(X)
Sigma <- cov(X)
MD2 <- mahalanobis(X, mu, Sigma)

# Threshold at 97.5% chi-squared quantile
cutoff <- qchisq(0.975, df = 4)
which(MD2 > cutoff)

# Robust version (MCD)
mcd <- covMcd(X)
MD2_rob <- mahalanobis(X, mcd$center, mcd$cov)
which(MD2_rob > cutoff)
```

8.21.6 Output & Results

`mahalanobis()` returns squared distances under the supplied centre and covariance. Indexing by the chi-squared cutoff identifies candidate outliers. The robust MCD-based version typically flags more outliers because the classical estimator is itself distorted by the very points it should flag.

8.21.7 Interpretation

A reporting sentence: "Classical Mahalanobis distance flagged 4 iris observations above the χ^2_4 97.5 % cutoff, while the robust MCD-based estimator flagged 12; the additional observations were masked by their joint influence on the classical covariance estimate, illustrating why robust methods are preferable for outlier diagnostics in non-trivial multivariate data." Always pair classical and robust Mahalanobis when reporting outliers; agreement increases confidence, disagreement signals masking.

8.21.8 Practical Tips

- Classical Mahalanobis suffers from masking — outliers distort the very covariance used to detect them. Always pair with MCD or MVE for any non-trivial outlier-detection task.
- For high-dimensional data ($p \gg n$), covariance is rank-deficient; use regularised estimators (`corpcor::cov.shrink`) or first reduce dimensionality.
- Mahalanobis distance underlies LDA and QDA: the discriminant function is the Mahalanobis distance to each class centre, weighted by class priors.
- For multivariate Normality assessment, plot ordered Mahalanobis distances against chi-squared quantiles; deviations from the diagonal indicate non-Normality or outliers.
- The cutoff χ^2_p assumes Normality; for non-Normal data, calibrate the cutoff via simulation or use a robust universal cutoff.

- Use the squared form D_M^2 in reporting; it has the chi-squared reference distribution and avoids the square-root computation.

8.21.9 R Packages Used

Base R `mahalanobis()`; `robustbase::covMcd()` for MCD-based robust covariance; `MASS::cov.mve()` for MVE; `rrcov::CovMcd()` and `CovMve()` for additional robust estimators with formal classes; `mvoutlier::aq.plot()` for adjusted quantile plots that compare classical and robust Mahalanobis distances.

8.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.21.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.22 MANOVA

8.22.1 Introduction

MANOVA — multivariate analysis of variance — tests whether group means differ across several dependent variables simultaneously, accounting for the correlations among the outcomes. It is more powerful than running separate one-way ANOVAs when outcomes are correlated, because it pools information across the response set instead of testing each in isolation. It also controls Type I error at the family-wise level for the joint hypothesis, removing the need for multiple-comparison correction across a battery of univariate ANOVAs.

8.22.2 Prerequisites

A working understanding of one-way ANOVA, multivariate Normal distributions, and covariance matrices.

8.22.3 Theory

MANOVA decomposes the total cross-product matrix into hypothesis (H) and error (E) components and constructs four standard multivariate test statistics from their joint structure:

- **Wilks' lambda**: $\Lambda = |E|/|H+E|$, the classical choice; small values reject the null.
- **Pillai's trace**: $V = \text{tr}(H(H+E)^{-1})$, robust to covariance heterogeneity and the recommended default.
- **Hotelling–Lawley trace**: $T = \text{tr}(HE^{-1})$, more powerful when one dimension dominates.
- **Roy's greatest root**: $\theta =$ largest eigenvalue of HE^{-1}, most powerful when group differences are concentrated in one direction but liberal otherwise.

All four test the same omnibus null hypothesis but have slightly different operating characteristics under non-Normality and unequal covariances.

8.22.4 Assumptions

Multivariate Normal within groups, homogeneous covariance matrices across groups (Box's M test), and independent observations.

8.22.5 R Implementation

```
fit <- manova(cbind(Sepal.Length, Sepal.Width, Petal.Length,
 ↪  Petal.Width) ~
              Species, data = iris)
summary(fit)                      # Pillai default
summary(fit, test = "Wilks")
summary(fit, test = "Hotelling-Lawley")

# Follow-up univariate ANOVAs
summary.aov(fit)

# car Anova for Type II/III
library(car)
Manova(lm(cbind(Sepal.Length, Sepal.Width) ~ Species, data = iris))
```

8.22.6 Output & Results

`summary(fit)` returns the Pillai trace by default, with F approximation, degrees of freedom, and p-value. Specifying other tests via the `test` argument produces Wilks, Hotelling-Lawley, and Roy variants. `summary.aov(fit)` runs follow-up univariate ANOVAs on each outcome.

8.22.7 Interpretation

A reporting sentence: "MANOVA showed a significant multivariate species effect (Wilks' $\Lambda = 0.02$, $F(8, 288) = 199$, $p < 0.001$, multivariate $\eta^2 = 0.86$); univariate follow-up ANOVAs confirmed significant species differences on each of the four

flower dimensions ($p < 0.001$ for each), with petal length carrying the strongest effect ($\eta^2 = 0.94$)." Always pair MANOVA with both univariate follow-ups and a multivariate effect-size measure.

8.22.8 Practical Tips

- Pillai's trace is the recommended default — it is the most robust to covariance heterogeneity and unequal sample sizes.
- For a significant MANOVA, follow up with univariate ANOVAs on each outcome to identify which dimensions drive the effect; corrected p-values (Bonferroni or Holm) are appropriate at this stage.
- Test homogeneity of covariance with Box's M (`biotools::boxM`) before relying on Wilks; a significant Box's M motivates Pillai.
- For unequal sample sizes and multifactorial designs, use `car::Manova()` with explicit Type II or Type III sums of squares; base R's `summary.manova` defaults to sequential (Type I).
- Discriminant analysis follows naturally as a post-MANOVA exploration; the linear discriminants visualise where the multivariate group differences live.
- For few groups with many highly-correlated outcomes, MANOVA is more powerful than Bonferroni-corrected univariate tests; for many groups with few outcomes, separate ANOVAs may be more interpretable.

8.22.9 R Packages Used

Base R `manova()` and `summary.manova()`; `car::Manova()` for Type II / III sums of squares; `biotools::boxM()` for the homogeneity-of-covariance test; `MASS::lda()` for discriminant-analysis follow-up; `heplots` for visualising hypothesis-error decompositions.

8.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.22.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.23 Metric Multidimensional Scaling

8.23.1 Introduction

Metric multidimensional scaling — also known as classical MDS or principal coordinates analysis (PCoA) — takes a pairwise distance matrix and produces a low-dimensional Euclidean configuration whose pairwise distances approximate the input as closely as possible. It is the workhorse for visualising similarity or distance data when raw coordinates are unavailable: think reconstructing a map from inter-city distances, projecting microbial-community Bray-Curtis dissimilarities, or visualising a kernel-derived similarity matrix in a 2D plot.

8.23.2 Prerequisites

A working understanding of distance and similarity matrices, principal components analysis, and basic linear algebra (eigendecomposition).

8.23.3 Theory

Given an $n \times n$ distance matrix D, form the doubly-centred inner-product matrix

$$B = -\tfrac{1}{2}(I - J/n)D^{(2)}(I - J/n),$$

where $D^{(2)}$ has elements d_{ij}^2 and J is the all-ones matrix. Eigendecompose $B = U\Lambda U^\top$. The top k eigenvectors scaled by $\sqrt{\lambda}$ give a k-dimensional configuration whose Euclidean inter-point distances approximate the original distances. When D comes from Euclidean distances on raw data, classical MDS is equivalent to PCA on the centred data — the eigenvalues match the squared singular values of the centred data matrix.

For non-Euclidean distances, B may have negative eigenvalues; the configuration still uses only the top positive eigenvalues, with the discarded negative mass quantifying the embedding error.

8.23.4 Assumptions

A pairwise distance matrix that is at least approximately Euclidean-embeddable. Severe violations (negative eigenvalues dominating) suggest non-metric MDS as a more appropriate alternative.

8.23.5 R Implementation

```r
# Classical MDS on US city distances
data(UScitiesD)
mds <- cmdscale(UScitiesD, k = 2, eig = TRUE)
plot(mds$points, type = "n")
```

```r
text(mds$points, labels = labels(UScitiesD))

# Eigenvalues and variance explained
mds$eig[1:5]
cumsum(mds$eig[1:5] / sum(mds$eig[1:5]))
```

8.23.6 Output & Results

`cmdscale()` returns the low-dimensional configuration (`$points`), eigenvalues, and an indicator of how well the chosen k approximates the original distances. The plot of US-city MDS coordinates recovers the recognisable geographical structure with a near-perfect rotation and reflection of the actual map.

8.23.7 Interpretation

A reporting sentence: "Classical MDS on the 10 inter-city distances produced a two-dimensional configuration that captured 99 % of the eigen-mass; the resulting layout matches actual geography after a reflection, with east-west and north-south structure immediately visible." Always report the proportion of eigen-mass retained in the displayed dimensions; a poor approximation signals that more dimensions or non-metric MDS would help.

8.23.8 Practical Tips

- For Euclidean distances, classical MDS is equivalent to PCA on the centred data; if you have raw data, run PCA directly for cleaner output.
- For non-Euclidean distances (Bray-Curtis, Jaccard, network distances), check the eigenvalue structure — large negative eigenvalues indicate a non-Euclidean metric and motivate non-metric MDS instead.
- In ecology and microbiome research, this method is called principal coordinates analysis (PCoA); the algorithm is identical.
- Add Procrustes analysis (`vegan::procrustes`) when comparing MDS solutions across related datasets; it aligns configurations up to rotation, reflection, and scaling.
- For configurations beyond two dimensions, plot pairs of axes; `factoextra::fviz_mds()` provides a ggplot-style alternative.
- Pair the configuration with a stress measure or the proportion of variance retained to communicate fit quality alongside the visual.

8.23.9 R Packages Used

Base R `cmdscale()` for classical MDS; `vegan::wcmdscale()` for weighted classical MDS in ecology; `MASS::isoMDS()` and `vegan::metaMDS()` for non-metric alternatives; `factoextra::fviz_mds()` for ggplot-style visualisation; `smacof` for advanced MDS variants including weighted and constrained versions.

8.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.23.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.24 Non-Metric Multidimensional Scaling

8.24.1 Introduction

Non-metric multidimensional scaling (NMDS) preserves the *rank order* of pairwise dissimilarities rather than their absolute magnitudes. This robustness makes it the standard ordination method for ecological community data, where Bray-Curtis and Jaccard dissimilarities are not Euclidean and where distance values themselves carry less meaning than the ordering of similarities. Where classical MDS works on the squared distances directly, NMDS minimises a stress function that depends only on whether the configuration's distances respect the input ranking.

8.24.2 Prerequisites

A working understanding of metric MDS (classical / PCoA), rank-based statistics, and the distinction between metric and ordinal distance information.

8.24.3 Theory

NMDS finds a configuration Y in k dimensions whose pairwise Euclidean distances $d(Y_i, Y_j)$ are monotonically related to the input dissimilarities δ_{ij}. The optimisation minimises Kruskal stress:

$$\text{Stress} = \sqrt{\frac{\sum_{i<j}(d_{ij} - \hat{d}_{ij})^2}{\sum_{i<j} d_{ij}^2}},$$

where $\hat{d}_{ij}$ are *disparities* — values fitted via isotonic regression to be a non-decreasing function of δ_{ij}. The Shepard diagram plots δ_{ij} against d_{ij} with the isotonic fit overlaid; tight scatter around a smooth monotone curve indicates good ordination.

Stress interpretation (Kruskal): below 0.05 is excellent, 0.05–0.10 good, 0.10–0.20 fair, above 0.20 poor and the configuration should not be trusted.

8.24.4 Assumptions

Dissimilarity ranks are meaningful; sufficient observations for a stable configuration; the chosen number of dimensions (k) is large enough to accommodate the rank structure.

8.24.5 R Implementation

```r
library(MASS); library(vegan)

# Non-metric MDS on UScities
data(UScitiesD)
iso <- isoMDS(UScitiesD, k = 2)
iso$stress

# ecology example with vegan
data(varespec)
nmds <- metaMDS(varespec, distance = "bray", k = 2, trymax = 50)
nmds$stress
plot(nmds, type = "t")
```

8.24.6 Output & Results

`isoMDS()` and `metaMDS()` return a 2D configuration plus the stress value. `metaMDS()` also runs multiple random starts (`trymax`) and returns the best solution; this is essential because NMDS is non-convex and prone to local minima.

8.24.7 Interpretation

A reporting sentence: "NMDS of the vegetation community (Bray-Curtis dissimilarity, two dimensions, 50 random starts) achieved a stress of 0.18 — fair fit — and the configuration revealed two dominant environmental gradients corresponding to moisture and nutrient availability when site-level covariates were overlaid via `envfit()`." Always report the stress value and the number of starts; a single-start fit can land in a poor local minimum.

8.24.8 Practical Tips

- Always check stress; values above 0.2 indicate the configuration is unreliable and a higher-dimensional solution may be needed.
- Use multiple random starts (`trymax=50` or more in `metaMDS`); NMDS is non-convex and a single start often lands in a local minimum.

- NMDS is the standard for non-Euclidean ecological dissimilarities; metric MDS forces a Euclidean embedding that may not respect the underlying geometry.
- Procrustes-align configurations across related datasets (`vegan::procrustes`); NMDS solutions are defined up to rotation, reflection, and uniform scaling.
- Pair NMDS with `vegan::envfit()` or `ordisurf()` to overlay environmental covariates as arrows or contours, exposing the gradients the ordination has captured.
- Interpret relative positions only; absolute distances in the NMDS plot do not have a single fixed scale.

8.24.9 R Packages Used

`MASS::isoMDS()` for the classical Kruskal NMDS implementation; `vegan::metaMDS()` for the ecology-flavoured workflow with multiple starts and stress reporting; `smacof::smacofSym()` for advanced NMDS variants with weights and constraints; `ade4::nmds` for an alternative interface integrated with the dudi family of multivariate methods.

8.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.24.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.25 Multiple Correspondence Analysis

8.25.1 Introduction

Multiple correspondence analysis (MCA) is the categorical analogue of principal components analysis. Where PCA reduces a matrix of continuous variables to a small number of orthogonal axes capturing maximal variance, MCA reduces a set of categorical variables to axes capturing maximal chi-squared inertia. It is the natural exploratory tool for survey data, lifestyle patterns, ICD code panels, and any context in which several categorical variables jointly characterise units of observation.

8.25.2 Prerequisites

A working understanding of correspondence analysis on a two-way contingency table, indicator (disjunctive) coding of categorical variables, and the chi-squared distance interpretation.

8.25.3 Theory

MCA operates on the disjunctive-coded indicator matrix that represents each categorical variable by one binary column per category. Two equivalent computations exist:

- **Indicator-matrix CA**: apply standard correspondence analysis to the $n \times \sum K_q$ indicator matrix.
- **Burt-matrix CA**: apply CA to the Burt table, the cross-tabulation of all pairs of variables; this gives identical row and column structure with sometimes more intuitive geometry.

Each category becomes a point in the resulting low-dimensional space; each individual is also placed in the space (the "cloud of individuals"). Inertia per axis is typically low — much lower than typical PCA explained variance — because each indicator column contributes only one dimension of variation. Benzécri's or Greenacre's adjusted inertia formulas correct for this and give more meaningful variance-explained percentages.

8.25.4 Assumptions

Categorical variables (no strong zero cells), sufficient sample size relative to the number of categories, and a meaningful joint structure to extract.

8.25.5 R Implementation

```r
library(FactoMineR); library(factoextra)

data(hobbies)
mca <- MCA(hobbies[, 1:8], graph = FALSE)
fviz_mca_biplot(mca)
fviz_mca_var(mca)

summary(mca)
```

8.25.6 Output & Results

`MCA()` returns coordinates for individuals and category points, inertia per axis, contributions, and squared cosines (quality of representation). `fviz_mca_biplot()` displays both individuals and categories in the first two dimensions; `fviz_mca_var()` zooms in on the category cloud.

8.25.7 Interpretation

A reporting sentence: "MCA of 8 hobby variables (n = 8,403 respondents) extracted two interpretable dimensions: the first contrasts active vs sedentary hobbies (Benzécri-adjusted inertia 28 %), the second social vs solitary hobbies (adjusted inertia 19 %); supplementary projection of age and gender showed a clear age gradient on the first dimension and a weak gender effect on the second." Always report adjusted inertia rather than raw inertia; raw values systematically understate explained variation.

8.25.8 Practical Tips

- Use Benzécri's or Greenacre's adjusted inertia (`FactoMineR::MCA()` reports both) for variance-explained percentages; raw inertias are pessimistic and misleading.
- Supplementary categorical variables (`quali.sup`) project onto the axes without affecting their estimation; useful for adding demographic markers post-hoc.
- For mixed continuous-categorical data, switch to factor analysis of mixed data (FAMD); it generalises PCA and MCA jointly.
- Visualise the category cloud separately from the individual cloud when the latter is dense; `fviz_mca_var()` gives a cleaner view of category geometry.
- For ordinal categorical data, MCA loses the ordering — consider polychoric PCA or graded-response IRT models instead.
- Pair MCA with hierarchical clustering on the principal component scores (`HCPC()`) for a single workflow that combines exploration and partitioning.

8.25.9 R Packages Used

`FactoMineR` for `MCA()`, `FAMD()`, and `HCPC()`; `factoextra` for `fviz_mca_*` ggplot-style visualisation; `ca::mjca()` for an alternative Burt-table implementation; `ade4::dudi.acm()` for ecology-flavoured MCA.

8.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.25.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.26 Partitioning Around Medoids

8.26.1 Introduction

Partitioning around medoids (PAM) is the canonical k-medoids clustering algorithm. Like k-means, it partitions n observations into k clusters; unlike k-means, the cluster centres are actual observations (medoids) rather than coordinate means. This single change makes PAM robust to outliers (a single extreme value cannot drag a medoid away from the cluster bulk), enables clustering on arbitrary dissimilarity matrices including those derived from non-Euclidean distances or mixed numeric-categorical data, and produces interpretable centre observations that can be quoted in clinical or biological contexts ("the typical patient in cluster 2 is patient 47").

8.26.2 Prerequisites

A working understanding of k-means, distance and dissimilarity matrices, and the difference between mean-based and median-like cluster centres.

8.26.3 Theory

PAM alternates two steps:

1. **Build / swap**: for each cluster, choose the medoid that minimises the total dissimilarity to all other cluster members.
2. **Assign**: assign each observation to the cluster of its nearest medoid.

The objective is the sum of dissimilarities to the assigned medoid, summed over all observations. The algorithm converges to a local minimum; multiple random initialisations help find a better local optimum. Because PAM consumes a dissimilarity matrix, it works with Gower distance for mixed numeric-categorical data, Manhattan distance for outlier-robust clustering, correlation-based distances for gene-expression patterns, and any custom dissimilarity that the analyst can define.

8.26.4 Assumptions

A meaningful dissimilarity (or distance) on the input space. Approximately equal cluster sizes are preferable; PAM does not handle very imbalanced clusters as gracefully as model-based methods (Gaussian mixtures).

8.26.5 R Implementation

```
library(cluster)

# Mixed-type data with Gower distance
d_iris <- iris
```

```r
set.seed(2026)
d_iris$Category <- factor(sample(c("A", "B", "C"), 150, replace = TRUE))

g <- daisy(d_iris, metric = "gower")
pam_fit <- pam(g, k = 3)

pam_fit$medoids
table(pam_fit$clustering, iris$Species)

# Silhouette
summary(silhouette(pam_fit))
```

8.26.6 Output & Results

`pam()` returns medoid identifiers (actual case indices), cluster assignments, and within-cluster summaries. `silhouette()` returns per-observation silhouette widths and an average — a primary internal validation metric.

8.26.7 Interpretation

A reporting sentence: "Partitioning around medoids on Gower-distance dissimilarities produced three clusters that closely match the iris species (Cluster 1: setosa; Cluster 2: 92 % versicolor; Cluster 3: 86 % virginica) with average silhouette width 0.45 (range −0.10 to 0.78)." Always pair PAM with a silhouette analysis to verify that the chosen k produces well-separated clusters.

8.26.8 Practical Tips

- Choose PAM over k-means when outliers are likely or when the data are mixed-type or genuinely non-Euclidean; the medoid is robust where the mean is not.
- `cluster::pam()` accepts either a data matrix (computing Euclidean distance internally) or a pre-computed `dist` object via the `diss=TRUE` argument.
- For large datasets ($n > 1{,}000$), `cluster::clara()` runs PAM on subsamples and combines results, scaling to tens of thousands of observations.
- Use `fpc::pamk()` to automatically select k via the silhouette method; it returns the optimal k along with the fit.
- For Gower distances on mixed numeric and categorical data, `cluster::daisy(metric="gower")` is the standard input; categorical variables become indicator distances, numeric ones become Manhattan after range normalisation.
- For very large n, alternatives such as CLARANS, BIRCH, or mini-batch k-means scale better; the price is reduced robustness or interpretability.

8.26.9 R Packages Used

`cluster` for `pam()`, `clara()`, `daisy()`, and `silhouette()`; `fpc` for `pamk()` automatic k selection; `factoextra` for `fviz_cluster()` ggplot-style cluster visualisation; `klaR::kmodes()` for an analogous categorical-only k-modes algorithm.

8.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.26.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.27 PLS Discriminant Analysis

8.27.1 Introduction

PLS discriminant analysis (PLS-DA) extends partial least squares regression to classification by dummy-coding the class variable as a binary indicator matrix. It has become a standard tool in metabolomics, transcriptomics, and other omics disciplines because it handles the $p \gg n$ regime gracefully and, in its sparse variant, performs simultaneous classification and feature selection. The output — a small set of latent components plus the discriminant boundary in the projected space — is interpretable in a way that black-box classifiers (random forests, deep nets) often are not.

8.27.2 Prerequisites

A working understanding of PLS regression, classification metrics, and the high-dimensional ($p \gg n$) regime that motivates latent-component methods.

8.27.3 Theory

Encode the class variable as one-hot indicators $Y \in \{0,1\}^{n \times K}$ for K classes. PLS regression on (X, Y) produces latent components that maximise covariance with the indicator matrix; the predicted indicators are continuous, and a new sample is assigned to the class with the largest prediction. The number of components is selected by cross-validation (`mixOmics::perf()`).

Sparse PLS-DA (sPLS-DA) imposes an L1 penalty per component, retaining only `keepX` predictors per component. This produces a signature — a small list of variables driving the classification — that can be reported alongside the classifier itself.

8.27.4 Assumptions

Linear relationships between latent components and outcome probabilities, sample sizes large enough per class to estimate the components stably (rule of thumb: at least 5–10 observations per class), and meaningful covariance between predictors and the encoded indicators.

8.27.5 R Implementation

```
library(mixOmics)

data(srbct)
X <- srbct$gene
Y <- srbct$class

# PLS-DA
fit_plsda <- plsda(X, Y, ncomp = 3)
plotIndiv(fit_plsda, ind.names = FALSE, legend = TRUE)

# Sparse PLS-DA: select 50 variables per component
fit_splsda <- splsda(X, Y, ncomp = 3, keepX = c(50, 50, 50))
selectVar(fit_splsda, comp = 1)$name |> head()
```

8.27.6 Output & Results

`plsda()` returns latent component scores per sample, loadings for each predictor, and predicted class labels. `splsda()` adds variable-selection-per-component, with `selectVar()` extracting the chosen predictors. `plotIndiv()` displays samples in the latent component space, coloured by class.

8.27.7 Interpretation

A reporting sentence: "Sparse PLS-DA on the 2,308-gene SRBCT dataset selected 50 genes per component across three components, achieving 100 % balanced accuracy in 10-fold cross-validation; the resulting 150-gene signature substantially overlaps with the published transcriptomic markers of the four cancer subtypes." Always report both classification performance (accuracy, balanced accuracy, AUC) and the selected feature signature.

8.27.8 Practical Tips

- Use `mixOmics::perf()` with cross-validation to select `ncomp` and `keepX`; a fixed choice without tuning over-fits in high-dimensional omics.
- For two-class problems, logistic regression with elastic net regularisation often performs comparably and is more familiar; reserve sPLS-DA for multi-class or when the covariance structure is strongly multivariate.
- Sparse PLS-DA can produce unstable variable selections in the presence of highly correlated predictors; report selection stability via bootstrap resampling (`mixOmics::perf()` with `nrepeat=50`).
- The selected signature is the headline result for biological audiences; pair it with pathway-enrichment analysis to interpret functionally.
- Compare against alternative classifiers (random forest, elastic-net logistic, sPLS-DA) on identical CV folds; modern omics papers typically report multiple methods.
- For very imbalanced classes, use `near.zero.var=TRUE` and balanced cross-validation to prevent the majority class from dominating component construction.

8.27.9 R Packages Used

`mixOmics` for `plsda()`, `splsda()`, `perf()`, `selectVar()`, and the broader multi-block extensions; `caret` for unified cross-validated benchmarking against alternative classifiers; `ropls` for an alternative PLS-DA implementation popular in metabolomics; `glmnet` for elastic-net logistic regression as a complementary high-dimensional classifier.

8.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.27.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.28 Partial Least Squares Regression

8.28.1 Introduction

Partial least squares (PLS) regression is a latent-component method designed for the regime where predictors are numerous, correlated, and possibly outnumber observations — exactly the situation in which ordinary least squares fails. It extracts components from the predictor matrix that are simultaneously high-variance and strongly correlated with the outcome, then uses those components as the regression basis. Chemometrics and metabolomics use PLS as their default regression workhorse, and in bioinformatics it competes with elastic net and principal-components regression for high-dimensional prediction tasks.

8.28.2 Prerequisites

A working understanding of principal-components regression, multiple linear regression, cross-validation, and the trade-off between variance explanation and prediction.

8.28.3 Theory

PCA finds orthogonal directions in X that maximise variance of X. PLS finds directions that maximise *covariance* with Y — the latent components are simultaneously high-variance in X and predictive of Y. With a components, the model is

$$\hat{Y} = T_a Q_a^\top, \qquad T_a = X W_a,$$

where W_a are the PLS weights and T_a the resulting latent scores. The number of components is chosen by cross-validation, typically minimising root-mean-square error of prediction (RMSEP). PLS regression coefficients in the original predictor space are recovered as $\hat{\beta}_{\text{PLS}} = W_a (P_a^\top W_a)^{-1} Q_a^\top$.

8.28.4 Assumptions

Linear relationships between latent components and the outcome, independent observations, and (for inference) homoscedastic Normal residuals. PLS itself is fairly assumption-light for prediction; cross-validated RMSEP is the primary quality criterion.

8.28.5 R Implementation

```
library(pls)

data(mtcars)
```

```r
fit <- plsr(mpg ~ ., data = mtcars, validation = "CV", ncomp = 5)
summary(fit)
validationplot(fit, val.type = "RMSEP")

# Use the best number of components
n_comp <- 2
pred <- predict(fit, ncomp = n_comp)
```

8.28.6 Output & Results

`summary(fit)` reports cumulative explained variance in X and in Y for each component count, plus cross-validated RMSEP. `validationplot()` shows the RMSEP curve against component count, with a clear minimum at the optimal number of components.

8.28.7 Interpretation

A reporting sentence: "PLS regression with two latent components yielded a cross-validated RMSEP of 2.3 mpg on the 32-row `mtcars` dataset; the two components captured 85 % of variance in the predictors and 92 % of variance in the outcome, after which additional components gave diminishing returns." Report both X- and Y-variance explained; PLS optimises the latter, but readers familiar with PCA may expect the former.

8.28.8 Practical Tips

- Always use cross-validation to choose the number of components; in-sample fit is monotonically better with more components and selects too many.
- For classification, PLS-DA (`mixOmics::plsda()`) dummy-codes the outcome and applies PLS regression; it remains popular in metabolomics.
- Sparse PLS (`mixOmics::spls()`) adds variable selection at each component, useful when interpretability of individual predictors matters.
- For truly wide data ($p \gg n$), PLS and sparse PLS are core tools alongside ridge, lasso, and elastic net; cross-validate all four for benchmarking.
- Interpret latent components cautiously; they are linear combinations of all (or many) predictors and rarely have direct biological interpretations.
- For non-linear extensions, kernel PLS and PLS with spline-transformed predictors generalise to non-linear response surfaces.

8.28.9 R Packages Used

`pls` for `plsr()`, `mvr()`, and the standard cross-validation framework; `mixOmics` for PLS-DA, sparse PLS, and multi-block extensions popular in omics; `caret::tra`

`in()` for unified cross-validated comparison against ridge, lasso, and elastic-net regression; `plsmod` for tidymodels-flavoured PLS interfaces.

8.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.28.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.29 Principal Component Analysis

8.29.1 Introduction

Principal component analysis (PCA) takes a data matrix with many correlated variables and re-expresses it in a smaller number of uncorrelated components that capture most of the variance. It is used for exploratory visualisation, for compression, as a preprocessing step for regression and classification, and as a quality-control tool in high-dimensional experiments such as RNA-seq. It is not a model for the data; it is a change of basis.

8.29.2 Prerequisites

The reader should be familiar with vectors, variance, covariance, and correlation. Some understanding of eigenvalues and eigenvectors helps but is not essential.

8.29.3 Theory

Given a data matrix $\mathbf{X}$ with n rows (observations) and p columns (variables), PCA finds orthogonal directions in the p-dimensional space of variables that successively maximise the projected variance. The first principal component (PC1) is the direction of greatest variance; PC2 is the direction of greatest variance orthogonal to PC1; and so on up to $\min(n-1, p)$.

Algebraically, if $\mathbf{X}$ is centred (each column has mean zero), PCA is the eigendecomposition of the covariance matrix $\mathbf{S} = \frac{1}{n-1}\mathbf{X}^\top\mathbf{X}$:

$$\mathbf{S} = \mathbf{V}\Lambda\mathbf{V}^\top,$$

where the columns of $\mathbf{V}$ are the principal-component directions (loadings), and Λ is the diagonal matrix of variances along each direction (eigenvalues). The scores – coordinates of the observations in the new basis – are $\mathbf{Z} = \mathbf{XV}$.

A practical issue: if the variables are on different scales, the variance-maximisation criterion is dominated by whichever variable happens to have the biggest numeric range. For heterogeneous data, PCA should be performed on the correlation matrix instead – equivalent to standardising each variable to unit variance before the decomposition.

8.29.4 Assumptions

PCA has no formal probabilistic assumptions, but its interpretation rests on:

1. The relationships of interest are linear in the variables.
2. Variance is a reasonable proxy for "importance".
3. The first few components contain most of the useful signal.

The first is violated when relationships are non-linear; alternative methods (kernel PCA, UMAP) may be needed. The second is violated when small-variance components encode the scientific signal (for example, subtle gene regulators in a bulk RNA-seq study).

8.29.5 R Implementation

```
library(FactoMineR)
library(factoextra)

data(iris)
X <- iris[, 1:4]

pca <- PCA(X, scale.unit = TRUE, graph = FALSE)

fviz_eig(pca, addlabels = TRUE)
fviz_pca_biplot(pca,
                habillage = iris$Species,
                palette = "Set2",
                addEllipses = TRUE,
                label = "var")

summary(pca)
```

`PCA()` from `FactoMineR` handles centring and scaling by default. `fviz_eig()` draws the scree plot; `fviz_pca_biplot()` draws the biplot with observations coloured by species and variable-loading arrows overlaid. `summary()` produces a textual summary including variance explained, variable loadings, and contribution percentages.

8.29.6 Output & Results

For the `iris` data, the first component captures approximately 73% of the variance, the second another 23%; together the first two components retain 96% of the information. The biplot shows that the three species separate cleanly along PC1, which is heavily loaded on petal length, petal width, and sepal length.

8.29.7 Interpretation

In a manuscript, a PCA is usually reported in two pieces: the proportion of variance explained by the first k components, and the biological or physical interpretation of each component's loadings. "PC1 (73% of variance) is a size component, loading positively on all length measurements; PC2 (23%) contrasts sepal width against the other measurements and separates *I. setosa* from the other two species."

8.29.8 Practical Tips

- Always standardise (`scale.unit=TRUE`) unless all variables share the same units and you have a reason to preserve absolute-scale differences.
- Inspect the scree plot and keep components up to an elbow, or enough for 80-90% cumulative variance, or until an interpretable pattern emerges. There is no universal rule.
- The sign of each component is arbitrary (eigenvectors are unique up to sign), so "PC1 goes up" and "PC1 goes down" are equivalent representations; do not over-interpret the direction.
- For high-dimensional sparse data (e.g., scRNA-seq), use randomised SVD (`irlba`, `rsvd`) or the PCA implementations in `Seurat` or `scran` rather than dense eigendecomposition.
- PCA is sensitive to outliers; robust variants (ROBPCA, projection pursuit PCA) are available in the `rrcov` package.

8.29.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.29.10 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.30 Quadratic Discriminant Analysis

8.30.1 Introduction

Quadratic discriminant analysis (QDA) extends linear discriminant analysis (LDA) by allowing each class to have its own covariance matrix instead of pooling across classes. The relaxation transforms the decision boundary from a hyperplane (LDA) into a conic section (ellipse, parabola, or hyperbola) and gives the classifier flexibility to handle classes whose spread differs in shape or orientation. The cost is parameter count: K separate covariance matrices instead of one, which means more training data is required to estimate them precisely.

8.30.2 Prerequisites

A working understanding of linear discriminant analysis, multivariate Normal distributions, and the role of class covariance matrices in shaping decision boundaries.

8.30.3 Theory

The QDA discriminant for class k is

$$\delta_k(x) = -\tfrac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k) - \tfrac{1}{2}\log|\Sigma_k| + \log \pi_k,$$

where μ_k, Σ_k, π_k are the class mean, class covariance, and class prior probability respectively. A test observation is assigned to the class with the largest $\delta_k(x)$. The quadratic term in x is what produces non-linear decision boundaries; pooling covariances across classes (as LDA does) eliminates the quadratic term and recovers linearity.

8.30.4 Assumptions

Multivariate Normal within each class, with possibly different covariance matrices across classes; independent observations; sufficient observations per class to estimate Σ_k stably (rule of thumb: at least 5–10 observations per dimension per class).

8.30.5 R Implementation

```
library(MASS)

set.seed(2026)
train_idx <- sample(nrow(iris), 100)
train <- iris[train_idx, ]; test <- iris[-train_idx, ]
```

```
fit_qda <- qda(Species ~ ., data = train)
pred <- predict(fit_qda, test)
table(pred$class, test$Species)
```

8.30.6 Output & Results

`qda()` returns class priors, class means, and class-specific group counts. `predict()` provides class labels and posterior probabilities. The confusion matrix (`table(pred$class,test$Species)`) summarises classification accuracy on the held-out fold.

8.30.7 Interpretation

A reporting sentence: "QDA classified the iris test set with 94 % accuracy (47/50 correct); LDA reached 98 % on the same split, suggesting the equal-covariance assumption was reasonable here and the additional flexibility of QDA is not warranted." Compare QDA against LDA whenever class covariances are not obviously different; the simpler model often wins.

8.30.8 Practical Tips

- Prefer QDA when within-class covariance structures clearly differ in scale or orientation; if pooled covariance is a good approximation, LDA generalises better.
- QDA requires more training data than LDA because of the per-class covariance estimation; in high dimensions or with small classes, LDA or regularised discriminant analysis (RDA) is more reliable.
- RDA (`klaR::rda`) interpolates between LDA and QDA via shrinkage parameters; cross-validate the shrinkage to find the data's preferred point on the spectrum.
- QDA decision boundaries are conic — ellipses, parabolas, or hyperbolas; visualise on a 2D projection (`klaR::partimat`) to see the curvature.
- For high-dimensional or non-Gaussian data, compare against flexible classifiers (random forests, SVMs with RBF kernels); QDA's quadratic boundary may not match the true structure.
- Diagnose the equal-covariance assumption with Box's M test (`biotools::boxM`); a significant test motivates QDA over LDA, but the test is itself sensitive to non-Normality.

8.30.9 R Packages Used

`MASS` for `qda()` and `lda()`; `klaR` for `rda()` (regularised discriminant analysis) and `partimat()` for partition diagnostics; `caret` for cross-validated comparison across discriminant variants; `biotools::boxM()` for Box's M test of homogeneity of covariance.

8.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.30.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.31 Structural Equation Models

8.31.1 Introduction

Structural equation modelling (SEM) is the unified framework that integrates measurement (how observed items reflect underlying latent constructs) and structure (how latent constructs relate to each other) into a single estimation problem. It generalises confirmatory factor analysis (measurement only), path analysis (regressions among observed variables only), and multivariate regression. Modern psychometrics, organisational research, and increasingly clinical-outcomes work depend on SEM as the primary analytical tool because it isolates relationships among true constructs from those distorted by measurement error.

8.31.2 Prerequisites

A working understanding of confirmatory factor analysis, multiple regression, latent-variable thinking, and basic familiarity with `lavaan` syntax.

8.31.3 Theory

SEM estimates measurement and structural parameters jointly by maximum likelihood (or robust ML, weighted least squares for ordinal indicators). Components in `lavaan` syntax:

- **Measurement model** (`=~`): latent factor defined by its observed indicators.
- **Structural regressions** (`~`): latent-on-latent or latent-on-observed regressions.
- **Variances / covariances** (`~~`): residual correlations or factor covariances.
- **Defined parameters** (`:=`): derived quantities such as indirect effects.

Standard fit indices (CFI, TLI, RMSEA, SRMR) parallel those of CFA. Indirect effects and confidence intervals are computed at the level of MCMC or bootstrap draws.

8.31.4 Assumptions

Correct specification of both the measurement and structural components, multivariate Normal continuous indicators (or robust estimators for non-Normal data), and sufficient sample size — at least 200 observations as a rough rule, more for complex models.

8.31.5 R Implementation

```
library(lavaan); library(semPlot)

model <- '
  # Measurement
  ind60 =~ x1 + x2 + x3
  dem60 =~ y1 + y2 + y3 + y4
  dem65 =~ y5 + y6 + y7 + y8

  # Structural
  dem60 ~ ind60
  dem65 ~ ind60 + dem60

  # Residual correlations
  y1 ~~ y5
  y2 ~~ y4 + y6
  y3 ~~ y7
  y4 ~~ y8
  y6 ~~ y8
'
fit <- sem(model, data = PoliticalDemocracy)
summary(fit, standardized = TRUE, fit.measures = TRUE)

semPaths(fit, "std", layout = "tree2")
```

8.31.6 Output & Results

`summary(fit,standardized=TRUE)` reports unstandardised and standardised coefficients, factor loadings, residual covariances, and the full set of fit indices. `semPaths()` renders a path diagram with arrow widths proportional to standardised coefficients.

8.31.7 Interpretation

A reporting sentence: "The structural equation model fit the Political Democracy data well (CFI 0.99, RMSEA 0.04 [90 % CI 0.02 to 0.06], SRMR 0.05); industrialisation in 1960 predicted both 1960 and 1965 democracy ($\beta = 0.45$ and 0.39 respectively), and 1960 democracy strongly predicted 1965 democracy ($\beta = 0.84$, $p < 0.001$)." Always report standardised coefficients alongside fit indices.

8.31.8 Practical Tips

- Check the measurement model in isolation (run a CFA) before adding structural paths; weak factors produce uninterpretable structural relationships.
- Use `modificationIndices()` cautiously; data-driven modifications blur the confirmatory status of the analysis and inflate Type I error.
- For mediation analysis, define indirect effects with `:=` and request bootstrap CIs (`sem(...,se="bootstrap")`); the indirect effect is a product of coefficients with non-Normal sampling distribution.
- For ordinal indicators, switch to weighted least squares (`estimator="WLSMV"`); ordinary ML on Likert items violates Normality assumptions.
- Very complex models need very large samples; if $n < 5$ per estimated parameter, results are unstable and a simpler path model may be more robust.
- For cross-cultural or multi-group comparisons, fit measurement-invariance hierarchies (configural, metric, scalar) before comparing latent means across groups.

8.31.9 R Packages Used

`lavaan` for `sem()`, `cfa()`, `growth()`, and `lavInspect()`; `semPlot` for path diagrams; `semTools` for measurement-invariance tests, reliability coefficients, and modification heuristics; `blavaan` for Bayesian SEM that handles small samples and complex priors.

8.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.31.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA

- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.32 UMAP and t-SNE: Overview

8.32.1 Introduction

UMAP (uniform manifold approximation and projection) and t-SNE (t-distributed stochastic neighbour embedding) are the dominant non-linear dimensionality-reduction techniques for visualising very high-dimensional data — single-cell RNA sequencing, large image embeddings, language-model representations. They produce striking 2D or 3D scatter plots in which similar observations cluster together, and they are now near-default visualisations in many computational-biology and machine-learning pipelines. They also have well-documented pitfalls, and learning what they do *not* show is as important as learning to apply them.

8.32.2 Prerequisites

A working understanding of principal components analysis, distance metrics, and the trade-off between preserving local and global structure in a low-dimensional embedding.

8.32.3 Theory

Both methods optimise a 2D or 3D embedding such that pairwise similarities computed in the original high-dimensional space are preserved as well as possible.

- **t-SNE** computes Gaussian-weighted neighbour probabilities in the high-dimensional space, t-distributed neighbour probabilities in the low-dimensional space, and minimises the KL divergence between them. The Student-t distribution in low dimensions prevents the "crowding problem" (compressing many high-dimensional neighbours into a small low-dimensional area).
- **UMAP** constructs a fuzzy topological representation of the high-dimensional data using local distance scaling, then optimises a similar fuzzy structure in low dimensions via a cross-entropy loss. UMAP is faster, scales better to large n, and is often considered to preserve global structure better than t-SNE.

Both algorithms are stochastic; setting the random seed is required for reproducible figures.

8.32.4 Assumptions

A similarity-preserving 2D / 3D embedding is informative for the inferential question. Both algorithms assume meaningful local structure exists in the data; they do not assume Gaussianity, linearity, or any specific distance metric (UMAP's metric is configurable).

8.32.5 R Implementation

```
library(umap); library(Rtsne)

set.seed(2026)
X <- as.matrix(iris[, 1:4])

# UMAP
u <- umap(X)
plot(u$layout, col = iris$Species, pch = 16,
     main = "UMAP of iris")

# t-SNE
t <- Rtsne(X, perplexity = 15, check_duplicates = FALSE)
plot(t$Y, col = iris$Species, pch = 16,
     main = "t-SNE of iris")
```

8.32.6 Output & Results

Both methods produce 2D coordinates per observation. UMAP and t-SNE both separate the three iris species clearly; differences in the layout (relative cluster positions, cluster sizes) reflect algorithm-specific choices rather than data-driven structure.

8.32.7 Interpretation

A reporting sentence: "UMAP and t-SNE embeddings of the four iris measurements both separate the three species, with setosa well-isolated and versicolor / virginica close but distinguishable; UMAP preserves the inter-cluster gap more faithfully than t-SNE, consistent with its better global-structure properties." Always describe the algorithm and key hyperparameters in the figure caption; readers should not have to guess.

8.32.8 Practical Tips

- Treat distances in the embedding as relative neighbourhoods only; absolute distances and cluster *sizes* are not meaningful, despite how the visualisation looks.

- t-SNE's `perplexity` parameter (typically 5–50) sets the effective neighbourhood scale; vary it across at least three values to confirm the structure is robust.
- UMAP's `n_neighbors` and `min_dist` control the same trade-offs; defaults work for most data but should be tuned for very small or very large n.
- For very high-dimensional input, run PCA first to reduce noise to the top 50 components, then UMAP/t-SNE on the PCA scores; this is the standard pipeline in single-cell analysis.
- Cluster sizes in the embedding are meaningless; do not interpret a "small cluster" or "large cluster" as biologically real.
- Reproducibility requires a fixed seed *and* fixed software versions; UMAP results in particular can shift between releases.

8.32.9 R Packages Used

`umap` for the canonical UMAP implementation; `uwot` for a faster UMAP variant with more options; `Rtsne` for the standard t-SNE; `M3C` and `embed` for tidymodels-flavoured wrappers; `dimRed` for a unified interface across UMAP, t-SNE, Isomap, and other non-linear embeddings.

8.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

8.32.11 See also — labs in this chapter

- PCA, factor analysis, CCA, LDA
- Clustering: k-means, hierarchical, model-based
- UMAP and t-SNE

8.33 Week 1, Session 4 — Clustering: k-means, hierarchical, model-based

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

8.33.1 Learning objectives

- Fit k-means and hierarchical clusterings, and choose k by elbow and silhouette.

- Fit a model-based clustering with `mclust` and select the number of components by BIC.
- Explain why "number of clusters" is often not the right question.

8.33.2 Prerequisites

PCA; distance and linkage.

8.33.3 Background

Clustering is a search for structure in unlabelled data, and unlike classification it has no gold standard. Any algorithm will return a partition if asked to, and the partitions it returns depend on assumptions about what a cluster is. K-means assumes roughly spherical, equal-variance clusters and needs k in advance. Hierarchical clustering produces a nested sequence of partitions at every k, which you cut at a level you choose. Model-based clustering fits a Gaussian mixture and selects k by information criterion, which at least makes the assumption explicit.

In biomedicine, the honest version of "how many clusters?" is usually "is there more than one?" A good clustering lab focuses on stability and interpretability of the partition rather than on the number returned.

When you report a clustering, report the number of clusters, the criterion used to select it, a visualisation in a 2-D projection (PCA or UMAP), and the size and key features of each cluster. Do not report p-values for cluster labels against the data that produced the labels; that circularity is a classic error.

8.33.4 Setup

```
library(tidyverse)
library(mclust)
library(cluster)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

8.33.5 1. Goal

Cluster a simulated dataset with three known groups using three algorithms, and see which recovers the structure.

8.33.6 2. Approach

Three Gaussian blobs in 5 dimensions with different sizes.

```
make_blob <- function(n, mu, sd = 1) matrix(rnorm(n * 5, 0, sd), n, 5) +
  rep(mu, each = n)
```

```
X <- rbind(
  make_blob(60, c(0, 0, 0, 0, 0)),
  make_blob(40, c(3, 0, 0, 0, 0)),
  make_blob(30, c(0, 3, 0, 0, 0))
)
true_lbl <- rep(c("A", "B", "C"), c(60, 40, 30))

pc <- prcomp(X)$x[, 1:2]
tibble(PC1 = pc[, 1], PC2 = pc[, 2], group = true_lbl) |>
  ggplot(aes(PC1, PC2, colour = group)) + geom_point(alpha = 0.7)
```

8.33.7 3. Execution

K-means with k sweep; silhouette.

```
wss <- sapply(2:8, function(k) kmeans(X, centers = k, nstart =
 ↪  10)$tot.withinss)
sil <- sapply(2:8, function(k) {
  km <- kmeans(X, centers = k, nstart = 10)
  mean(silhouette(km$cluster, dist(X))[, 3])
})
tibble(k = 2:8, wss = wss, sil = sil) |>
  pivot_longer(-k) |>
  ggplot(aes(k, value)) + geom_line() + geom_point() +
  facet_wrap(~ name, scales = "free_y")
```

Hierarchical with Ward linkage.

```
hc <- hclust(dist(X), method = "ward.D2")
plot(hc, labels = FALSE, main = "Ward hierarchical clustering")
```

8.33.8 4. Check

Model-based clustering by BIC.

```
mc <- Mclust(X, G = 1:6)
summary(mc)
```

8.33.9 5. Report

> On three simulated Gaussian blobs (n = 130 in $\mathbb{R}^5$), model-based clustering selected G = `mc$G` by BIC; k-means with k = `which.max(sil)+1` maximised mean silhouette (`round(max(sil),2)`). Hierarchical clustering with Ward linkage produced a dendrogram consistent with three primary branches.

When the generating model is actually Gaussian, `mclust` tends to win. On real data, no single method is best; the test is whether the partition is stable under resampling and interpretable in terms of the features.

8.33.10 Common pitfalls

- Treating k-means output as ground truth; it is an optimisation answer given k.
- Running hierarchical clustering on a very large dataset without noting the $O(n^2)$ memory cost.
- Testing post-hoc differences between clusters with the same data that produced them.

8.33.11 Further reading

- Fraley C, Raftery AE (2002), *Model-based clustering, discriminant analysis, and density estimation.*
- Hastie, Tibshirani, Friedman, *ESL*, §14.3.

8.33.12 Session info

```
sessionInfo()
```

8.33.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

8.34 Week 1, Session 3 — PCA, factor analysis, CCA, LDA

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

8.34.1 Learning objectives

- Compute a PCA, read a biplot, and explain loadings and scores without resorting to magical reasoning about "directions of variance".
- Distinguish PCA, factor analysis, canonical correlation analysis, and linear discriminant analysis by the question each answers.
- Fit an LDA classifier and interpret the discriminant coefficients.

8.34.2 Prerequisites

Matrix algebra (eigendecomposition); basic multivariate intuition.

8.34.3 Background

Dimension reduction is often described as "finding structure", but each of these methods has a concrete question behind it. PCA asks which directions in feature space have the largest variance; its axes are eigenvectors of the covariance matrix. Factor analysis asks which latent factors, plus unique variance per feature, reproduce the observed covariance; it is PCA's statistical cousin with a proper error model. Canonical correlation analysis asks which linear combinations of two feature sets are maximally correlated. Linear discriminant analysis asks which direction separates known classes while collapsing within-class variance.

They share machinery — all four involve eigenvalues of something — but they are not interchangeable. Using PCA where LDA was called for will waste the label information; using LDA where PCA was called for will produce a single, class-dependent axis and nothing else.

When you report a PCA, report the variance explained by each of the first few components and a biplot with named observations and loadings. When you report an LDA, report the classification accuracy on held-out data and the direction of each discriminant, not just the confusion matrix.

8.34.4 Setup

```
library(tidyverse)
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

8.34.5 1. Goal

Use `iris` to illustrate PCA and LDA on the same data, compare their projections, and classify species with LDA.

8.34.6 2. Approach

Four morphometric variables, three species. PCA ignores the label; LDA uses it.

```
iris |>
  pivot_longer(-Species) |>
  ggplot(aes(value, fill = Species)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~ name, scales = "free") +
  labs(x = NULL, y = "density")
```

8.34.7 3. Execution

PCA on the four numeric features, then LDA.

```
pca <- prcomp(iris[, 1:4], scale. = TRUE)
summary(pca)$importance[, 1:3]

lda_fit <- lda(Species ~ ., data = iris)
lda_fit$scaling
```

```
scores_pca <- as_tibble(pca$x[, 1:2]) |> mutate(Species = iris$Species)
scores_lda <- as_tibble(predict(lda_fit)$x) |> mutate(Species =
↪  iris$Species)
p1 <- ggplot(scores_pca, aes(PC1, PC2, colour = Species)) +
  geom_point(alpha = 0.7) + labs(title = "PCA (unsupervised)")
p2 <- ggplot(scores_lda, aes(LD1, LD2, colour = Species)) +
  geom_point(alpha = 0.7) + labs(title = "LDA (uses labels)")
gridExtra::grid.arrange(p1, p2, ncol = 2)
```

8.34.8 4. Check

Hold-out accuracy for LDA (simple 70/30 split).

```
idx <- sample(nrow(iris), 0.7 * nrow(iris))
fit2 <- lda(Species ~ ., data = iris[idx, ])
pred <- predict(fit2, iris[-idx, ])$class
mean(pred == iris$Species[-idx])
```

8.34.9 5. Report

> On the *iris* data, the first two principal components explained `round(sum(summary(pca)$importance[2,1:2])*100,1)`% of the total variance. A linear discriminant classifier achieved hold-out accuracy of `round(mean(pred==iris$Species[-idx])*100,1)`% on a 70/30 split.

In this low-dimensional problem, PCA already separates the species visually because the morphometric variables are correlated with species. In realistic biomedical data, that is rarely true: the supervised projection (LDA, or a classifier's decision function) is usually sharper.

8.34.10 Common pitfalls

- Reporting "PC1 is age" because age happens to correlate with it.
- Running LDA when classes are hugely imbalanced; use QDA or a proper classifier instead.
- Comparing variance-explained across datasets as if it were a quality score.

8.34.11 Further reading

- Jolliffe IT (2002), *Principal Component Analysis*, 2nd ed.
- Mardia, Kent, Bibby, *Multivariate Analysis*, chapters on LDA/CCA.

8.34.12 Session info

```
sessionInfo()
```

8.34.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

8.35 Week 1, Session 5 — UMAP and t-SNE

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

8.35.1 Learning objectives

- Compute UMAP and t-SNE embeddings and describe what each optimisation is doing.
- Interpret global vs local structure in a UMAP/t-SNE plot and avoid over-reading cluster distances.
- Tune `n_neighbors` / `perplexity` and see the visual effect.

8.35.2 Prerequisites

PCA; basic notion of neighbour graphs.

8.35.3 Background

UMAP and t-SNE are nonlinear embedding methods widely used in single-cell and imaging biomedicine. Both build a neighbour graph in high dimensions and then try to match a lower-dimensional graph with similar neighbour structure. They are not clustering algorithms, and distances in the embedding are not Euclidean distances in the original space. The prominent visual clusters in a UMAP often exist in the data, but the distance between clusters is close to meaningless and the density inside a cluster is not comparable across clusters.

t-SNE emphasises local structure; UMAP, with default settings, is a little more faithful to global structure but can still distort. Both have knobs — `perplexity` for t-SNE, `n_neighbors` and `min_dist` for UMAP — that change the resulting picture substantially. A reviewer who sees only one embedding has no way to tell if the picture is a stable feature of the data or an artefact of the settings.

Treat UMAP/t-SNE as a visualisation of neighbourhood structure, not as a geometric truth. Do not run statistical tests on the coordinates. Do not claim a trajectory unless you used a trajectory method.

8.35.4 Setup

```
library(tidyverse)
library(uwot)
library(Rtsne)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

8.35.5 1. Goal

Embed a simulated high-dimensional dataset with three blobs into two dimensions using UMAP and t-SNE, and compare against PCA.

8.35.6 2. Approach

```
make_blob <- function(n, mu, sd = 1, d = 20) matrix(rnorm(n * d, 0, sd),
 ↪  n, d) +
  rep(mu, each = n)
X <- rbind(
  make_blob(100, c(0, 0, rep(0, 18))),
  make_blob(100, c(4, 0, rep(0, 18))),
  make_blob(100, c(0, 4, rep(0, 18)))
)
lbl <- rep(c("A", "B", "C"), each = 100)
```

8.35.7 3. Execution

```
emb_pca  <- prcomp(X)$x[, 1:2]
emb_umap <- umap(X, n_neighbors = 15, min_dist = 0.1)
emb_tsne <- Rtsne(X, perplexity = 30, check_duplicates = FALSE)$Y

df <- bind_rows(
  tibble(method = "PCA",  x = emb_pca[, 1],  y = emb_pca[, 2],  lbl =
   ↪  lbl),
  tibble(method = "UMAP", x = emb_umap[, 1], y = emb_umap[, 2], lbl =
   ↪  lbl),
  tibble(method = "tSNE", x = emb_tsne[, 1], y = emb_tsne[, 2], lbl = lbl)
)
ggplot(df, aes(x, y, colour = lbl)) +
  geom_point(alpha = 0.7, size = 0.8) +
```

```
  facet_wrap(~ method, scales = "free") +
  labs(x = NULL, y = NULL)
```

8.35.8 4. Check

Sensitivity to `n_neighbors`.

```
do_umap <- function(k) {
  e <- umap(X, n_neighbors = k, min_dist = 0.1)
  tibble(x = e[, 1], y = e[, 2], lbl = lbl, k = paste0("nn=", k))
}
bind_rows(do_umap(5), do_umap(15), do_umap(50)) |>
  ggplot(aes(x, y, colour = lbl)) + geom_point(alpha = 0.7, size = 0.8) +
  facet_wrap(~ k, scales = "free")
```

8.35.9 5. Report

> Both UMAP and t-SNE recovered the three simulated blobs as distinct visual clusters; PCA separated them more weakly because the signal lay along two of 20 dimensions. UMAP embeddings with `n_neighbors` $\in \{5, 15, 50\}$ all identified the same three groups but differed substantially in their relative layout.

The message is not which method is best, but that the embedding is best used as a visual companion, with another method — clustering, classification, or differential analysis — doing the statistical work.

8.35.10 Common pitfalls

- Reading distances between clusters off a UMAP as if they were biological distances.
- Running one embedding with defaults and treating the result as the data.
- Using UMAP coordinates as features in a downstream supervised model.

8.35.11 Further reading

- McInnes L, Healy J, Melville J (2018), *UMAP: Uniform manifold approximation and projection*.
- Kobak D, Berens P (2019), *The art of using t-SNE for single-cell transcriptomics*.

8.35.12 Session info

```
sessionInfo()
```

8.35.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 9

Time-Series Analysis

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Stationarity, ARIMA, exponential smoothing, state-space models, spectral analysis, and forecasting evaluation. Examples use biomedical signals — heart rate, glucose, EEG — rather than macroeconomic data.

This chapter contains **30 method pages** and **1 lab**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

9.1 Method pages

Method	Source slug
ARIMA Models	`arima-models`
Augmented Dickey-Fuller Test	`adf-test`
Bayesian Changepoint Detection	`bcp-bayesian-changepoint`
Changepoint Detection	`changepoint-detection`
Diebold-Mariano Test	`diebold-mariano`
Differencing a Time Series	`differencing`
ETS Models	`ets-models`
Exponential Smoothing	`exponential-smoothing`
Forecast Accuracy Metrics	`forecast-accuracy-metrics`
GARCH Models	`garch-volatility`
Granger Causality	`granger-causality`
Holt-Winters Method	`holt-winters`
Interpreting ACF and PACF	`acf-pacf-interpretation`
KPSS Test	`kpss-test`
Moving Averages	`moving-averages`
Phillips-Perron Test	`phillips-perron`

Method	Source slug
Rolling-Origin Cross-Validation	`rolling-origin-cv`
Seasonal ARIMA (SARIMA)	`seasonal-arima`
Seasonal Decomposition (STL)	`seasonal-decomposition-stl`
Spectral Analysis	`spectral-analysis`
State-Space Models	`state-space-models`
Stationarity Tests	`stationarity-tests`
The Kalman Filter	`kalman-filter`
The Ljung-Box Test	`ljung-box`
Time Series: Introduction	`time-series-introduction`
VECM and Cointegration	`vecm-cointegration`
Vector Autoregression (VAR)	`multivariate-var`
Wavelet Analysis	`wavelet-analysis`
White Noise Tests	`white-noise-tests`
X-13ARIMA-SEATS Decomposition	`x13-decomposition`

9.2 Labs

Lab
Time series basics

9.3 Interpreting ACF and PACF

9.3.1 Introduction

The autocorrelation function (ACF) and partial autocorrelation function (PACF) are the foundational diagnostic tools for ARIMA model identification. The Box-Jenkins approach uses their characteristic patterns to guess plausible orders for AR and MA terms before fitting. Even with modern automated tools (`auto.arima`), reading the ACF and PACF remains an essential skill for sanity-checking automated choices, identifying seasonal structure, and diagnosing residuals.

9.3.2 Prerequisites

A working understanding of autoregressive (AR) and moving-average (MA) processes, the difference between correlation and partial correlation, and stationarity as a precondition for meaningful ACF interpretation.

9.3.3 Theory

The ACF ρ_k measures correlation between y_t and y_{t-k}. The PACF ϕ_{kk} measures correlation between y_t and y_{t-k} after removing the effects of shorter lags

$1, 2, \ldots, k-1$. Box-Jenkins identification rules:

Process	ACF	PACF
AR(p)	Tails off (decays slowly)	Cuts off at lag p
MA(q)	Cuts off at lag q	Tails off
ARMA(p, q)	Tails off	Tails off

The dashed lines in R's ACF plot show approximate 95 % confidence bands ($\pm 1.96/\sqrt{n}$) under a white-noise null; values outside the bands are statistically significant at the 5 % level.

For seasonal data, expect spikes at multiples of the seasonal period; multiplicative seasonal patterns produce decaying spikes at $s, 2s, 3s, \ldots$ in the ACF.

9.3.4 Assumptions

The series is stationary; if not, difference first until it becomes stationary, then read the ACF / PACF on the differenced series.

9.3.5 R Implementation

```r
library(forecast)

set.seed(2026)
ar_series <- arima.sim(list(ar = 0.7), n = 200)
ma_series <- arima.sim(list(ma = -0.5), n = 200)

par(mfrow = c(2, 2))
acf(ar_series, main = "ACF of AR(1)")
pacf(ar_series, main = "PACF of AR(1)")
acf(ma_series, main = "ACF of MA(1)")
pacf(ma_series, main = "PACF of MA(1)")
par(mfrow = c(1, 1))

ggtsdisplay(ar_series)
```

9.3.6 Output & Results

The four-panel display shows characteristic Box-Jenkins patterns: AR(1) ACF tails off geometrically while its PACF cuts off after lag 1; MA(1) ACF cuts off after lag 1 while its PACF tails off. `ggtsdisplay()` integrates the time-series plot with ACF and PACF in a single ggplot-style view.

9.3.7 Interpretation

A reporting sentence: "The ACF of the simulated AR(1) tailed off geometrically (lag-1 correlation 0.69, decaying smoothly); the PACF cut off sharply after lag 1 (0.69 at lag 1, well within bands at lags 2 onward), consistent with the AR(1) data-generating process. The MA(1) showed the reverse pattern, confirming Box-Jenkins identification rules." Identify the model class from the patterns before fitting any model.

9.3.8 Practical Tips

- Always difference the series until stationary before reading ACF / PACF on the levels; non-stationary series produce decaying ACFs that mimic AR.
- Use `forecast::ggtsdisplay()` for a combined time-series + ACF + PACF view; faster than three separate plots.
- Significance bands assume a white-noise null; after model fitting, the residual ACF is the key diagnostic and should be approximately within bands.
- Seasonal patterns appear at lags equal to the seasonal period; spikes at $s, 2s, 3s$ on the ACF indicate seasonal AR/MA structure.
- Complex ACF / PACF patterns (mixed ARMA, seasonal-plus-non-seasonal) are difficult to identify by eye; reach for `auto.arima` and verify the chosen orders against the ACF.
- For very long series, even tiny correlations are statistically significant; focus on the magnitude and pattern, not on which lags fall just outside the bands.

9.3.9 R Packages Used

Base R `acf()` and `pacf()`; `forecast::ggtsdisplay()` for ggplot-style integrated diagnostics; `feasts::ACF()`, `PACF()`, `gg_tsdisplay()` for the modern tidyverts equivalent; `astsa::acf2()` for compact two-panel display.

9.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.3.11 See also — labs in this chapter

- Time series basics

9.4 Augmented Dickey-Fuller Test

9.4.1 Introduction

The augmented Dickey-Fuller (ADF) test is the canonical test for unit roots in time series. A unit root corresponds to random-walk-like non-stationarity in which shocks have permanent effects on the level of the series; standard regression and ARIMA(p, 0, q) models give nonsense results when applied to such data. The ADF test asks whether a unit root is present and, if rejected, supports modelling the series as stationary (or trend-stationary). Pairing ADF with KPSS gives complementary information: the two tests have flipped null hypotheses, and agreement strengthens the verdict.

9.4.2 Prerequisites

A working understanding of stationarity, autoregressive models, the difference between mean-reverting and random-walk processes, and the role of differencing in inducing stationarity.

9.4.3 Theory

The ADF test fits the regression

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \sum_{i=1}^{p} \delta_i \Delta y_{t-i} + \varepsilon_t,$$

and tests $\gamma = 0$ (unit root) against $\gamma < 0$ (stationary). The "augmented" p lagged differences absorb autocorrelation in the residuals and validate the test under more general AR processes than the original Dickey-Fuller. Under the unit-root null, the test statistic does not follow a standard normal distribution; tabulated Dickey-Fuller critical values are used.

Three deterministic specifications are common: no intercept (rare), intercept only ("drift"), and intercept plus linear trend. The choice should reflect the visible behaviour of the series.

9.4.4 Assumptions

A linear autoregressive structure; no structural breaks (which invalidate the test); the lag order p is large enough to whiten the residuals.

9.4.5 R Implementation

```
library(tseries); library(urca)

x <- cumsum(rnorm(100))     # random walk = unit root
```

```
adf.test(x)                # default lag order
adf.test(diff(x))          # stationary after differencing

# urca version with formal output
summary(ur.df(x, type = "drift", selectlags = "AIC"))
```

9.4.6 Output & Results

`adf.test()` returns the ADF statistic and a p-value computed from the Dickey-Fuller distribution. `urca::ur.df()` provides richer output with critical values at multiple significance levels and automatic lag selection by AIC or BIC.

9.4.7 Interpretation

A reporting sentence: "The ADF test on the random-walk series gave a statistic of -1.50 ($p = 0.52$, fail to reject unit root); after first-differencing, ADF gave -9.83 ($p < 0.01$, reject unit root in favour of stationarity), confirming the series is integrated of order 1." Always state the deterministic specification (drift, trend, none) and the lag order chosen.

9.4.8 Practical Tips

- Include an intercept (`type="drift"`) for series that fluctuate around a non-zero mean; add a trend (`type="trend"`) for series with a visible linear trend.
- Use `selectlags="AIC"` or `"BIC"` in `urca::ur.df()` for automatic lag selection; the default in `tseries::adf.test()` is `trunc(...)` which is sometimes too small.
- Structural breaks invalidate ADF; if a break is plausible, use Zivot-Andrews (`urca::ur.za()`) which extends the test to allow one endogenous break.
- Pair with KPSS for the complementary perspective: agreement (ADF rejects, KPSS does not) confirms stationarity; the reverse confirms a unit root.
- Results are sensitive to specification; try several deterministic options and lag orders, and report a robust summary if conclusions hold across them.
- For multi-series unit-root testing on panel data, use Im-Pesaran-Shin (`plm::purtest`) or Levin-Lin-Chu.

9.4.9 R Packages Used

`tseries::adf.test()` for the canonical ADF; `urca::ur.df()` for richer output and lag-selection tools; `urca::ur.za()` for the Zivot-Andrews test with structural breaks; `forecast::ndiffs()` for an integrated workflow that determines

the required differencing order from KPSS / ADF / Phillips-Perron.

9.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.4.11 See also — labs in this chapter

- Time series basics

9.5 ARIMA Models

9.5.1 Introduction

ARIMA models are the classical workhorse of time-series forecasting. They combine three ideas: autoregression (the current value depends on lagged values), differencing (the current value depends on changes rather than levels), and moving averages of the errors. An ARIMA model is specified by three integers (p, d, q): the order of autoregression, the order of differencing, and the order of moving averages. With a seasonal extension (SARIMA), the same model captures weekly, monthly, or annual cycles.

9.5.2 Prerequisites

The reader should understand the concepts of a stationary time series, autocorrelation, and white noise, and should have seen an ACF/PACF plot before.

9.5.3 Theory

An ARMA(p, q) model for a stationary series Y_t is

$$Y_t = c + \phi_1 Y_{t-1} + ... + \phi_p Y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + ... + \theta_q \varepsilon_{t-q},$$

where ε_t is white noise. When the series is non-stationary, differencing is applied: $\Delta Y_t = Y_t - Y_{t-1}$, or more generally $\Delta^d Y_t$. An ARIMA(p, d, q) model is an ARMA(p, q) model fit to the d-th-order differenced series.

For seasonal data with period s, SARIMA$(p, d, q)(P, D, Q)_s$ adds seasonal autoregressive, differencing, and moving-average terms at lag s. A monthly retail sales series might be well described by a SARIMA$(1, 1, 1)(0, 1, 1)_{12}$.

Model identification traditionally follows the Box-Jenkins methodology:

1. Check stationarity; difference until stationary.
2. Inspect the ACF and PACF of the differenced series; the decay patterns suggest p and q.
3. Fit candidate models and choose by AIC/BIC.
4. Diagnose residuals: they should be white noise.

In practice, `auto.arima()` automates steps 1-3 and reliably finds a good model.

9.5.4 Assumptions

ARIMA models assume:

1. The differenced series is stationary (constant mean, constant variance, autocorrelation depending only on lag).
2. Errors are uncorrelated and have constant variance (homoscedastic).
3. Errors are approximately normally distributed (for prediction-interval validity).

Violations of normality affect interval coverage but not point forecasts; violations of stationarity invalidate the whole model.

9.5.5 R Implementation

The `forecast` package provides the classical interface; the newer `fable` package integrates with the tidyverse.

```
library(forecast)

data(AirPassengers)
plot(AirPassengers)

fit <- auto.arima(AirPassengers)
summary(fit)

checkresiduals(fit)

fc <- forecast(fit, h = 24)
autoplot(fc) +
  labs(x = "Year", y = "Air passengers (thousands)",
       title = "24-month ahead forecast with 80/95% PIs")
```

`auto.arima()` searches over a range of $(p, d, q)(P, D, Q)_{12}$ specifications and returns the best by corrected AIC. `checkresiduals()` plots the residual time series, ACF, histogram, and runs a Ljung-Box test for remaining autocorrelation.

9.5.6 Output & Results

For the `AirPassengers` series, `auto.arima()` selects an ARIMA$(0,1,1)(0,1,1)_{12}$ model on the log scale (with automatic Box-Cox transformation). The Ljung-Box test on residuals gives $p \approx 0.5$, indicating no remaining autocorrelation. The forecast plot shows point predictions with 80% and 95% prediction intervals that fan out over the 24-month horizon.

9.5.7 Interpretation

The model says that after seasonal and regular differencing on the log scale, the series is well described by a simple MA(1) structure in both the regular and seasonal dimensions. The growing width of the prediction intervals reflects accumulating uncertainty – a feature, not a bug, of any honest forecast.

9.5.8 Practical Tips

- Always inspect the series visually before and after differencing; automated tools can miss obvious issues like level shifts or outlier clusters.
- Use information criteria (AICc preferred for small samples) for model comparison, not p-values.
- Validate forecasts on a held-out tail of the series (train/test split in time), not by in-sample fit.
- If the series shows clear seasonality, try both SARIMA and ETS (`ets()`) and let cross-validated accuracy choose.
- For very long series with complex seasonality, consider `prophet`, `fable::fable`, or structural time-series models instead of ARIMA.

9.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.5.10 See also — labs in this chapter

- Time series basics

9.6 Bayesian Changepoint Detection

9.6.1 Introduction

Bayesian changepoint detection returns a posterior probability of a changepoint at each time, rather than a hard list of detected change times. The richer output

quantifies uncertainty about both the existence and the location of changes, making it preferable to classical methods (PELT, binary segmentation) when the analyst wants to communicate "the signal probably shifted near time 200, with some uncertainty about the exact time" rather than committing to an arbitrary threshold. Bayesian approaches also support online detection as data stream in.

9.6.2 Prerequisites

A working understanding of Bayesian inference, classical changepoint detection, and the difference between offline (batch) and online (streaming) analysis.

9.6.3 Theory

Offline Bayesian changepoint detection (Barry-Hartigan 1993, Fearnhead 2006) places a prior on the number and locations of changepoints plus segment parameters and samples from the joint posterior via Gibbs sampling, particle filters, or recursive forward-backward algorithms. The output is a marginal posterior probability of changepoint at each time and posterior summaries of segment parameters.

Online Bayesian changepoint detection (BOCD) (Adams & MacKay 2007) computes the run-length distribution — the posterior probability that the time since the last changepoint equals r — recursively as each new data point arrives. The run-length distribution updates in $O(t)$ per step, making the method viable for streaming applications.

9.6.4 Assumptions

Segment-parameter priors are chosen appropriately for the data; conditional independence between segments given the changepoint structure; the chosen segment model (Normal mean, regression coefficients) matches the underlying signal.

9.6.5 R Implementation

```r
library(bcp)

set.seed(2026)
x <- c(rnorm(100, 0, 1), rnorm(100, 3, 1), rnorm(100, -1, 1))

fit <- bcp(x)
plot(fit)
fit$posterior.prob   # per-time changepoint probability
```

9.6.6 Output & Results

`bcp()` returns the posterior changepoint probability at each time and the posterior mean of the segmented signal. The plot displays both — the smooth signal estimate above and the posterior probability below — making it easy to identify clearly supported changepoints (probability spikes near 1) versus uncertain candidates.

9.6.7 Interpretation

A reporting sentence: "Bayesian changepoint detection returned posterior changepoint probabilities exceeding 0.95 at times 100 and 200, recovering the simulated three-segment structure; the second changepoint had a slightly tighter posterior distribution (probability mass concentrated within ± 1 time step) than the first, reflecting the larger between-segment mean difference at that point." Communicate posterior probabilities, not hard threshold-based detections.

9.6.8 Practical Tips

- Tune the hyperparameter `w0` (prior on between-changepoint variance) for stability; larger values favour fewer, sharper changepoints.
- Online variants (`ocp` package) implement BOCD for real-time detection; useful in monitoring applications where decisions must be made as data arrive.
- Visualise the posterior probability time series rather than extracting hard changepoints by thresholding; thresholding throws away the uncertainty information that motivates the Bayesian approach.
- Small samples give diffuse posteriors; in tight cases, the posterior probability at the true changepoint may peak at only 0.5 to 0.7. More data sharpens the posterior.
- For multivariate changepoint detection, `MultiBNPRiver` and `mbcp` extend the framework to vector-valued series, at substantially higher computational cost.
- Compare against classical methods (PELT) on the same data; Bayesian methods are richer in output but classical methods are typically much faster.

9.6.9 R Packages Used

`bcp` for the canonical offline Barry-Hartigan implementation; `ocp` for online Bayesian changepoint detection; `mcp` for Bayesian changepoint regression with explicit segment models; `bnpa` for Bayesian non-parametric changepoint detection; `Rbeast` for Bayesian time-series decomposition with changepoint awareness.

9.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.6.11 See also — labs in this chapter

- Time series basics

9.7 Changepoint Detection

9.7.1 Introduction

Changepoint detection identifies the times at which the statistical properties of a time series — mean, variance, regression coefficients, or full distribution — change abruptly. Applications range from monitoring physiological signals (heart-rate or EEG state changes), to detecting regime shifts in financial and economic series, to segmenting long sensor recordings into homogeneous epochs. The output is a list of changepoint times together with the within-segment summaries between them; the methodological choices concern the change type, the detection algorithm, and the penalty controlling sensitivity.

9.7.2 Prerequisites

A working understanding of hypothesis testing, segmentation, and the trade-off between detecting true changepoints and false positives.

9.7.3 Theory

Three algorithm classes dominate:

- **Binary segmentation**: recursively detect a single changepoint, partition the series, and apply the same procedure to each segment. Fast but greedy; can miss configurations of multiple nearby changepoints.
- **PELT** (Pruned Exact Linear Time): finds the optimal segmentation under a penalty in $O(n)$ when the penalty satisfies a regularity condition. The standard modern method.
- **Segment Neighbourhood**: dynamic programming for the optimal partition under a fixed maximum number of changepoints.

The penalty (BIC, MBIC, Hannan-Quinn, or AIC) controls how many changepoints are reported; a larger penalty yields fewer changepoints. Different change types have different test statistics: mean changes use the likelihood-ratio test

under a Normal model; variance changes use the same with σ as the parameter; meanvar combines both; regression changepoints generalise to coefficient breaks.

9.7.4 Assumptions

Independent observations within segments (or correctly specified within-segment dependence structure); the change type matches the underlying signal (mean shifts vs variance shifts vs regression breaks).

9.7.5 R Implementation

```r
library(changepoint)

set.seed(2026)
x <- c(rnorm(100, 0, 1), rnorm(100, 3, 1), rnorm(100, -1, 1))

# Mean changes via PELT
cp <- cpt.mean(x, method = "PELT", penalty = "BIC")
cpts(cp)
plot(cp)

# Variance changes
cp_v <- cpt.var(x, method = "PELT")
cpts(cp_v)
```

9.7.6 Output & Results

`cpt.mean()` returns a `cpt` object with the changepoint times accessible via `cpts()`, the within-segment means via `param.est()`, and a plot showing the detected segments. The simulated three-segment structure is recovered with changepoints at times 100 and 200.

9.7.7 Interpretation

A reporting sentence: "PELT with BIC penalty detected two changepoints at times 100 and 200, recovering the simulated three-segment mean structure (segment means 0.04, 2.95, −0.97); a sensitivity analysis with MBIC and Hannan-Quinn penalties yielded the same changepoints." Always report the algorithm, the penalty, and a sensitivity check across penalties.

9.7.8 Practical Tips

- Choose the right change type: `cpt.mean` for mean shifts, `cpt.var` for variance shifts, `cpt.meanvar` for both, regression-based methods for coefficient breaks.

- The penalty controls false positives; try several (BIC, MBIC, Hannan-Quinn) and check that conclusions are robust.
- For non-stationary noise within segments, model the noise (ARIMA) first and apply changepoint detection to the residuals.
- Bayesian alternatives (`bcp::bcp`) give per-time changepoint posterior probabilities, which are more informative than a hard list of detected changepoints.
- PELT is offline — it requires the full series. For online (real-time) detection, use CUSUM, Page-Hinkley, or Bayesian online methods.
- For multivariate changepoint detection, `ecp` and `InspectChangepoint` extend PELT-style methods to vector time series.

9.7.9 R Packages Used

`changepoint` for `cpt.mean()`, `cpt.var()`, `cpt.meanvar()`, and PELT/Binary Segmentation algorithms; `bcp` for Bayesian changepoint detection with posterior probabilities; `ecp` for non-parametric multivariate changepoints; `cpm` for online changepoint methods.

9.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.7.11 See also — labs in this chapter

- Time series basics

9.8 Diebold-Mariano Test

9.8.1 Introduction

The Diebold-Mariano test compares the forecasting accuracy of two competing models on a held-out test window. Where naive accuracy comparisons (one model has lower MSE than another) ignore the statistical noise in the difference, the DM test treats the loss-difference series as a stochastic process and asks whether its mean is significantly non-zero. The test works with any loss function and accommodates the autocorrelation that multi-step-ahead forecast errors typically exhibit, making it the standard inferential tool for forecast comparisons.

9.8.2 Prerequisites

A working understanding of forecast accuracy metrics, the difference between in-sample and out-of-sample evaluation, and basic familiarity with autocorrelation in residual series.

9.8.3 Theory

For two forecasts of the same series with errors $e_{1,t}$ and $e_{2,t}$ on a test window of length T, define $d_t = L(e_{1,t}) - L(e_{2,t})$ where L is a chosen loss (squared error, absolute error, log-squared error). The null is $\mathbb{E}[d_t] = 0$ — equal expected forecast accuracy. The test statistic is the standardised mean of d_t with a long-run-variance estimator that accounts for the natural autocorrelation in the loss-difference series.

The Harvey-Leybourne-Newbold (HLN) small-sample correction adjusts the asymptotic Normal reference for finite T and forecast horizon h, reducing the over-rejection of the original DM test in small samples.

9.8.4 Assumptions

The loss-difference series is covariance-stationary; forecast errors are observable on overlapping or sequential test sets; the chosen loss is appropriate for the forecasting question.

9.8.5 R Implementation

```
library(forecast)

train <- window(AirPassengers, end = c(1957, 12))
test  <- window(AirPassengers, start = c(1958, 1))

e1 <- test - forecast(auto.arima(train), h = length(test))$mean
e2 <- test - forecast(ets(train), h = length(test))$mean

dm.test(e1, e2, alternative = "two.sided", h = 1, power = 2)
```

9.8.6 Output & Results

`dm.test()` returns the DM statistic, p-value, and the chosen lag for autocovariance estimation. The HLN correction is applied by default in `forecast::dm.test()`.

9.8.7 Interpretation

A reporting sentence: “On the held-out test window of three years (1958–1960), the Diebold-Mariano test comparing ARIMA and ETS forecasts at one-step-

ahead horizon gave DM = −1.42 with the HLN small-sample correction (p = 0.16); no significant difference in forecast accuracy under squared-error loss." Always state the test window, forecast horizon, and loss function.

9.8.8 Practical Tips

- Use `power=2` for squared-error loss (default), `power=1` for absolute error; choose based on the substantive forecasting cost function.
- For multi-step-ahead horizons, set `h` to the forecast horizon; the long-run variance uses h to determine the truncation lag for autocovariance estimation.
- Report the test window length, forecast horizon, and chosen loss; results depend on all three.
- For model selection across many candidate models, the DM test is pairwise; the model-confidence-set approach (`MCS::mcsTest`) extends to many models with multiple-comparison correction.
- The DM test compares forecast methods, not models per se; methods that produce the same point forecast (different prediction-interval widths) may be DM-equivalent despite different underlying assumptions.
- For nested models, the conventional DM test under-covers; use Clark-West or West tests instead.

9.8.9 R Packages Used

`forecast::dm.test()` for the canonical implementation with HLN correction; `MCS` for the model-confidence-set extension to multi-model comparisons; `multDM` for multivariate DM tests; `multMCS` for MCS extensions to multivariate forecasts.

9.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.8.11 See also — labs in this chapter

- Time series basics

9.9 Differencing a Time Series

9.9.1 Introduction

Differencing is the simplest tool for making a non-stationary series stationary: replace each observation by its difference from the previous one. First differences

remove linear trends; seasonal differences remove fixed-period seasonality; combined regular and seasonal differencing handles both. The order of differencing d in ARIMA(p, d, q) is exactly the count of first differences applied; the seasonal differencing order D in SARIMA$(p, d, q)(P, D, Q)_s$ counts seasonal differences.

9.9.2 Prerequisites

A working understanding of stationarity, AR models, and the relationship between trend, seasonality, and the spectrum of a series.

9.9.3 Theory

Standard differencing operations:

- **First differences**: $\Delta y_t = y_t - y_{t-1}$. Removes linear (level-shift) non-stationarity.
- **Second differences**: $\Delta^2 y_t = \Delta y_t - \Delta y_{t-1}$. Removes quadratic trend. Rarely needed in practice; second differences typically over-difference and inflate residual variance.
- **Seasonal differences** at period s: $\Delta_s y_t = y_t - y_{t-s}$. Removes constant seasonal patterns at frequency $1/s$.
- **Combined regular and seasonal**: $\Delta\Delta_s y_t = (y_t - y_{t-1}) - (y_{t-s} - y_{t-s-1})$.

Over-differencing introduces unnecessary moving-average structure (the differenced series has a strong negative lag-1 autocorrelation); always use the minimum differencing order needed for stationarity. KPSS- or ADF-based automated selection (`forecast::ndiffs` and `nsdiffs`) chooses the required orders.

9.9.4 Assumptions

The non-stationarity is of the kind that differencing addresses (stochastic trend or fixed-period seasonality); deterministic polynomial trends are better handled by detrending regression.

9.9.5 R Implementation

```r
library(forecast)

x <- AirPassengers
plot(x)

# First difference
plot(diff(x))

# Seasonal difference (lag = 12 for monthly)
plot(diff(x, lag = 12))
```

```r
# Both
plot(diff(diff(x, lag = 12)))

# Automatic: number of differences needed
ndiffs(log(x))         # regular
nsdiffs(log(x))        # seasonal
```

9.9.6 Output & Results

`diff(x,lag=s,differences=d)` produces $\Delta_s^d y$. `ndiffs()` returns the required regular differencing order based on KPSS by default; `nsdiffs()` returns the seasonal differencing order. For the log-AirPassengers series, both regular and seasonal differencing once each produces a stationary residual.

9.9.7 Interpretation

A reporting sentence: "Monthly air-passenger counts on the log scale required one regular difference and one seasonal difference at lag 12 to achieve stationarity (KPSS $p > 0.10$ on the doubly-differenced series); the resulting series motivates an ARIMA$(0,1,1)(0,1,1)_{12}$ model on the log scale, equivalent to a multiplicative seasonal ARIMA on the original scale." Always confirm stationarity after differencing.

9.9.8 Practical Tips

- Log-transform first for series with variance that grows with the level; the log scale is then additive and the differencing is well-behaved.
- ARIMA(p,d,q) encodes the differencing order d; the corresponding `Arima()` call automatically applies the difference and works with the original series for prediction.
- `forecast::ndiffs()` and `nsdiffs()` automate the choice of regular and seasonal differencing orders; useful as a starting point for `auto.arima`.
- Over-differencing inflates residual variance and introduces a strong negative MA(1) coefficient; if you see θ_1 near -0.95 in a differenced model, consider whether the differencing was excessive.
- Always confirm stationarity after differencing with ADF and KPSS; if either test still indicates non-stationarity, examine the data for structural breaks or non-linear trends.
- For deterministic polynomial trends, detrending regression is more efficient than differencing; ARIMA's stochastic-trend interpretation differs.

9.9.9 R Packages Used

Base R `diff()` for differencing operations; `forecast::ndiffs()`, `nsdiffs()`, and `auto.arima()` for automated differencing decisions; `feasts::unitroot_ndiffs()`

and `unitroot_nsdiffs()` for the modern tidyverts equivalents; `tseries::adf.test()` and `kpss.test()` for confirming stationarity after differencing.

9.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.9.11 See also — labs in this chapter

- Time series basics

9.10 ETS Models

9.10.1 Introduction

ETS models — Error, Trend, Seasonal — express the exponential-smoothing family in a unified state-space form. Each component (error, trend, seasonal) is independently classified as additive, multiplicative, or absent, giving a small library of well-understood models. The state-space framework allows proper likelihood-based estimation, AIC-based model selection, and principled prediction intervals derived from the underlying stochastic structure rather than ad-hoc heuristics.

9.10.2 Prerequisites

A working understanding of exponential smoothing (simple, Holt's, Holt-Winters), state-space models, and the additive vs multiplicative distinction for time-series components.

9.10.3 Theory

The notation ETS(E, T, S) classifies each component:

- **Error**: A (additive) or M (multiplicative).
- **Trend**: N (none), A (additive), Ad (damped additive), M (multiplicative), Md (damped multiplicative).
- **Seasonal**: N (none), A (additive), or M (multiplicative).

Examples: ETS(A, N, N) is simple exponential smoothing; ETS(A, A, N) is Holt's linear trend; ETS(A, A, A) is additive Holt-Winters; ETS(M, A, M) gives multiplicative errors plus additive trend plus multiplicative seasonality. `forecast::ets()` enumerates valid candidates and selects by AIC.

The state-space form provides exact likelihood, AIC for model selection, and analytic (or simulation-based) prediction intervals. Multiplicative components require strictly positive data; the AIC-based selection avoids choosing an invalid model.

9.10.4 Assumptions

The chosen ETS specification matches the data's components; for prediction intervals, the innovations are approximately Normal (or simulated from the fitted residual distribution).

9.10.5 R Implementation

```r
library(forecast)

x <- AirPassengers

fit <- ets(x)
summary(fit)
autoplot(fit)

fc <- forecast(fit, h = 24)
autoplot(fc)

# Residual diagnostics
checkresiduals(fit)
```

9.10.6 Output & Results

`summary(fit)` returns the chosen model code (e.g., "M, A, M"), smoothing parameters α, β, γ, AIC, and BIC. `forecast()` produces point forecasts and 80 % / 95 % prediction intervals from the fitted state-space form. `checkresiduals()` provides Ljung-Box and graphical diagnostics.

9.10.7 Interpretation

A reporting sentence: "`ets()` selected ETS(M, A, M) for the airline series ($\alpha = 0.34$, $\beta = 0.01$, $\gamma = 1.0$, AIC = 1{,}393); the model captures the upward trend and the multiplicative seasonality whose amplitude grows with the level. Standardised residuals passed Ljung-Box at lag 24 ($p = 0.61$)." Always state the chosen ETS specification and the smoothing parameters.

9.10.8 Practical Tips

- `ets()` automatically selects the optimal ETS variant by AIC; override with `model="AAA"` etc. when a specific form is required for replication.

- Multiplicative components require strictly positive data; AIC-based selection avoids invalid choices automatically.
- For complex multi-seasonality (e.g. weekly + yearly), TBATS (`forecast::tbats()`) extends ETS with multiple seasonal layers and Box-Cox transformation.
- ETS prediction intervals can be very wide at long horizons because the multiplicative variance compounds; for a more nuanced view, simulate many sample paths via `forecast(fit,simulate=TRUE,npaths=1000)`.
- Compare ETS against ARIMA via rolling-origin cross-validation; both are strong baseline forecasters and the better choice depends on the series.
- For damped trend (which extrapolates more conservatively at long horizons), use `model="AAdA"` or let `ets()` discover it through AIC.

9.10.9 R Packages Used

`forecast::ets()` for the canonical ETS implementation with AIC-based selection; `fable::ETS()` for the modern tidyverts equivalent; `forecast::tbats()` for multiple seasonalities and Box-Cox transformation; `smooth::es()` for advanced ETS variants including ETS plus regressors.

9.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.10.11 See also — labs in this chapter

- Time series basics

9.11 Exponential Smoothing

9.11.1 Introduction

Exponential smoothing is a family of forecasting methods that produces predictions as weighted averages of past observations with geometrically declining weights. The simplest form (simple exponential smoothing, SES) handles series with no trend and no seasonality; Holt's method adds a linear trend; Holt-Winters extends to seasonal series with additive or multiplicative seasonality. The methods are competitive with ARIMA at short horizons, conceptually simpler, and the natural choice when the series structure matches one of the supported variants.

9.11.2 Prerequisites

A working understanding of moving averages, forecasting basics, and the trade-off between adaptiveness and smoothness.

9.11.3 Theory

Simple exponential smoothing recursively updates a level estimate:

$$\hat{y}_{t+1|t} = \alpha y_t + (1-\alpha)\hat{y}_{t|t-1},$$

with smoothing parameter $\alpha \in (0,1)$. Larger α adapts more quickly to recent values; smaller α smooths more aggressively.

Holt's method adds a trend component with its own smoothing parameter β. Holt-Winters adds a seasonal component with parameter γ in either additive (constant seasonal amplitude) or multiplicative (seasonal amplitude grows with level) form. The equivalent state-space representations form the ETS family — Error/Trend/Seasonal — coded as ETS(A, A, A), ETS(M, Ad, M), and so on, with A = additive, M = multiplicative, N = none, Ad = damped additive trend.

9.11.4 Assumptions

The chosen ETS specification matches the data's components; smoothing parameters are constant over time; for prediction intervals, innovations are approximately Normal.

9.11.5 R Implementation

```r
library(forecast)

x <- AirPassengers

# Holt-Winters multiplicative (for multiplicative seasonality)
hw <- HoltWinters(x, seasonal = "multiplicative")
plot(hw)
fc <- forecast(hw, h = 24)
plot(fc)

# Equivalent via ets()
fit_ets <- ets(x)
summary(fit_ets)
plot(forecast(fit_ets, h = 24))
```

9.11.6 Output & Results

`HoltWinters()` returns smoothing parameters and the fitted level, trend, and seasonal components. `ets()` automatically selects the ETS variant by AIC and produces equivalent output with prediction intervals from the state-space form.

9.11.7 Interpretation

A reporting sentence: "Holt-Winters multiplicative smoothing captured the growing seasonal amplitude in airline passengers (smoothing parameters $\alpha = 0.34$, $\beta = 0.01$, $\gamma = 1.0$); 24-month forecasts extend the trend with realistic seasonal variation, and 95 % prediction intervals widen from $\pm 4\%$ at 1 month to $\pm 18\%$ at 24 months." Always state the ETS variant chosen and the smoothing parameter values.

9.11.8 Practical Tips

- `ets()` automatically selects the ETS family by AIC; specify `model` argument explicitly to force a specific structure.
- For multiplicative seasonality, either fit ETS(…,M) directly or log-transform the series and fit ETS(…,A); both approaches give equivalent forecasts.
- Exponential smoothing is competitive with ARIMA for short horizons (1–4 periods); ARIMA tends to win at longer horizons.
- Prediction intervals from `forecast()` assume Normal innovations; check residuals for skewness and heavy tails.
- For short series ($T < 30$), simple SES or Holt often beats Holt-Winters because parameter estimation is unstable for the more complex models.
- The damped-trend ETS(…,Ad) variant is a sensible default when the trend is plausibly slowing in the forecast horizon; pure additive trends extrapolate forever.

9.11.9 R Packages Used

Base R `HoltWinters()` for the classical implementation; `forecast::ets()` for the modern state-space ETS family with AIC-based selection; `fable::ETS()` for the tidyverts equivalent; `smooth::es()` for advanced exponential-smoothing variants including ETS plus regressors.

9.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.11.11 See also — labs in this chapter

- Time series basics

9.12 Forecast Accuracy Metrics

9.12.1 Introduction

Forecast accuracy is evaluated through error metrics that compare predictions against held-out actuals. Each metric encodes a different loss function and has different units, so the choice of metric depends on the substantive cost structure of forecasting errors. The four most widely reported are MAE, RMSE, MAPE, and MASE; understanding their differences and trade-offs is essential for fair comparison across forecasting methods and across series.

9.12.2 Prerequisites

A working understanding of forecasting, the difference between in-sample and out-of-sample evaluation, and basic loss-function concepts.

9.12.3 Theory

Let $e_t = y_t - \hat{y}_t$ be the forecast error on the test window of length n.

- **Mean Absolute Error**: MAE $= \frac{1}{n}\sum |e_t|$. Scale-dependent, same units as the data; treats all errors equally.
- **Root Mean Squared Error**: RMSE $= \sqrt{\frac{1}{n}\sum e_t^2}$. Scale-dependent, penalises large errors more heavily; the natural metric when squared loss is the cost function.
- **Mean Absolute Percentage Error**: MAPE $= \frac{100}{n}\sum |e_t/y_t|$. Scale-independent percentage; undefined for $y_t = 0$ and asymmetric (penalises over-forecasts more than under-forecasts at the same absolute level).
- **Mean Absolute Scaled Error**: MASE $=$ MAE$/\bar{e}_{\text{naive}}$, scaled by the in-sample MAE of a naive forecast. Scale-independent, symmetric, defined for all data (including zeros). Values below 1 mean "better than naive seasonal forecast in sample".

For probabilistic forecasts, the continuous ranked probability score (CRPS) generalises MAE to forecast distributions and is the standard scoring rule for prediction intervals.

9.12.4 Assumptions

A comparable held-out test window of sufficient length to give stable error estimates; the chosen metric matches the substantive loss function of forecasting errors.

9.12.5 R Implementation

```r
library(forecast)

x <- AirPassengers
train <- window(x, end = c(1957, 12))
test  <- window(x, start = c(1958, 1))

fit  <- auto.arima(train)
fc   <- forecast(fit, h = length(test))
accuracy(fc, test)
```

9.12.6 Output & Results

`accuracy()` returns a table with ME (mean error, indicates bias), RMSE, MAE, MPE, MAPE, MASE, and ACF1 (lag-1 autocorrelation of residuals) on both the training and test sets. The training-set numbers measure in-sample fit; the test-set numbers are the genuine out-of-sample predictive performance.

9.12.7 Interpretation

A reporting sentence: "On the held-out three-year window, the ARIMA forecast achieved test-set RMSE of 22.8 passengers, MAE of 18.5, and MASE of 0.42; the MASE below 1 confirms substantially better predictive performance than a naive seasonal forecast." Always report multiple metrics — RMSE for squared-error context, MAE for absolute-error context, MASE for scale-free comparison.

9.12.8 Practical Tips

- Report multiple metrics; they reward different error properties and a model can dominate on one while losing on another.
- MAPE is popular in business reporting but misleading for series with values near zero or with negative values; use sMAPE or MASE instead.
- MASE is the recommended scale-free metric in academic forecasting evaluation; it is symmetric, defined for zeros, and benchmarked against a meaningful baseline.
- Evaluate on a held-out tail (chronological train/test split), never in-sample; in-sample errors are systematically optimistic.
- For comparing many forecasts across many series, aggregate metrics via rolling-origin cross-validation; a single train/test split is sensitive to the choice of cutoff.
- For probabilistic forecasts, report CRPS or interval scores; point-forecast metrics ignore the prediction-interval calibration.

9.12.9 R Packages Used

`forecast::accuracy()` for the standard suite of point-forecast metrics; `Metrics::rmse()`, `mae()`, `mape()` for individual metrics; `scoringRules` for proper scoring rules including CRPS for probabilistic forecasts; `fabletools::accuracy()` for the modern tidyverts equivalent.

9.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.12.11 See also — labs in this chapter

- Time series basics

9.13 GARCH Models

9.13.1 Introduction

GARCH (Generalised Autoregressive Conditional Heteroscedasticity) models capture time-varying conditional variance — the volatility clustering pattern in which quiet periods alternate with volatile ones. Financial returns are the prototypical application: stock prices show stretches of low daily variability punctuated by high-volatility crisis episodes, a structure that conditional-Normal ARMA models cannot accommodate. GARCH extends ARMA by adding an autoregressive equation for the conditional variance itself, making volatility a forecastable quantity.

9.13.2 Prerequisites

A working understanding of ARMA models, conditional distributions, and the distinction between unconditional and conditional variance.

9.13.3 Theory

The GARCH(1, 1) model decomposes returns as

$$r_t = \mu + \varepsilon_t, \qquad \varepsilon_t = \sigma_t z_t, \quad z_t \sim \mathcal{N}(0,1),$$

with the conditional variance evolving as

$$\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2.$$

The mean structure can be combined with an ARMA component; GARCH specifically models the variance of the innovations. Stationary unconditional variance requires $\alpha+\beta<1$; values close to 1 indicate highly persistent volatility (typical for daily financial returns).

Extensions handle asymmetric responses to positive vs negative shocks (EGARCH, TGARCH, APARCH), variance feedback into the mean (GARCH-M), and multivariate volatility (DCC, BEKK). For non-Normal returns, t- or skewed-t-distributed innovations are standard.

9.13.4 Assumptions

Conditional Normality (or t, skewed-t) of innovations z_t; stationarity of the conditional variance ($\alpha+\beta<1$); the chosen GARCH order is appropriate for the data.

9.13.5 R Implementation

```r
library(rugarch)

# Simulate GARCH(1,1) returns
spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder =
 ↪  c(1, 1)),
                   mean.model = list(armaOrder = c(0, 0), include.mean =
                    ↪  TRUE),
                   distribution.model = "norm")

set.seed(2026)
sim <- ugarchpath(spec = ugarchspec(
  variance.model = list(garchOrder = c(1, 1)),
  mean.model = list(armaOrder = c(0, 0))),
  n.sim = 500)
r <- as.numeric(fitted(sim))

# Fit
fit <- ugarchfit(spec, data = r)
show(fit)
plot(fit, which = 3)   # conditional sd
```

9.13.6 Output & Results

`ugarchfit()` returns parameter estimates (ω, α, β, mean parameters), log-likelihood, AIC/BIC, and Ljung-Box-style tests on standardised and squared

standardised residuals. The conditional-standard-deviation plot shows the time-varying volatility extracted by the model.

9.13.7 Interpretation

A reporting sentence: "A GARCH(1, 1) fit to daily returns estimated $\hat{\omega} = 0.0001$, $\hat{\alpha} = 0.08$, $\hat{\beta} = 0.91$ ($\hat{\alpha} + \hat{\beta} = 0.99$, very persistent volatility); standardised residuals showed no remaining ARCH effects (Ljung-Box on squared residuals $p = 0.42$). Conditional-volatility forecasts widen modestly over the 30-day horizon." Always report the persistence sum and standardised-residual diagnostics.

9.13.8 Practical Tips

- Check standardised residuals (`residuals(fit,standardize=TRUE)`); they should be approximately iid and often heavier-tailed than Normal, motivating t or skewed-t innovations.
- For asymmetric responses to positive vs negative shocks (the leverage effect in equity returns), use EGARCH or TGARCH; standard GARCH is symmetric and misses this.
- `rugarch` is the industry standard for univariate GARCH; for multivariate volatility (covariance matrices), `rmgarch` extends to DCC, BEKK, and copula-GARCH.
- Value-at-Risk forecasts come from the conditional-variance forecast plus a quantile of the innovation distribution; `ugarchforecast()` and `ugarchroll()` support rolling VaR estimation.
- GARCH on returns is conventional, but the framework applies to any series with volatility clustering — physiological signals, atmospheric data, traffic counts.
- For very long series, the parameter estimates can be stable but the conditional-volatility series may need recursive re-estimation; use `ugarchroll()` for genuine out-of-sample evaluation.

9.13.9 R Packages Used

`rugarch` for univariate GARCH and its many extensions; `rmgarch` for multivariate GARCH (DCC, BEKK); `fGarch` as an alternative implementation; `bayesGARCH` for Bayesian GARCH; `tsgarch` for the modern tidyverse-flavoured interface.

9.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.13.11 See also — labs in this chapter

- Time series basics

9.14 Granger Causality

9.14.1 Introduction

Granger causality tests whether past values of one time series improve the prediction of another, beyond what is already explained by the latter's own history. It is a *statistical* concept, not a *causal* one in the philosophical sense: a series Granger-causes another if including its lags reduces the residual variance of the predicted series, which is a useful but limited notion. Two series can Granger-cause each other without any direct causal link if both are driven by a common unobserved factor.

9.14.2 Prerequisites

A working understanding of vector autoregressive (VAR) models, F-tests on joint coefficient hypotheses, and the difference between predictive and causal claims.

9.14.3 Theory

Series X Granger-causes series Y at lag p if the lagged values $X_{t-1}, \dots, X_{t-p}$ significantly improve the regression of Y_t on its own lags $Y_{t-1}, \dots, Y_{t-p}$. The test is an F-test on the joint hypothesis that all coefficients on lagged X are zero. Within a VAR system with K series, Granger causality from one series to another (or to all others) is computed similarly with the residuals jointly estimated.

A non-trivial extension is "instantaneous" causality: X and Y contemporaneously correlated after conditioning on lagged values of both. This is included by some implementations (e.g., `vars::causality`).

9.14.4 Assumptions

Stationary series — the test is invalid for unit-root processes. No unmeasured common cause that drives both series; otherwise the Granger-causal direction is uninterpretable.

9.14.5 R Implementation

```
library(lmtest); library(vars)

data(ChickEgg)   # eggs / chickens
```

```r
# Does egg production Granger-cause chicken population?
grangertest(chicken ~ egg, order = 3, data = ChickEgg)
grangertest(egg ~ chicken, order = 3, data = ChickEgg)

# Within a VAR
fit <- VAR(ChickEgg, p = 3)
causality(fit, cause = "egg")$Granger
causality(fit, cause = "chicken")$Granger
```

9.14.6 Output & Results

`grangertest()` returns the F-statistic, degrees of freedom, and p-value for the joint hypothesis of zero coefficients on the cause's lags. `vars::causality()` extends this within a VAR framework, handling the multivariate residual structure correctly and adding the instantaneous-causality test.

9.14.7 Interpretation

A reporting sentence: "Egg production Granger-caused chicken population at lag 3 ($F(3, 48) = 4.9$, $p = 0.005$); the reverse test was not significant ($p = 0.39$). This suggests egg counts contain predictive information about future chicken populations beyond past chicken counts, but does not establish a direct causal mechanism." Always frame Granger results as predictive, not causal.

9.14.8 Practical Tips

- Granger causality requires stationarity; difference or detrend first using KPSS / ADF guidance.
- Results are sensitive to lag order; try several plausible lags and report the chosen value with justification.
- Granger causality does not imply causation; a common confounder produces apparent bidirectional Granger causality between two effects.
- For more than two series, use the VAR-based `vars::causality()` rather than pairwise `grangertest()`; the joint estimation handles cross-series correlations.
- Non-linear Granger causality exists (`bootGran`, `NlinTS`); the standard tests are linear and miss non-linear predictive relationships.
- For frequency-domain Granger causality, the `vars::fevd()` and Geweke decomposition tools partition predictive contribution by frequency band.

9.14.9 R Packages Used

`lmtest::grangertest()` for the bivariate test; `vars::VAR()` and `causality()` for VAR-based multivariate Granger causality; `tsDyn` for non-linear extensions; `Nli`

`nTS` for non-linear and information-theoretic causality measures.

9.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.14.11 See also — labs in this chapter

- Time series basics

9.15 Holt-Winters Method

9.15.1 Introduction

The Holt-Winters method extends exponential smoothing to handle both trend and seasonality via three coupled smoothing equations — level, trend, and seasonal — each with its own smoothing parameter. It is the seasonal generalisation of Holt's linear method and the conceptual ancestor of the modern ETS family. Its strength is interpretability and competitive forecasting performance with relatively few parameters; its weakness is the assumption of a fixed seasonal period and stable parameters across the entire series.

9.15.2 Prerequisites

A working understanding of simple exponential smoothing, Holt's linear-trend method, and the additive-vs-multiplicative seasonal distinction.

9.15.3 Theory

The additive Holt-Winters equations are

$$\begin{aligned} L_t &= \alpha(y_t - S_{t-s}) + (1-\alpha)(L_{t-1} + T_{t-1}), \\ T_t &= \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}, \\ S_t &= \gamma(y_t - L_t) + (1-\gamma)S_{t-s}. \end{aligned}$$

The forecast is $\hat{y}_{t+h} = L_t + hT_t + S_{t-s+1+(h-1) \bmod s}$. The multiplicative variant replaces the seasonal additions with multiplications and is appropriate when seasonal amplitude grows with the level.

The three smoothing parameters α, β, γ each lie in $[0, 1]$: small values give slow adaptation (smoother estimates), large values give fast adaptation (more responsive but noisier).

9.15.4 Assumptions

The seasonal period s is fixed and known; the smoothing parameters are stable over the series; for prediction intervals, residuals are approximately Normal.

9.15.5 R Implementation

```r
library(forecast)

x <- AirPassengers

# Multiplicative
hw_m <- HoltWinters(x, seasonal = "multiplicative")
hw_m$alpha; hw_m$beta; hw_m$gamma

fc <- forecast(hw_m, h = 24)
plot(fc)
```

9.15.6 Output & Results

`HoltWinters()` returns the three smoothing parameters, the fitted level / trend / seasonal components, and the fitted values on the original series. `forecast()` extends to multi-step-ahead predictions with 80 % / 95 % intervals.

9.15.7 Interpretation

A reporting sentence: "Holt-Winters multiplicative smoothing of monthly air-passenger counts estimated $\hat{\alpha} = 0.20$, $\hat{\beta} = 0.03$, $\hat{\gamma} = 0.80$; the resulting 24-month forecast continues the upward trend with proportionally growing seasonal amplitude. The slow β indicates a stable trend slope, while the fast γ allows the seasonal shape to adapt over time." Always state the smoothing parameters and the additive-vs-multiplicative choice.

9.15.8 Practical Tips

- Use additive Holt-Winters for stable seasonal amplitude; multiplicative when seasonal amplitude scales with the level.
- Log-transforming the series often allows additive Holt-Winters to substitute for multiplicative; both produce essentially equivalent forecasts.
- Parameters α, β, γ near 0 give slow updates (highly smoothed); near 1 give fast adaptation (responsive but noisy).
- For complex seasonality (multiple periods, irregular spacing), the older Holt-Winters fails; use BATS or TBATS (`forecast::bats()`, `tbats()`) which handle these cases.
- `forecast::ets()` automates the choice between additive, multiplicative, and damped variants by AIC; reach for it before `HoltWinters()` for new

analyses.
- Diagnostic check: the residuals should look like white noise; `checkresiduals(fc)` runs Ljung-Box and visual diagnostics.

9.15.9 R Packages Used

Base R `HoltWinters()` for the classical implementation; `forecast::ets()` for the modern state-space ETS family that includes Holt-Winters as ETS(…,A,A) or ETS(…,M,M); `forecast::tbats()` for multi-seasonal extensions; `fable::ETS()` for the tidyverts equivalent.

9.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.15.11 See also — labs in this chapter

- Time series basics

9.16 The Kalman Filter

9.16.1 Introduction

The Kalman filter is the workhorse algorithm for state-space models with linear-Gaussian observation and transition equations. It recursively combines the previous state estimate with each new observation to produce an updated state estimate together with its uncertainty, in a closed-form Gaussian update that is provably optimal under the linear-Gaussian assumptions. Originally developed for aerospace tracking, it now underpins virtually every classical time-series filtering application — economic indicators, signal processing, target tracking, financial volatility estimation.

9.16.2 Prerequisites

A working understanding of state-space models, linear algebra, and the multivariate Normal distribution.

9.16.3 Theory

For a linear-Gaussian state-space model with observation $y_t = Fx_t + v_t$ and transition $x_t = Gx_{t-1} + w_t$, the Kalman filter alternates two steps at each time:

Predict:

$$x_{t|t-1} = G x_{t-1|t-1}, \qquad P_{t|t-1} = G P_{t-1|t-1} G^\top + W.$$

Update:

$$x_{t|t} = x_{t|t-1} + K_t(y_t - F x_{t|t-1}), \qquad P_{t|t} = (I - K_t F) P_{t|t-1},$$

with Kalman gain $K_t = P_{t|t-1} F^\top (F P_{t|t-1} F^\top + V)^{-1}$.

The smoother (Rauch-Tung-Striebel) is a backward recursion that combines forward filter estimates with future observations to produce smoothed estimates $x_{t|T}$. Non-linear extensions include the extended Kalman filter (linearisation around the current estimate) and the unscented Kalman filter (sigma-point sampling); particle filters handle fully non-Gaussian and non-linear models via Monte Carlo.

9.16.4 Assumptions

Linear Gaussian observation and transition equations with known parameters (F, G, V, W). Parameters can be estimated by MLE on the innovations.

9.16.5 R Implementation

```r
library(dlm)

# Local-level model on Nile
mod <- dlmModPoly(order = 1, dV = 15100, dW = 1470)
filt <- dlmFilter(Nile, mod)

plot(Nile)
lines(dropFirst(filt$m), col = "red", lwd = 2)

# Smoother
smo <- dlmSmooth(filt)
lines(dropFirst(smo$s), col = "blue", lwd = 2)
```

9.16.6 Output & Results

`dlmFilter()` returns the filtered state estimates ($x_{t|t}$) and their covariances. `dlmSmooth()` produces the retrospective smoothed estimates ($x_{t|T}$). Plotting both on the original series shows the filter's adaptation in real time and the smoother's retrospective best estimate.

9.16.7 Interpretation

A reporting sentence: "The Kalman filter applied to the Nile river-flow series produced one-step-ahead forecasts (filtered states) that adapt to each new observation; the smoothed estimate, using the full series in retrospect, identified a sharp level drop near 1899 that the forward filter takes several years to track." Distinguish filtered (real-time) from smoothed (retrospective) estimates in any report.

9.16.8 Practical Tips

- For non-linear models, use the extended Kalman filter (`KFAS::EKF` or hand-coded) for mild non-linearity, the unscented Kalman filter for stronger non-linearity, or particle filters (`SMCSamplers`, `SMC2`) for arbitrary non-linear / non-Gaussian models.
- Uncertainty propagates naturally through both predict and update steps; report state estimates with their estimated standard errors.
- The log-likelihood of the model can be computed from the innovations (`dlmLL` in `dlm`) and used for MLE-based parameter estimation; the Kalman filter is the engine, MLE is the parameter-fitting wrapper.
- The choice of initial state $x_{0|0}$ and initial variance $P_{0|0}$ matters for the first few time steps; diffuse initialisation ($P_{0|0}$ very large) is the standard choice when no prior information is available.
- For extremely high-dimensional state spaces (large meteorological models), the ensemble Kalman filter approximates the covariance with a low-rank ensemble; standard Kalman fails in that setting.
- Pair filtering with diagnostic checks on the innovations (one-step-ahead residuals); they should be approximately white noise under correct model specification.

9.16.9 R Packages Used

`dlm` for `dlmFilter()`, `dlmSmooth()`, and `dlmLL()`; `KFAS` for fast filtering with non-Gaussian observations; `FKF` for the fast Kalman filter implementation; `bsts` for Bayesian structural time-series alternatives; `nimble` for hand-built non-linear filters via probabilistic programming.

9.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.16.11 See also — labs in this chapter

- Time series basics

9.17 KPSS Test

9.17.1 Introduction

The Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test is the natural complement to the augmented Dickey-Fuller (ADF) test for assessing stationarity. The two tests have flipped null hypotheses: ADF tests a unit-root null against a stationary alternative; KPSS tests a stationary null against a unit-root alternative. Running both and comparing their conclusions gives a more robust verdict than either alone, because a single test that fails to reject is uninformative — failure to reject a unit-root null is not evidence of stationarity, and vice versa.

9.17.2 Prerequisites

A working understanding of stationarity, unit-root processes (random walks), and the augmented Dickey-Fuller test as the complementary stationarity diagnostic.

9.17.3 Theory

The KPSS framework decomposes $y_t = r_t + \beta t + \varepsilon_t$, where $r_t = r_{t-1} + u_t$ is a random walk. Under the null of stationarity, the variance of the random-walk innovations is zero ($\sigma_u^2 = 0$); under the alternative, r_t contributes a stochastic trend. The test statistic is based on partial sums of residuals from regressing y_t on (optionally) a constant and a trend; large partial-sum behaviour indicates a unit root and the test rejects stationarity.

The "Level" version tests stationarity around a constant mean; the "Trend" version tests stationarity around a deterministic trend. Choosing between them depends on whether the series has a visible time trend.

9.17.4 Assumptions

Correct model specification (level vs trend stationarity); the lag truncation parameter is adequate for the underlying autocorrelation structure (default works for most applications).

9.17.5 R Implementation

```
library(tseries)

# Stationary series
```

```r
y1 <- rnorm(100)
kpss.test(y1, null = "Level")

# Random walk (unit root)
y2 <- cumsum(rnorm(100))
kpss.test(y2, null = "Level")

# Trend-stationary series
y3 <- 0.5 * (1:100) + rnorm(100)
kpss.test(y3, null = "Trend")
```

9.17.6 Output & Results

`kpss.test()` returns the KPSS statistic and the p-value (with a note about the truncation lag and the type of stationarity tested). Small statistics correspond to large p-values and indicate insufficient evidence to reject stationarity; large statistics reject and support a unit root.

9.17.7 Interpretation

A reporting sentence: "The stationary white-noise series gave a KPSS statistic of 0.08 ($p > 0.10$, fail to reject stationarity); the random-walk series gave a statistic of 2.34 ($p < 0.01$, reject stationarity)." Always state which null was tested (Level vs Trend) and pair with an ADF test for the complementary perspective.

9.17.8 Practical Tips

- `null="Level"` for plain stationarity around a constant mean; `"Trend"` for stationarity around a deterministic trend. Choose based on whether a time-trend is visible in the series.
- Bandwidth (lag-truncation) choice affects test size; the default is adequate for most series. For long-memory or strongly autocorrelated series, increase the truncation manually.
- Combine with the ADF test: if both agree (ADF rejects unit root, KPSS fails to reject stationarity), stationarity is well-supported; if both disagree (ADF fails, KPSS rejects), a unit root is well-supported.
- If both tests give ambiguous answers (ADF fails, KPSS fails), the series is borderline and additional diagnostics (autocorrelation structure, Phillips-Perron, fractional integration) help.
- For multivariate or panel data, use Im-Pesaran-Shin or Levin-Lin-Chu tests; KPSS is univariate.
- Differencing the series and re-testing is the standard workflow if a unit root is detected; the differenced series should pass both KPSS and ADF.

9.17.9 R Packages Used

`tseries::kpss.test()` for the canonical implementation; `urca::ur.kpss()` for an alternative with finer control over deterministic components and bandwidth selection; `tseries::adf.test()` for the complementary ADF test; `forecast::ndiffs()` for an integrated workflow that automatically applies KPSS to determine the required differencing order.

9.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.17.11 See also — labs in this chapter

- Time series basics

9.18 The Ljung-Box Test

9.18.1 Introduction

The Ljung-Box test is the standard portmanteau test for residual autocorrelation in time-series models. After fitting an ARIMA, ETS, or other time-series model, the question is whether the residuals are white noise; the Ljung-Box test pools sample autocorrelations over the first several lags into a single chi-squared statistic. A non-significant result supports model adequacy; a significant result indicates missed dynamics that should motivate refining the model.

9.18.2 Prerequisites

A working understanding of autocorrelation functions, ARIMA models, and the white-noise assumption on time-series residuals.

9.18.3 Theory

$$Q = n(n+2)\sum_{k=1}^{h}\frac{\hat{\rho}_k^2}{n-k} \sim \chi^2_{h-p-q} \text{ under white noise,}$$

where $\hat{\rho}_k$ is the sample ACF at lag k, h is the maximum lag, and $p+q$ is the number of ARMA parameters fit. The `fitdf` argument in `Box.test()` adjusts the degrees of freedom for parameters estimated from the data; for residuals from raw data (no model fitted), set `fitdf=0`.

The Ljung-Box improves on the older Box-Pierce statistic by introducing the small-sample correction $n(n+2)$ that prevents under-rejection at small n. For seasonal models, the maximum lag should span at least one seasonal period.

9.18.4 Assumptions

Residuals are iid under the null; the maximum lag h is small relative to the series length n.

9.18.5 R Implementation

```r
set.seed(2026)
x <- arima.sim(list(ar = 0.5), n = 200)
fit <- arima(x, order = c(1, 0, 0))
e <- residuals(fit)

# Ljung-Box at lag 20, 1 parameter fit
Box.test(e, lag = 20, type = "Ljung-Box", fitdf = 1)

# On an unfit series (no fitdf adjustment)
Box.test(x, lag = 20, type = "Ljung-Box")
```

9.18.6 Output & Results

`Box.test()` returns the chi-squared statistic and the p-value at the chosen lag. Pair with the ACF and PACF plots of residuals for a visual confirmation; isolated significant autocorrelations may not move the omnibus test but are still substantively meaningful.

9.18.7 Interpretation

A reporting sentence: "Residuals from the AR(1) fit gave a Ljung-Box statistic at lag 20 of $Q = 18.9$ on 19 degrees of freedom ($p = 0.45$), consistent with white noise. The ACF and PACF of residuals showed no isolated significant lags, supporting model adequacy." Always pair the test result with the lag chosen and the parameters subtracted.

9.18.8 Practical Tips

- Choose the maximum lag h around $\min(10, n/5)$ or based on domain knowledge; for monthly data with annual seasonality, $h \geq 24$ is sensible.
- Use `fitdf=p+q` for ARMA(p, q) fitted models; the test ignores the parameter cost otherwise and is anti-conservative.
- For seasonal models, set the maximum lag to at least one full seasonal period to detect missed seasonal autocorrelation.

- Rejection at any lag indicates missed dynamics; re-examine the model order, add seasonal terms, or consider a different family (state-space, GARCH).
- `forecast::checkresiduals()` shows the Ljung-Box result alongside ACF and time-series plots, integrating numerical and visual diagnostics.
- The Ljung-Box is sensitive to autocorrelation but not to non-stationarity or distributional assumptions; complement with KPSS / ADF and a residual normality plot.

9.18.9 R Packages Used

Base R `Box.test()` for the canonical implementation; `forecast::checkresiduals()` for an integrated diagnostic with time-series and ACF plots; `feasts::ljung_box()` for the modern tidyverts interface; `astsa::sarima()` provides automatic Ljung-Box reporting in its diagnostic output.

9.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.18.11 See also — labs in this chapter

- Time series basics

9.19 Moving Averages

9.19.1 Introduction

A moving average smooths a time series by replacing each observation with an average of nearby values. It is the oldest and simplest tool for highlighting trend by suppressing high-frequency noise; it underpins more sophisticated techniques (X-11 decomposition, Henderson trend filters) and is fundamental to financial chart analysis. The choice between trailing, centred, and weighted moving averages governs the trade-off between smoothness and lag, and choosing the right window size for the application is the principal craft of moving-average smoothing.

9.19.2 Prerequisites

A working understanding of time series, the trade-off between smoothness and lag, and the role of window size in noise suppression.

9.19.3 Theory

Three main variants:

- **Simple moving average**: arithmetic mean of k consecutive observations. Trailing alignment uses the last k values up to time t (one-sided, introduces lag); centred alignment averages $(k-1)/2$ before and after (two-sided, no lag but undefined at the boundaries).
- **Weighted moving average**: assigns weights to each observation in the window — triangular, Hamming, Spencer 15-term (used in classical seasonal adjustment), or any user-defined kernel.
- **Exponential moving average**: geometrically declining weights, $\hat{y}_t = \alpha y_t + (1-\alpha)\hat{y}_{t-1}$. Closely related to simple exponential smoothing.

A centred MA with window equal to the seasonal period eliminates the seasonal component, leaving trend plus residual; this is the foundation of classical seasonal decomposition.

9.19.4 Assumptions

Regular sampling intervals (irregular spacing requires alternative methods); stationarity is not required, but the moving-average filter assumes a slowly-varying trend.

9.19.5 R Implementation

```r
library(zoo); library(TTR)

x <- AirPassengers

# Trailing mean
sm_3 <- rollmean(x, 3, align = "right", fill = NA)

# Centred 12-month MA (removes annual seasonality)
sm_12 <- rollmean(x, 12, align = "center", fill = NA)

# Weighted MA
sm_w <- TTR::WMA(x, n = 12, wts = 1:12)

plot(x); lines(sm_12, col = "red", lwd = 2)
```

9.19.6 Output & Results

`rollmean()` returns the smoothed series with NA at boundaries (trailing alignment leaves NAs at the start; centred alignment leaves NAs at both ends). `TTR::WMA()` produces weighted alternatives. Plotting the centred 12-month MA on top of the raw series reveals the trend that the seasonal cycle hides.

9.19.7 Interpretation

A reporting sentence: "A centred 12-month moving average revealed the secular trend in monthly air-passenger counts, with the smoothed series showing approximately 5 % annual growth and the seasonal cycle largely eliminated by averaging over a full period." Quote the window size and alignment in any report.

9.19.8 Practical Tips

- A centred moving average of window equal to the seasonal period eliminates the seasonal cycle exactly; this is the X-11 first step.
- Trailing moving averages introduce lag proportional to half the window size; use only when real-time estimation is needed and the lag is acceptable.
- For smoother curves with minimal lag, exponential smoothing, LOESS, or spline smoothers are often preferable to wide moving averages.
- Centred MA is undefined at the start and end of the series; for the boundary, use one-sided MA, asymmetric weights, or an end-effects correction.
- Weighted MAs with carefully chosen kernels (Spencer's 15-term, Henderson's filters) are used in official-statistics seasonal adjustment because they preserve cubic polynomial trends while suppressing irregular components.
- Pair MA smoothing with the original series on the same plot; the gap between them is the seasonal-plus-irregular component.

9.19.9 R Packages Used

`zoo::rollmean()` and `rollmeanr()` for simple moving averages with control over alignment; `TTR::SMA()`, `WMA()`, and `EMA()` for technical-analysis moving averages including weighted and exponential; `forecast::ma()` for centred MAs with seasonal-period defaults; `slider::slide_dbl()` for tidyverse-friendly sliding-window computations.

9.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.19.11 See also — labs in this chapter

- Time series basics

9.20 Vector Autoregression (VAR)

9.20.1 Introduction

Vector autoregression (VAR) is the multivariate generalisation of an autoregressive model: each variable in a k-variate time series depends linearly on lags of itself and lags of every other variable in the system. It is the workhorse of empirical macroeconomics, where modelling GDP, inflation, and interest rates jointly produces forecasts and impulse-response analyses that single-equation models cannot. Outside macro, VAR is the natural tool whenever multiple time series are believed to be jointly determined and forecasting all of them is the goal.

9.20.2 Prerequisites

A working understanding of univariate AR models, multivariate data, and the concept of system-wide forecasting and identification.

9.20.3 Theory

A VAR(p) model is

$$\mathbf{y}_t = \mathbf{c} + A_1\mathbf{y}_{t-1} + A_2\mathbf{y}_{t-2} + \cdots + A_p\mathbf{y}_{t-p} + \mathbf{u}_t,$$

where $\mathbf{y}_t$ is a k-vector, A_i are $k \times k$ coefficient matrices, and $\mathbf{u}_t$ are jointly white-noise innovations with covariance Σ_u. The model is estimated equation-by-equation by OLS (the same as multivariate OLS under spherical errors), and lag length p is chosen by AIC, BIC, HQ, or sequential LRT.

The VAR feeds three key downstream analyses: multi-step forecasts with system-coherent uncertainty bands, impulse-response functions that trace the dynamic effect of a shock to one variable through the system, and Granger-causality tests for predictive relationships among the variables.

9.20.4 Assumptions

Stationary series (difference first if not, or use VECM for cointegrated I(1) series); independent, homoscedastic, jointly white-noise innovations; correctly specified lag order.

9.20.5 R Implementation

```r
library(vars)

data(Canada)   # four-variate quarterly macroeconomic series
VARselect(Canada, lag.max = 10, type = "const")$selection
```

```r
fit <- VAR(Canada, p = 2, type = "const")
summary(fit)

# Forecast
fc <- predict(fit, n.ahead = 8)
plot(fc)
```

9.20.6 Output & Results

`VARselect()` returns suggested lag orders by four information criteria; differing recommendations across criteria are common. `summary()` returns equation-by-equation coefficient tables. `predict()` produces multi-step forecasts with system-coherent confidence bands.

9.20.7 Interpretation

A reporting sentence: "VARselect identified $p = 2$ as the AIC- and HQ-optimal lag order for the four-variate Canada series; the VAR(2) model produced eight-quarter forecasts with prediction intervals that widen from ± 0.5 % at one quarter to ± 1.8 % at eight, with the multivariate covariance structure preserving the empirical co-movements between employment, GDP, and inflation." Always state the lag selection criterion and the deterministic specification (constant, trend, or both).

9.20.8 Practical Tips

- Select lag order by AIC, BIC, or HQ information criteria; differing recommendations are common, and the conservative HQ or BIC choice is usually preferable for forecasting.
- Check residual cross-correlations with `serial.test(fit)`; significant off-diagonals indicate missed cross-series dynamics.
- For cointegrated I(1) series, use VECM (`urca::ca.jo()`) on the levels rather than VAR on differences; differencing throws away the long-run equilibrium relationship.
- Structural VARs impose identifying restrictions (Cholesky, sign, long-run) on Σ_u to give impulse responses a causal interpretation; the standard reduced-form VAR does not.
- Forecast-error variance decompositions (`fevd(fit)`) partition forecast uncertainty across innovations; useful for understanding which variables drive forecast errors at each horizon.
- For high-dimensional VARs ($k > 10$), use Bayesian VARs with Minnesota or stochastic-search variable-selection priors; reduced-form OLS becomes unstable.

9.20.9 R Packages Used

`vars::VAR()`, `predict()`, `irf()`, `fevd()`, `causality()` for the canonical VAR workflow; `urca::ca.jo()` for cointegration tests as the precursor to VECM; `tsDyn::lineVar()` for an alternative implementation; `BVAR` for Bayesian VARs with hierarchical priors; `fable::VAR()` for the modern tidyverts interface.

9.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.20.11 See also — labs in this chapter

- Time series basics

9.21 Phillips-Perron Test

9.21.1 Introduction

The Phillips-Perron (PP) test is a unit-root test that modifies the Dickey-Fuller statistic to account for autocorrelation and heteroscedasticity in the errors non-parametrically. Where the augmented Dickey-Fuller test handles serial correlation by including lagged differences in the regression, PP applies a Newey-West long-run variance correction directly to the test statistic. The two approaches usually agree; PP is preferable when the lag specification of ADF is uncertain or when heteroscedasticity is suspected.

9.21.2 Prerequisites

A working understanding of the augmented Dickey-Fuller test, the Newey-West long-run variance estimator, and the unit-root vs stationarity distinction.

9.21.3 Theory

PP fits the unit-root regression $\Delta y_t = \alpha + \gamma y_{t-1} + \varepsilon_t$ (without lagged differences) and computes a test statistic that is non-parametrically corrected for residual autocorrelation and heteroscedasticity using the Newey-West estimator of the long-run variance. Two flavours exist: $Z(\alpha)$ (based on the OLS coefficient) and $Z(t)$ (based on the t-statistic). The asymptotic distribution under the unit-root null is the same Dickey-Fuller distribution as ADF, so critical values are tabulated.

The non-parametric correction sidesteps the ADF lag-order choice but requires a bandwidth choice for the long-run variance estimator; defaults work for most series.

9.21.4 Assumptions

Errors are stationary after the unit-root transformation; the long-run variance bandwidth is appropriate for the autocorrelation horizon.

9.21.5 R Implementation

```
library(tseries)

x <- cumsum(rnorm(100))

pp.test(x)                      # default
pp.test(x, type = "Z(alpha)")
pp.test(diff(x))

# Compare with ADF
adf.test(x)
```

9.21.6 Output & Results

`pp.test()` returns the test statistic and a p-value computed from the Dickey-Fuller distribution. Comparing PP and ADF on the same series is a useful robustness check; large discrepancies signal autocorrelation or heteroscedasticity issues that one test handles better than the other.

9.21.7 Interpretation

A reporting sentence: "The Phillips-Perron test on the random-walk series gave $p = 0.58$, failing to reject the unit-root null; the augmented Dickey-Fuller agreed ($p = 0.52$). After first-differencing both tests rejected the null at $p < 0.01$, confirming the series is integrated of order 1." Always pair PP with ADF and ideally with KPSS for the most robust unit-root verdict.

9.21.8 Practical Tips

- PP and ADF usually agree; if they disagree, examine the residual autocorrelation structure to identify which test is more appropriate.
- PP's non-parametric correction avoids the ADF lag-order sensitivity but introduces a bandwidth choice; the default Newey-West truncation is usually fine.

- Both PP and ADF have low power against near-unit-root stationary processes (e.g., AR(1) with coefficient 0.95); for borderline series, no test gives a confident answer.
- Combine PP with KPSS for the two-sided perspective: PP / ADF test the unit-root null while KPSS tests the stationarity null.
- For series with structural breaks, PP is invalid; use Zivot-Andrews (`urca::ur.za()`) instead.
- For panel data, panel unit-root tests (Im-Pesaran-Shin, Levin-Lin-Chu) extend the framework to multiple series simultaneously.

9.21.9 R Packages Used

`tseries::pp.test()` for the canonical Phillips-Perron implementation; `urca::ur.pp()` for richer output with multiple critical values; `tseries::adf.test()` and `urca::ur.df()` for the complementary ADF; `urca::ur.za()` for Zivot-Andrews when structural breaks are plausible.

9.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.21.11 See also — labs in this chapter

- Time series basics

9.22 Rolling-Origin Cross-Validation

9.22.1 Introduction

Standard k-fold cross-validation shuffles observations across folds, which violates time ordering and leaks future information into the training data. For time series, this invalidates predictive performance estimates entirely. Rolling-origin cross-validation (also called time-series CV or walk-forward validation) preserves temporal ordering by training up to a cutoff and forecasting the subsequent horizon, then sliding the cutoff forward and repeating. It is the only valid CV scheme for time-series forecast evaluation.

9.22.2 Prerequisites

A working understanding of cross-validation, time-series forecasting, and the difference between in-sample fit and out-of-sample predictive performance.

9.22.3 Theory

For a series of length T, choose an initial training-window size t_0, a forecast horizon h, and (optionally) a step size for cutoff advancement. For each cutoff $t = t_0, t_0 + s, \dots, T - h$:

1. Train on $\{y_1, \dots, y_t\}$.
2. Forecast $\hat{y}_{t+1}, \dots, \hat{y}_{t+h}$.
3. Compute the error against observed $y_{t+1}, \dots, y_{t+h}$.

Average errors across cutoffs for each horizon $1, 2, \dots, h$ to obtain horizon-specific accuracy estimates. Two windowing variants exist: expanding window (training set grows) and rolling window (fixed-size training set slides forward). Expanding is more efficient when the underlying process is stable; rolling is more robust to regime changes.

9.22.4 Assumptions

Sufficient history that the initial training window estimates the model adequately; the model can be refit at each cutoff (or parameters updated efficiently for computational savings).

9.22.5 R Implementation

```r
library(forecast)

x <- AirPassengers
h <- 12

# tsCV: rolling-origin with one-step expanding window
f_ets <- function(y, h) forecast(ets(y), h = h)$mean
err_ets <- tsCV(x, f_ets, h = h)

# RMSE by horizon
sqrt(colMeans(err_ets^2, na.rm = TRUE))

# Compare ARIMA
f_arima <- function(y, h) forecast(auto.arima(y), h = h)$mean
err_arima <- tsCV(x, f_arima, h = h)
sqrt(colMeans(err_arima^2, na.rm = TRUE))
```

9.22.6 Output & Results

`tsCV()` returns a matrix of forecast errors with one row per cutoff and one column per horizon. Column-wise summaries give horizon-specific RMSE, MAE, or other metrics. Comparing matrices across models identifies where each performs best.

9.22.7 Interpretation

A reporting sentence: "Rolling-origin CV with one-step expanding window and forecast horizon 12 months showed ARIMA outperforming ETS at short horizons (1–3 months: RMSE 22 vs 28) but ETS was more accurate at longer horizons (10–12 months: RMSE 38 vs 45). The horizon-specific differences justify reporting CV-RMSE per horizon rather than a single averaged metric." Report horizon-specific errors; collapsing them hides the structure that informs model choice.

9.22.8 Practical Tips

- `forecast::tsCV()` implements expanding-window CV with one-step cutoff advancement; for rolling fixed-window, write a small loop.
- For computational cost, increase the initial window and skip cutoffs (advance by more than one observation per step); this trades estimation precision for speed.
- Always report horizon-specific errors; a single averaged RMSE hides the structure that determines which model wins at which horizon.
- Refitting the model at every cutoff is expensive; for ARIMA, fixed-parameter updates (`forecast(Arima(y,model=fit),h=h)`) are faster but ignore changing dynamics.
- `fable::stretch_tsibble()` provides tidy rolling-CV splits that integrate with the modern tidyverts forecasting workflow.
- For nested or hierarchical forecasts, run rolling-origin CV at the bottom level and aggregate; never CV at the aggregated level alone.

9.22.9 R Packages Used

`forecast::tsCV()` for expanding-window time-series CV; `fable::stretch_tsibble()` and `accuracy()` for tidyverts CV workflows; `tibbletime::rollify()` for general rolling functions; `slider::slide_dbl()` for sliding-window operations.

9.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.22.11 See also — labs in this chapter

- Time series basics

9.23 Seasonal ARIMA (SARIMA)

9.23.1 Introduction

Seasonal ARIMA (SARIMA) extends the standard ARIMA framework to series with periodic seasonality. The standard notation is ARIMA$(p, d, q)(P, D, Q)_s$ where the lower-case letters are non-seasonal orders, the upper-case are seasonal orders, and s is the seasonal period (12 for monthly with annual seasonality, 4 for quarterly, 24 for hourly with daily seasonality). SARIMA is the workhorse parametric forecasting tool for economic, financial, and many biomedical time series with stable seasonal patterns.

9.23.2 Prerequisites

A working understanding of ARIMA models, differencing, the backshift operator, and seasonal decomposition.

9.23.3 Theory

The full SARIMA equation is

$$\Phi_P(B^s)\phi_p(B)(1-B)^d(1-B^s)^D y_t = \Theta_Q(B^s)\theta_q(B)\varepsilon_t,$$

where B is the backshift operator ($By_t = y_{t-1}$). The non-seasonal AR, differencing, and MA components compose multiplicatively with their seasonal counterparts at lag s. The model accommodates trends through differencing (d) and seasonal patterns through both seasonal differencing (D) and seasonal AR/MA terms.

The famous "airline model" $(0, 1, 1)(0, 1, 1)_{12}$ for monthly airline passengers — Box and Jenkins's headline example — demonstrates that even highly trending and strongly seasonal series can fit into the SARIMA family with modest order.

9.23.4 Assumptions

The series is stationary after regular and seasonal differencing; residuals are white noise; the seasonal period s is correctly specified by the `frequency()` of the input `ts` object.

9.23.5 R Implementation

```
library(forecast)

x <- AirPassengers

fit <- auto.arima(log(x), seasonal = TRUE)
```

```r
summary(fit)

fc <- forecast(fit, h = 24)
autoplot(fc)

checkresiduals(fit)
```

9.23.6 Output & Results

`auto.arima()` searches over candidate orders using AICc and returns the chosen SARIMA model with its coefficients and standard errors. `forecast()` produces point forecasts and 80 % / 95 % prediction intervals. `checkresiduals()` reports the Ljung-Box test, ACF, and time-series plot of residuals.

9.23.7 Interpretation

A reporting sentence: "`auto.arima` on log-transformed AirPassengers selected an ARIMA$(0,1,1)(0,1,1)_{12}$ structure (AICc $= -483$, $\sigma^2 = 0.0014$); residuals passed Ljung-Box at lag 24 ($p = 0.61$). Forecasts for the next 24 months show continued multiplicative growth and stable seasonal pattern, with 95 % prediction intervals widening from $\pm 5\%$ at 1 month to $\pm 18\%$ at 24 months." Always check residual diagnostics before reporting forecasts.

9.23.8 Practical Tips

- Use `auto.arima()` for automatic order selection; double-check the result against ACF/PACF inspection of the differenced series.
- Log-transform for multiplicative seasonality (where seasonal amplitude grows with the level); SARIMA on the log scale is then additive and well-behaved.
- For very long seasonal periods ($s \geq 52$), SARIMA becomes computationally heavy; consider regression with Fourier seasonal terms (`forecast::tslm` with `fourier()`) instead.
- Always use chronological train/test splits; random CV is invalid for time series because it leaks future information into past folds.
- Box-Jenkins residual diagnostics (Ljung-Box, ACF, PACF, normality plot) are essential after fitting; `checkresiduals()` aggregates them.
- For multiple seasonalities (weekly + yearly), use TBATS (`forecast::tbats`) or `mstl()` decomposition with non-seasonal forecasting on the adjusted series.

9.23.9 R Packages Used

`forecast::auto.arima()` and `Arima()` for canonical SARIMA fitting and forecasting; `fable::ARIMA()` for the modern tidyverts interface; `feasts::ACF()` and

`PACF()` for diagnostic visualisations; `seasonal::seas()` for X-13ARIMA-SEATS as the official-statistics alternative.

9.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.23.11 See also — labs in this chapter

- Time series basics

9.24 Seasonal Decomposition (STL)

9.24.1 Introduction

STL (Seasonal and Trend decomposition using Loess) is the most flexible classical decomposition method for time series. It separates a series into trend, seasonal, and remainder components using iterative locally-weighted regression (loess) smoothers. Three properties distinguish STL from alternatives: it handles any kind of seasonality (not just monthly or quarterly), it allows the seasonal pattern to change over time, and it is robust to outliers when run with the robust option enabled.

9.24.2 Prerequisites

A working understanding of time-series decomposition, the additive vs multiplicative model distinction, and basic familiarity with loess smoothers.

9.24.3 Theory

STL alternates two loess smoothing operations until convergence: a seasonal smoother that operates within each cycle position (all Januaries together, all Februaries together, etc.) and a trend smoother that operates across the deseasonalised series. Two window parameters control the resulting decomposition:

- `s.window`: seasonal-window length. `"periodic"` forces a constant seasonal pattern across the series; an integer (typical 7, 9, 11) allows the seasonal pattern to change gradually.
- `t.window`: trend-window length. Larger values produce smoother trend estimates.

For multiplicative seasonality (where seasonal amplitude grows with the level), apply STL to the log of the series; the result is interpretable as an additive decomposition on the log scale.

9.24.4 Assumptions

Additive components on the chosen scale; the seasonal cycle is correctly specified (`frequency()` of the `ts` object); sufficient series length to estimate the decomposition (at least two full seasonal cycles).

9.24.5 R Implementation

```
library(forecast)

# log-transform for multiplicative data
x <- log(AirPassengers)

# STL with fixed seasonal
stl_fit <- stl(x, s.window = "periodic")
autoplot(stl_fit)

# Seasonal adjustment
seasadj <- seasadj(stl_fit)
plot(seasadj)

# Forecast via STL + ETS
fc <- forecast(stl_fit, method = "ets", h = 24)
autoplot(fc)
```

9.24.6 Output & Results

`stl()` returns an `stl` object with the decomposition. `autoplot()` produces a four-panel display: original series, trend, seasonal, and remainder. `seasadj()` extracts the seasonally adjusted series (original minus seasonal); `forecast(stl_fit)` applies a chosen method (ETS, ARIMA) to the seasonally adjusted component and adds back the seasonal pattern.

9.24.7 Interpretation

A reporting sentence: "STL decomposition of log-AirPassengers showed a smooth upward trend (annual growth approximately 5 %), stable multiplicative seasonality (constant on the log scale, multiplicative on the original), and a remainder series consistent with white noise (Ljung-Box $p = 0.32$); the seasonal pattern peaked in July–August, consistent with summer holiday travel." Always describe trend and seasonality separately and check that the remainder is approximately white noise.

9.24.8 Practical Tips

- `s.window="periodic"` forces a constant seasonal pattern across the series; integer values (7, 11) allow the seasonal pattern to evolve. Use the integer when seasonality has visibly shifted over the series.
- Set `robust=TRUE` for outlier-resistant fitting; the standard STL is sensitive to single extreme observations.
- For forecasting, `stlf()` is a one-call wrapper that applies STL + an automatic ETS or ARIMA forecast on the seasonally adjusted series.
- STL handles any seasonal period; for multiple seasonalities (e.g. weekly and yearly), use `forecast::mstl()` which decomposes into multiple seasonal layers.
- Residuals (remainder) should look like white noise; systematic structure indicates the trend or seasonal windows need tuning.
- Compare STL to classical seasonal decomposition (`decompose`) and X-13 (`seasonal::seas`); STL is generally more flexible but X-13 is the regulatory standard for official statistics.

9.24.9 R Packages Used

Base R `stl()` for the canonical implementation; `forecast::stlf()` for STL-based forecasting; `forecast::mstl()` for multi-seasonal decomposition; `seasonal::seas()` for X-13 ARIMA-SEATS as the official-statistics alternative; `feasts::STL()` for the modern tidyverts interface.

9.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.24.11 See also — labs in this chapter

- Time series basics

9.25 Spectral Analysis

9.25.1 Introduction

Spectral analysis examines a time series in the frequency domain rather than the time domain. Periodicities appear as peaks in the power spectrum, slow trends dominate the low-frequency end, and white noise has a flat spectrum. The view is complementary to autocorrelation-based time-domain analysis: the spectral density and the autocovariance function are Fourier-transform pairs,

so neither view contains information the other lacks, but each makes different features easier to see.

9.25.2 Prerequisites

A working understanding of Fourier analysis, the autocovariance function, and the time-frequency duality of stationary processes.

9.25.3 Theory

For a stationary process with autocovariance function γ_k, the spectral density is

$$S(f) = \sum_{k=-\infty}^{\infty} \gamma_k e^{-2\pi i f k},$$

defined on $f \in [-1/2, 1/2]$ for unit-spaced data. The periodogram is the sample-based estimator, computed as the squared magnitude of the FFT of the centred series divided by the series length. Raw periodograms are inconsistent (variance does not decrease with n), so smoothing — Bartlett, Welch, kernel-based, or multi-taper — is applied to produce stable spectral estimates.

A peak at frequency f indicates a cyclic component with period $1/f$ time units; the area under a frequency band is the variance contribution from that band.

9.25.4 Assumptions

The series is approximately stationary (or has been made so by detrending and differencing); sufficient length for frequency resolution $1/n$ to distinguish nearby peaks.

9.25.5 R Implementation

```r
# Simulated series with two cycles
set.seed(2026)
t <- 1:500
y <- sin(2 * pi * t / 12) + 0.5 * sin(2 * pi * t / 4) + rnorm(500, 0, 0.3)

# Periodogram
spec_p <- spectrum(y, plot = FALSE)
plot(spec_p, main = "Raw periodogram")

# Smoothed
spec_s <- spectrum(y, spans = c(5, 5), plot = FALSE)
plot(spec_s, main = "Smoothed periodogram")
abline(v = 1/12, col = "red", lty = 2)
```

```r
abline(v = 1/4,  col = "red", lty = 2)
```

9.25.6 Output & Results

`spectrum()` returns the periodogram with frequencies and spectral-density estimates, plus the chosen smoothing-window structure. The plot shows distinct peaks at $f = 1/12$ and $f = 1/4$, matching the simulated cycles.

9.25.7 Interpretation

A reporting sentence: "Spectral analysis of the simulated series with two embedded cycles produced peaks at frequencies $f = 0.083$ (period 12) and $f = 0.25$ (period 4) on the smoothed periodogram with spans = c(5, 5); the relative peak heights match the relative amplitudes of the simulated components." Always state the smoothing window in the figure caption; raw periodograms are noisy and often misread.

9.25.8 Practical Tips

- Smoothing (`spans=c(5,5)` or similar) trades variance for bias; tune to balance peak detection against resolution.
- Frequency resolution is $1/n$; for closely spaced periodicities, longer series are essential.
- Use a log-log plot of frequency vs spectral density to expose power-law behaviour and broadband features.
- Multi-taper estimation (`multitaper::spec.mtm`) provides better bias-variance trade-off than spans-based smoothing for short series.
- Spectral analysis complements ACF / PACF; use both, especially when looking for cyclic versus trend structure.
- For non-stationary series, time-frequency methods (short-time Fourier, wavelets) generalise the stationary spectral framework.

9.25.9 R Packages Used

Base R `spectrum()` for periodograms with smoothing; `multitaper::spec.mtm()` for multi-taper estimation; `astsa::mvspec()` for multivariate spectral analysis; `WaveletComp` for time-frequency localisation when stationarity fails.

9.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.25.11 See also — labs in this chapter

- Time series basics

9.26 State-Space Models

9.26.1 Introduction

State-space models describe a time series via a latent state vector that evolves over time, with observations as a noisy function of the state. This general framework encompasses essentially every classical time-series model — ARIMA, exponential smoothing (ETS), structural time series, hidden Markov models, dynamic linear models — as special cases. The unified framework provides a consistent inference machinery (the Kalman filter for linear Gaussian models; particle and extended-Kalman filters for non-linear or non-Gaussian extensions) and a flexible specification language for custom models.

9.26.2 Prerequisites

A working understanding of linear algebra, Bayesian or likelihood inference, and basic familiarity with classical ARIMA / ETS forecasting.

9.26.3 Theory

The standard linear Gaussian state-space model has two equations:

Observation equation: $y_t = F_t x_t + v_t, \quad v_t \sim \mathcal{N}(0, V_t)$.

State transition: $x_t = G_t x_{t-1} + w_t, \quad w_t \sim \mathcal{N}(0, W_t)$.

The matrices (F_t, G_t, V_t, W_t) specify the model — they are typically constant in time but can vary deterministically. Inference proceeds in three modes: **filtering** estimates $x_t \mid y_{1:t}$ in real time; **smoothing** estimates $x_t \mid y_{1:T}$ using the full series; **prediction** extends the state forward beyond the data.

The Kalman filter computes both filtered and smoothed states by recursive Gaussian updates; non-linear or non-Gaussian extensions use the extended or unscented Kalman filter, the particle filter, or fully Bayesian MCMC.

9.26.4 Assumptions

For the standard Kalman filter: linear Gaussian observation and state equations. Non-linearity or non-Gaussianity requires extended methods.

9.26.5 R Implementation

```
library(dlm)
```

```r
# Local level model (random walk + noise)
build_dlm <- function(theta)
  dlmModPoly(order = 1, dV = exp(theta[1]), dW = exp(theta[2]))

set.seed(2026)
y <- Nile
fit <- dlmMLE(y, parm = c(0, 0), build = build_dlm)
dlm_mod <- build_dlm(fit$par)

# Filtering
filt <- dlmFilter(y, dlm_mod)
smo  <- dlmSmooth(filt)

plot(y)
lines(dropFirst(smo$s), col = "red", lwd = 2)
```

9.26.6 Output & Results

`dlmMLE()` estimates the variance components by maximum likelihood. `dlmFilter()` and `dlmSmooth()` produce filtered and smoothed state estimates. The smoothed latent series captures the gradual level evolution; for the Nile, it shows the famous step-down around 1900 corresponding to the Aswan Low Dam construction.

9.26.7 Interpretation

A reporting sentence: "A local-level state-space model of Nile river flow fit by maximum likelihood estimated observation variance 15{,}100 and state-transition variance 1{,}470 (signal-to-noise ratio 0.10), favouring gradual state evolution; the smoothed latent series captured a clear level shift in the late 1890s." Always state the model structure (local level, local linear trend, etc.) and the estimated variance components.

9.26.8 Practical Tips

- ARIMA and ETS have exact state-space representations; `forecast::Arima()` and `ets()` use them internally.
- Unknown latent states are estimated automatically by the Kalman filter; the user typically only specifies the model structure and estimates variance components by MLE.
- For non-linear or non-Gaussian models, use extended Kalman, unscented Kalman, or particle filters; `bayesforecast` and `bsts` provide Bayesian alternatives.
- Parameter estimation by MLE (`dlmMLE`) or Bayesian MCMC (`bsts`); for Bayesian setups, weakly informative priors on variance components sta-

bilise inference when data are weak.
- State-space is the general framework; choose a specific model (local level, local linear trend, structural with seasonal, dynamic regression) based on substantive domain knowledge.
- For diagnostics, examine the standardised innovations (filter residuals); they should be approximately white noise under correct model specification.

9.26.9 R Packages Used

`dlm` for the canonical dynamic linear model framework; `KFAS` for fast Kalman filtering / smoothing including non-Gaussian extensions; `bsts` for Bayesian structural time series; `MARSS` for multivariate ARMA state-space models; `fable::SSM()` for tidyverts state-space modelling.

9.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.26.11 See also — labs in this chapter

- Time series basics

9.27 Stationarity Tests

9.27.1 Introduction

Stationarity — constant mean, variance, and autocovariance over time — is a fundamental prerequisite for most classical time-series models (ARIMA, VAR, spectral analysis). Non-stationary series typically need to be made stationary by differencing, detrending, or transformation before model fitting; forgetting this step produces nonsense estimates and invalid forecasts. Formal stationarity testing combines several tests with complementary nulls to give a robust verdict.

9.27.2 Prerequisites

A working understanding of time-series basics, autocorrelation, the difference between strict and weak stationarity, and the role of differencing in inducing stationarity.

9.27.3 Theory

The standard test battery comprises:

- **Augmented Dickey-Fuller (ADF)**: null = unit root (non-stationary), alternative = stationary or trend-stationary. Rejection supports stationarity.
- **KPSS**: null = stationarity (level or trend), alternative = unit root. Rejection supports non-stationarity.
- **Phillips-Perron (PP)**: non-parametric variant of ADF, robust to autocorrelation and heteroscedasticity in errors.

Pair them: stationary if ADF rejects *and* KPSS does not; non-stationary if ADF fails to reject *and* KPSS rejects. Disagreement (both reject, neither rejects) is ambiguous and may indicate near-unit-root behaviour, structural breaks, or fractional integration.

For series with structural breaks, standard tests are invalid; use Zivot-Andrews (`urca::ur.za()`) which extends the framework to allow one endogenous break.

9.27.4 Assumptions

Choice of test depends on the alternative — level vs trend stationarity for KPSS, drift vs trend specification for ADF. Adequate sample size for asymptotic critical values.

9.27.5 R Implementation

```r
library(tseries); library(urca)

x <- AirPassengers

# ADF
adf.test(log(x))

# KPSS
kpss.test(log(x))

# Phillips-Perron
pp.test(log(x))

# Differencing to achieve stationarity
ndiffs(log(x))   # forecast::ndiffs
```

9.27.6 Output & Results

Each test returns a statistic and p-value. `forecast::ndiffs()` and `nsdiffs()` automate the differencing decision by running KPSS or other tests at each candidate differencing order until stationarity is supported.

9.27.7 Interpretation

A reporting sentence: "On the log-transformed AirPassengers series, ADF failed to reject the unit-root null ($p = 0.99$) and KPSS rejected level stationarity ($p < 0.05$); both agree the series is non-stationary. After first-differencing plus seasonal differencing (order $(0,1,0)(0,1,0)_{12}$), all three tests supported stationarity, justifying ARIMA modelling on the doubly-differenced series." Always pair ADF with KPSS and report both verdicts.

9.27.8 Practical Tips

- Run ADF and KPSS together; their conclusions should agree, and disagreement is itself diagnostic information.
- For series with a clear trend, test trend-stationarity explicitly (`kpss.test(x,null="Trend")` and `adf.test` with trend specification).
- Structural breaks invalidate standard tests; use Zivot-Andrews (`urca::ur.za()`) when a break is plausible.
- `forecast::ndiffs()` and `nsdiffs()` automate the choice of regular and seasonal differencing orders for ARIMA model preparation.
- Visual inspection of the series — plotting it and looking for trends, level shifts, or changing variance — usually suggests the right transformation before any test is run.
- For panel data (multiple time series), use Im-Pesaran-Shin or Levin-Lin-Chu panel unit-root tests; running univariate tests on each series ignores the panel structure.

9.27.9 R Packages Used

`tseries::adf.test()`, `kpss.test()`, `pp.test()` for the standard battery; `urca::ur.df()`, `ur.kpss()`, `ur.pp()`, `ur.za()` for richer output and structural-break extensions; `forecast::ndiffs()` and `nsdiffs()` for automated differencing decisions; `feasts::unitroot_kpss()` and `unitroot_ndiffs()` for the modern tidyverts interface.

9.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

9.27.11 See also — labs in this chapter

- Time series basics

9.28 Time Series: Introduction

9.28.1 Introduction

A time series is a sequence of observations indexed by time. Time-series analysis decomposes the series into four interpretable components — trend, seasonality, cyclicality, and noise — and models each. The decomposition framework underpins virtually every classical method: ARIMA models the noise after differencing out trend and seasonality; ETS captures all four components in a state-space form; Bayesian structural time series isolates them as latent processes. Understanding the four-component vocabulary is the foundational step before reaching for any specific model.

9.28.2 Prerequisites

A working understanding of basic statistics, the `ts` class in R, and the difference between cross-sectional and longitudinal data structures.

9.28.3 Theory

The classical additive decomposition is

$$y_t = T_t + S_t + C_t + \varepsilon_t,$$

with components: trend T (long-term level change), seasonal S (regular fixed-period variation, e.g. monthly cycle), cycle C (irregular longer-than-seasonal swings, e.g. business cycles), and noise ε (residual stochastic variation). The multiplicative form $y_t = T_t \cdot S_t \cdot C_t \cdot \varepsilon_t$ applies when seasonal amplitude grows with the level; it is additive on the log scale.

In practice, the cycle is often combined with the trend (giving "trend-cycle" as one component), leaving three components in most decomposition implementations: trend-cycle, seasonal, and remainder.

9.28.4 Assumptions

Time order is preserved (no shuffling); regular sampling intervals (irregular sampling requires `zoo` / `xts` or modelling the spacing); the chosen decomposition (additive vs multiplicative) matches the data.

9.28.5 R Implementation

```r
library(forecast)

data(AirPassengers)
plot(AirPassengers, main = "Monthly airline passengers")

# Classical decomposition
dec <- decompose(AirPassengers, type = "multiplicative")
plot(dec)

# STL: loess-based decomposition
stl_dec <- stl(log(AirPassengers), s.window = "periodic")
plot(stl_dec)
```

9.28.6 Output & Results

`decompose()` returns the classical additive or multiplicative decomposition; `stl()` returns the loess-based seasonal-trend decomposition. The four-panel plot shows observed, trend, seasonal, and remainder components stacked vertically.

9.28.7 Interpretation

A reporting sentence: "The monthly air-passenger series exhibited a clear upward trend (approximately 5 % per year), a stable annual seasonal cycle peaking in summer, and noise that grew with the level (motivating the log transformation). STL decomposition on the log scale produced trend, seasonal, and approximately white-noise remainder components." When introducing a time series, describe each component visible in the data.

9.28.8 Practical Tips

- Log-transform series whose variance grows with the level; the resulting decomposition on the log scale is additive and well-behaved.
- STL (`stl()`) is more robust than classical `decompose()`; the latter is sensitive to outliers and assumes fixed seasonal patterns.
- After decomposition, examine the remainder series with ACF / PACF plots and Ljung-Box; substantial autocorrelation indicates missed structure.
- The modern tidy workflow uses `tsibble` for time-aware tibbles, `feasts` for decomposition and diagnostics, and `fable` for forecasting; together they replace much of the older `forecast` workflow.
- For irregular sampling intervals, use `zoo` or `xts` time series; classical decomposition assumes regular sampling.
- For multiple seasonalities (e.g. weekly + yearly in retail data), `forecast::mstl()` produces a multi-layer decomposition that single-seasonal STL

cannot.

9.28.9 R Packages Used

Base R `ts()`, `decompose()`, and `stl()`; `forecast` for the broader classical workflow including `auto.arima`, `forecast`, `mstl`; `tsibble`, `feasts`, `fable` for the modern tidyverts ecosystem; `zoo` and `xts` for irregularly-spaced time series.

9.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.28.11 See also — labs in this chapter

- Time series basics

9.29 VECM and Cointegration

9.29.1 Introduction

Two or more non-stationary time series are *cointegrated* if some linear combination of them is stationary. The classic example is interest rates at different maturities: each rate is non-stationary (random-walk-like), but spreads are mean-reverting. A vector error-correction model (VECM) captures the short-run dynamics of cointegrated series while respecting the long-run equilibrium relationship; the Johansen procedure tests how many cointegrating relations exist and estimates them jointly with the dynamic parameters.

9.29.2 Prerequisites

A working understanding of vector autoregressive (VAR) models, unit roots, and the I(1)/I(0) terminology for integrated and stationary series.

9.29.3 Theory

The VECM is

$$\Delta y_t = \Pi y_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \Delta y_{t-i} + \varepsilon_t.$$

The matrix Π encodes the long-run relationships; its rank equals the number of cointegrating relations. Johansen's approach tests the rank via two statistics:

- **Trace test**: $\lambda_{\text{trace}}(r) = -T \sum_{i=r+1}^{k} \log(1 - \hat{\lambda}_i)$ tests $H_0 : \text{rank} \leq r$.
- **Maximum eigenvalue test**: $\lambda_{\max}(r) = -T \log(1 - \hat{\lambda}_{r+1})$ tests $H_0 : \text{rank} = r$ vs $H_1 : \text{rank} = r + 1$.

The two often agree; when they disagree, the trace test is generally preferred. Once the rank is determined, the cointegrating vectors and adjustment speeds can be estimated.

9.29.4 Assumptions

Each series is individually I(1) (verifiable by ADF and KPSS); k-variate Gaussian innovations; the lag length p is correctly specified.

9.29.5 R Implementation

```r
library(urca); library(vars)

data(Canada)
jo <- ca.jo(Canada, ecdet = "const", type = "eigen", K = 2, spec =
 ↪  "transitory")
summary(jo)

# Fit VECM with rank r = 1 (if supported by test)
vecm <- cajorls(jo, r = 1)
vecm
```

9.29.6 Output & Results

`ca.jo()` returns the eigenvalue and trace test statistics per candidate rank, along with critical values at 10/5/1 % levels. `cajorls()` fits the VECM at the chosen rank and returns the cointegrating vector and adjustment coefficients.

9.29.7 Interpretation

A reporting sentence: "Johansen's trace test supported one cointegrating relationship in the Canada macroeconomic data (test statistic 32.8 vs 5 % critical value 25.3, $p < 0.05$); the corresponding cointegrating vector linked money demand and interest rates with adjustment speeds of -0.10 for money demand toward equilibrium." Always report both trace and max-eigenvalue results, and the chosen lag length.

9.29.8 Practical Tips

- Confirm each series is I(1) via ADF / KPSS before cointegration testing; cointegration is meaningful only between integrated series.
- The `ecdet` argument specifies whether deterministic terms (constant, trend) enter inside the cointegrating relation; choose based on visible long-run drift.
- Engle-Granger two-step is simpler for two series and gives essentially equivalent conclusions; Johansen generalises to k variables.
- Report both trace and max-eigenvalue tests; they typically agree but can disagree when one or two cointegrating relations are weak.
- For short samples (under 100 observations) and many variables, Johansen loses power; reduce the dimension or use bias-corrected critical values.
- Once the VECM is fitted, impulse-response functions and forecast-error variance decompositions on `vec2var(vecm)` extend the standard VAR diagnostic toolkit.

9.29.9 R Packages Used

`urca` for `ca.jo()`, `cajorls()`, and Johansen-related cointegration tests; `vars` for VAR modelling and `vec2var()` to convert a VECM back to VAR form for IRF analysis; `tsDyn::VECM()` for an alternative VECM implementation; `urca::ca.po()` for the Phillips-Ouliaris cointegration tests.

9.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.29.11 See also — labs in this chapter

- Time series basics

9.30 Wavelet Analysis

9.30.1 Introduction

Wavelet analysis decomposes a time series into components localised in both time and frequency simultaneously. Unlike the Fourier transform, which assumes stationarity and produces a single spectrum for the entire series, wavelets capture cyclic components that appear, shift, or disappear over time — essential for non-stationary signals such as EEG recordings, climate proxies, or any series whose periodic content evolves. The output is a scalogram, a 2D map of power

across time and frequency that reveals the time-varying structure that Fourier methods miss.

9.30.2 Prerequisites

A working understanding of spectral analysis and Fourier transforms, the concept of time-frequency localisation, and basic familiarity with the trade-off between time and frequency resolution.

9.30.3 Theory

The continuous wavelet transform (CWT) convolves the signal with scaled and shifted copies of a mother wavelet ψ:

$$W(a,b) = \int y(t)\,\bar{\psi}\!\left(\frac{t-b}{a}\right)\frac{\mathrm{d}t}{\sqrt{|a|}}.$$

The scale a controls the wavelet's "width" and corresponds inversely to frequency; b is the time shift. The squared magnitude $|W(a,b)|^2$ is the wavelet power at time b and scale a. Common mother wavelets include Morlet (oscillatory, good time-frequency localisation), Haar (step-function), Daubechies (orthogonal, used in compression). The scalogram visualises $|W|^2$ as a heatmap over the time-frequency plane.

The cone of influence delimits regions of the scalogram free of edge artefacts; power within the cone is reliable, power outside should be discounted.

9.30.4 Assumptions

Depends on the wavelet choice; minimal beyond a sufficiently long series. The discrete wavelet transform (DWT) requires series length to be a power of 2; the CWT does not.

9.30.5 R Implementation

```
library(WaveletComp)

set.seed(2026)
t <- 1:500
y <- sin(2 * pi * t / 50) * (t < 250) + sin(2 * pi * t / 20) * (t >= 250) +
     rnorm(500, 0, 0.3)

df <- data.frame(date = t, x = y)
wt <- analyze.wavelet(df, "x", loess.span = 0, dt = 1,
                      dj = 1/50, lowerPeriod = 8,
                    upperPeriod = 128, make.pval = FALSE, verbose = FALSE)
```

```
wt.image(wt, main = "Wavelet power spectrum")
```

9.30.6 Output & Results

`analyze.wavelet()` returns a list containing the wavelet power, phase, and statistical-significance information. `wt.image()` plots the scalogram as a 2D heatmap with the cone of influence overlaid.

9.30.7 Interpretation

A reporting sentence: "The wavelet power spectrum revealed a 50-time-step cycle in the first half of the series and a 20-time-step cycle in the second half — a non-stationary feature invisible to a global Fourier analysis. The cone of influence excluded the first and last 30 time points from interpretation due to edge effects." Always describe the time-frequency structure visible and acknowledge the cone of influence.

9.30.8 Practical Tips

- Choose the wavelet to suit the signal: Morlet for oscillatory components, Mexican-hat for localised peaks, Daubechies for compression and denoising.
- Power outside the cone of influence is unreliable; mask it or interpret with caution.
- Statistical significance via red-noise surrogates (`make.pval=TRUE` in `WaveletComp`) tests power against an AR(1) null at each scale and time.
- The discrete wavelet transform (DWT, in `wavethresh`) supports compression and denoising; CWT is for visualisation and analysis.
- For EEG, fMRI, and other neurophysiological data, wavelets are the standard time-frequency tool; specialised packages (`eegkit`, `fmri.wavelets`) extend the framework.
- For multi-channel time-frequency analysis, cross-wavelet coherence and wavelet phase quantify time-varying co-movements between series.

9.30.9 R Packages Used

`WaveletComp` for `analyze.wavelet()`, `wt.image()`, and biwavelet coherence; `wavelets` and `wavethresh` for discrete wavelet transforms and threshold-based denoising; `dplR` for tree-ring-style wavelet analysis; `Rwave` for additional wavelet families and inversion.

9.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.30.11 See also — labs in this chapter

- Time series basics

9.31 White Noise Tests

9.31.1 Introduction

After fitting a time-series model — ARIMA, ETS, regression with ARIMA errors — the residuals should look like white noise: zero mean, constant variance, and no autocorrelation. White-noise tests check these properties formally and are the standard residual-diagnostic battery for time-series modelling. Significant rejection indicates the model has missed structure that should motivate refinement; failure to reject supports model adequacy.

9.31.2 Prerequisites

A working understanding of ARIMA models, residual diagnostics, and the autocorrelation function (ACF) as the visual counterpart to formal autocorrelation tests.

9.31.3 Theory

Three test families dominate practice:

- **Portmanteau tests** (Box-Pierce, Ljung-Box) aggregate squared sample autocorrelations over the first h lags into a single chi-squared statistic. Under the white-noise null, $Q \sim \chi^2_{h-p-q}$ where $p+q$ is the number of fitted ARMA parameters. The Ljung-Box variant has a small-sample correction and is preferred over Box-Pierce.
- **Turning-points test**: counts the number of local maxima and minima in the residual series; under iid, has known mean $2(n-2)/3$ and variance, giving an approximate Normal test for independence.
- **Runs test**: counts runs of consecutive same-sign residuals; tests independence without distributional assumptions.

A failure on Ljung-Box at any lag indicates missed autocorrelation that the model should address; a failure on the runs or turning-points test indicates more general non-randomness.

9.31.4 Assumptions

Null: residuals are iid white noise. The tests have different power against autocorrelation, heteroscedasticity, and non-linearity; no single test detects all departures, so multiple tests are run as a battery.

9.31.5 R Implementation

```r
library(forecast)

set.seed(2026)
fit <- arima(AirPassengers, order = c(0, 1, 1), seasonal = c(0, 1, 1))
resid_fit <- residuals(fit)

# Ljung-Box
Box.test(resid_fit, lag = 24, type = "Ljung-Box", fitdf = 2)

# Visual + ARIMA diagnostic
checkresiduals(fit)
```

9.31.6 Output & Results

`Box.test()` returns the test statistic and p-value; `checkresiduals()` aggregates the Ljung-Box test with ACF, histogram, and time-series plots into a single diagnostic display.

9.31.7 Interpretation

A reporting sentence: "Ljung-Box at lag 24 on the residuals from the ARIMA(0,1,1)(0,1,1)

fit gave $Q = 28.1$ on 22 degrees of freedom ($p = 0.17$); the ACF showed no isolated significant lags, the histogram was approximately Normal, and the residual time-series plot showed no remaining structure. Residuals are consistent with white noise." Always pair the numerical test with the ACF plot.

9.31.8 Practical Tips

- Choose the maximum lag h around $\log n$ to a small multiple thereof; for seasonal models, set h at least one full seasonal period to detect missed seasonal autocorrelation.
- Use `fitdf=p+q` for ARMA(p, q) fitted models; the test ignores the parameter cost otherwise and is anti-conservative.
- A significant Ljung-Box after model fitting indicates missed structure; add AR or MA terms, seasonal terms, or consider a different model family.

- Pair the formal Ljung-Box test with a visual ACF inspection; isolated significant lags can be substantively meaningful even if the omnibus test passes.
- `forecast::checkresiduals()` is the convenient one-call diagnostic battery; use it routinely after every time-series fit.
- For non-Normal residuals, a Jarque-Bera or Shapiro-Wilk test on the residuals complements the ACF-based tests.

9.31.9 R Packages Used

Base R `Box.test()` for Ljung-Box and Box-Pierce; `forecast::checkresiduals()` for the integrated diagnostic; `tseries::runs.test()` for the runs test; `randtests::turning.point.test()` for the turning-points test; `feasts::ljung_box()` for the modern tidyverts implementation.

9.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.31.11 See also — labs in this chapter

- Time series basics

9.32 X-13ARIMA-SEATS Decomposition

9.32.1 Introduction

X-13ARIMA-SEATS is the seasonal-adjustment workhorse of official statistics. It is the Census Bureau standard, used by Eurostat, the Bank of England, and most national statistical agencies for producing officially published seasonally adjusted economic time series. The combination of RegARIMA-based pre-adjustment (handling outliers, trading-day effects, Easter effects) with either X-11 filter-based or SEATS model-based decomposition makes it the most sophisticated decomposition tool routinely available — and the standard against which simpler methods (STL, classical decomposition) are benchmarked.

9.32.2 Prerequisites

A working understanding of seasonal decomposition, SARIMA models, and the role of calendar effects (trading days, leap years, moving holidays) in economic time series.

9.32.3 Theory

X-13 has two main computational stages:

1. **RegARIMA pre-adjustment**: fit a regression-with-ARIMA-errors model that absorbs outliers (additive, level shift, temporary change), trading-day effects, leap-year effects, and Easter effects. The pre-adjustment removes these from the original series.

2. **Decomposition**: either X-11-style iterative moving averages or SEATS (Signal Extraction in ARIMA Time Series), a model-based decomposition derived from the RegARIMA spectral structure. SEATS is preferred for series with stable seasonal patterns; X-11 is more robust to deviations from the fitted ARIMA.

The output includes the seasonally adjusted series, the trend-cycle, the seasonal component, and the irregular component, plus diagnostics for stability, residual autocorrelation, and seasonal-adjustment quality.

9.32.4 Assumptions

Monthly or quarterly data with several years of history (typically at least 3–5 years for stable seasonal-factor estimation); the RegARIMA pre-adjustment captures the calendar effects that matter for the series.

9.32.5 R Implementation

```r
library(seasonal)

x <- AirPassengers
m <- seas(x)
summary(m)

plot(m)
series(m, "s10")   # seasonal factors
series(m, "s12")   # seasonally adjusted
series(m, "s11")   # trend-cycle
```

9.32.6 Output & Results

`seas()` runs the X-13 binary in the background and returns a `seas` object with the RegARIMA model summary, the various decomposed series accessible via `series()`, and a panel of diagnostics. `view(m)` (or `inspect(m)`) opens an interactive dashboard for diagnostic exploration.

9.32.7 Interpretation

A reporting sentence: "X-13ARIMA-SEATS seasonal adjustment of monthly air-passenger counts identified a $(0,1,1)(0,1,1)_{12}$ RegARIMA structure with no significant outliers; the SEATS-based decomposition produced a smooth seasonally adjusted trend with stable seasonal factors that gradually amplified across the series, consistent with the multiplicative seasonality." Always state the chosen ARIMA structure and the decomposition method (SEATS or X-11).

9.32.8 Practical Tips

- The `seasonal` package automatically installs the X-13 binary on first use; verify with `seasonal::checkX13()`.
- X-13 is the official standard for seasonal adjustment of economic series; required by some regulators and statistical agencies.
- `view(m)` opens an interactive HTML dashboard with all diagnostics — extremely useful for exploring complex decompositions.
- Trading-day, Easter, and outlier regressors are added automatically in default mode but can be customised; well-chosen regressors substantially improve fit for retail and financial series.
- For non-economic series or series with non-standard seasonal periods, STL (`stl()` or `mstl()`) is a simpler alternative without the calendar-effect machinery.
- Compare X-13 against STL and classical decomposition; large discrepancies signal calendar or outlier effects that one method handles better than another.

9.32.9 R Packages Used

`seasonal` for `seas()`, `series()`, and the X-13 interface; `seasonalview` for the interactive HTML dashboard; `forecast::stl()` and `mstl()` for simpler decomposition alternatives; `feasts::X_13ARIMA_SEATS()` for the modern tidyverts wrapper.

9.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

9.32.11 See also — labs in this chapter

- Time series basics

9.33 Week 2, Session 5 — Time series basics

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

9.33.1 Learning objectives

- Decompose a seasonal time series into trend, seasonal, and residual components.
- Fit an ARIMA model automatically with `forecast::auto.arima`.
- Detect a mean shift using `changepoint::cpt.mean`.

9.33.2 Prerequisites

Regression; basic familiarity with the `ts` class.

9.33.3 Background

A time series differs from cross-sectional data in one crucial way: observations close in time are correlated with one another, so standard errors based on independence are wrong. Classical decomposition splits a series into a slowly varying trend, a periodic seasonal component, and a residual. ARIMA models capture the residual autocorrelation with autoregressive and moving-average terms, possibly after differencing to remove a unit root.

Change-point detection asks a different question: given a series, when (if ever) did the underlying mean or variance shift? The `changepoint` package implements several methods; `cpt.mean` with a penalised likelihood criterion is a common starting point. In epidemiology, change-points flag outbreaks, policy changes, and data-collection transitions.

A seasonally adjusted series is not the same as a detrended series. Seasonal adjustment removes only the regular period; detrending removes the slow evolution. Most anomalies you care about (an outbreak, an intervention) live in what remains.

9.33.4 Setup

```
library(tidyverse)
library(forecast)
library(changepoint)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

9.33.5 1. Hypothesis

A simulated monthly series has a linear trend, a yearly seasonal cycle, and a mean shift at month 80. Decomposition, ARIMA, and change-point detection should each recover an interpretable piece.

9.33.6 2. Visualise

```
months <- 144
t  <- seq_len(months)
trend <- 0.08 * t
season <- 4 * sin(2 * pi * t / 12)
shift  <- if_else(t >= 80, 5, 0)
y      <- 20 + trend + season + shift + rnorm(months, 0, 1.5)
ts_y   <- ts(y, frequency = 12, start = c(2014, 1))

autoplot(ts_y) +
  labs(x = "Year", y = "Simulated rate")
```

9.33.7 3. Assumptions

STL decomposition assumes the seasonal period is fixed and known. ARIMA assumes a linear, time-invariant generating process after differencing. `cpt.mean` assumes the variance is approximately constant — change-points in variance would need `cpt.var`.

9.33.8 4. Conduct

```
decomp <- stl(ts_y, s.window = "periodic")
autoplot(decomp)
```

```
fit_arima <- auto.arima(ts_y)
fit_arima
```

```
fc <- forecast(fit_arima, h = 12)
autoplot(fc) +
  labs(x = "Year", y = "Simulated rate")
```

```
cpt <- cpt.mean(as.numeric(ts_y), method = "PELT")
cpts(cpt)
```

```
tibble(t = t, y = as.numeric(ts_y)) |>
  ggplot(aes(t, y)) +
  geom_line() +
  geom_vline(xintercept = cpts(cpt), linetype = "dashed",
```

```
            colour = "firebrick") +
  labs(x = "Month index", y = "Rate")
```

9.33.9 5. Concluding statement

STL cleanly separated a linear trend, a 12-month seasonal cycle, and a residual containing the simulated mean shift. `auto.arima` selected `fit_arima$arma|>paste(collapse=",")` (p, q, P, Q, frequency, d, D). PELT detected change-points at months `paste(cpts(cpt),collapse =",")`, close to the simulated truth of 80.

9.33.10 Common pitfalls

- Using daily ARIMA on weekly-seasonal data without telling the model the period.
- Calling every bump a change-point; penalise harder and re-run.
- Using `lm()` on a time series as if residuals were independent.
- Forecasting far beyond the observed period without widening the intervals in the report.

9.33.11 Further reading

- Hyndman RJ, Athanasopoulos G. *Forecasting: Principles and Practice*.
- Killick R, Fearnhead P, Eckley IA (2012), *Optimal detection of change-points with a linear computational cost*.
- Box GEP, Jenkins GM. *Time Series Analysis: Forecasting and Control*.

9.33.12 Session info

```
sessionInfo()
```

9.33.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 10

Bayesian Statistics

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Posterior inference via `brms` and Stan: priors, hierarchical models, posterior predictive checks, leave-one-out cross-validation, and Bayes factors. The chapter is opinionated: it treats Bayesian inference as a default, not an alternative.

This chapter contains **35 method pages** and **2 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

10.1 Method pages

Method	Source slug
Bayes Factors	`bayes-factors`
Bayes' Theorem for Parameters	`bayes-theorem-for-parameters`
Bayesian ANOVA	`bayesian-anova`
Bayesian Hierarchical Models	`bayesian-hierarchical-intro`
Bayesian Hypothesis Testing	`bayes-hypothesis-testing`
Bayesian Linear Regression	`linear-regression-bayes`
Bayesian Logistic Regression	`logistic-regression-bayes`
Bayesian Mediation Analysis	`bayesian-mediation`
Bayesian Meta-Analysis	`bayesian-meta-analysis`
Bayesian Mixed Models	`bayesian-mixed-models`
brms Basics	`brms-basics`
Conjugate Priors	`conjugate-priors`
Credible Intervals	`credible-intervals`
Divergences and Pairs Plots	`divergences-pairs-plots`
Gibbs Sampling	`gibbs-sampling`
Hamiltonian Monte Carlo	`hamiltonian-monte-carlo`

Method	Source slug
Highest Posterior Density Interval	`hpd-interval`
Informative Priors	`informative-priors`
Jeffreys' Prior	`jeffreys-prior`
Leave-One-Out Cross-Validation	`loo-cv`
Metropolis-Hastings	`metropolis-hastings`
Posterior Mean and Variance	`posterior-mean-variance`
Posterior Predictive Checks	`posterior-predictive-checks`
Prior Specification	`prior-specification`
R-hat and ESS Diagnostics	`rhat-ess-diagnostics`
rstanarm	`rstanarm`
Stan: Introduction	`stan-introduction`
The Beta-Binomial Model	`beta-binomial-model`
The Gamma-Poisson Model	`gamma-poisson-model`
The Normal-Normal Model	`normal-normal-model`
The Posterior Predictive Distribution	`posterior-predictive-distribution`
The Savage-Dickey Density Ratio	`savage-dickey-ratio`
tidybayes Workflow	`tidybayes-workflow`
WAIC	`waic`
Weakly Informative Priors	`weakly-informative-priors`

10.2 Labs

Lab
Bayesian thinking; α control vs decision error
brms / Stan, LOO, hierarchical models

10.3 Bayes Factors

10.3.1 Introduction

A Bayes factor compares two models by the ratio of how plausible the observed data are under each, after integrating over the parameters of each model with respect to its prior. It is the Bayesian counterpart of a likelihood-ratio test, but it carries a richer interpretation: rather than rejecting or failing to reject a null, it returns a continuous strength-of-evidence measure that can equally support the null model when the data are inconsistent with the alternative. This symmetry — quantifying support for either side — is one of the main reasons Bayes factors recur in applied work where "no effect" is itself a substantively interesting conclusion.

10.3.2 Prerequisites

A working understanding of Bayes' theorem, the role of the marginal likelihood (evidence) in Bayesian inference, and the difference between point-null and composite hypotheses.

10.3.3 Theory

For two models M_0 and M_1 with parameter vectors θ_0 and θ_1 and priors $p(\theta_0 \mid M_0)$ and $p(\theta_1 \mid M_1)$, the Bayes factor in favour of M_1 is

$$\mathrm{BF}_{10} = \frac{p(y \mid M_1)}{p(y \mid M_0)} = \frac{\int p(y \mid \theta_1, M_1) p(\theta_1 \mid M_1)\, \mathrm{d}\theta_1}{\int p(y \mid \theta_0, M_0) p(\theta_0 \mid M_0)\, \mathrm{d}\theta_0}.$$

Posterior odds equal the Bayes factor times the prior odds; under equal prior odds, the posterior probability of M_1 is $\mathrm{BF}_{10}/(\mathrm{BF}_{10}+1)$. Jeffreys' interpretive scale calls $\mathrm{BF} > 3$ substantial, > 10 strong, > 30 very strong, and > 100 decisive evidence; the same thresholds apply, mirrored, for evidence in favour of M_0.

10.3.4 Assumptions

Both models are well-specified and proper priors are assigned to all parameters that differ between the two; improper priors leave the marginal likelihoods undefined up to an arbitrary constant and the Bayes factor inherits that indeterminacy. Numerical estimation must be stable — bridge sampling, Chib's method, or Savage-Dickey ratios for nested models.

10.3.5 R Implementation

```
library(BayesFactor)

# Simple example: one-sample t-test equivalent
x <- rnorm(40, mean = 0.5, sd = 1)
ttestBF(x)

# Regression
data(mtcars)
full <- lmBF(mpg ~ wt + hp, data = mtcars)
reduced <- lmBF(mpg ~ wt, data = mtcars)
full / reduced
```

10.3.6 Output & Results

The `BayesFactor` package returns a Bayes factor with an estimated error percentage. For regression-type models the output also includes the implied posterior

model probabilities; ratios of `BFBayesFactor` objects deliver pairwise comparisons directly.

10.3.7 Interpretation

A reporting sentence: "A default-prior t-test Bayes factor was $\mathrm{BF}_{10} = 4.7$ ($\pm 0.001\%$), interpreted as substantial evidence that the population mean differs from zero." When reporting, always include the prior and (for `BayesFactor`) the prior scale parameter `rscale`; readers cannot otherwise judge how much of the evidence is data-driven versus prior-driven.

10.3.8 Practical Tips

- Wider priors mechanically shift Bayes factors toward the null (Lindley–Bartlett paradox); a sensitivity analysis across at least two prior scales is mandatory for any Bayes-factor conclusion.
- `bridgesampling::bridge_sampler()` works directly with `brms` and Stan fits; it is the most general way to compute marginal likelihoods for non-trivial models.
- For nested null hypotheses, the Savage-Dickey density ratio is faster, more stable, and avoids marginal-likelihood estimation entirely.
- Bayes factors near 1 are genuinely inconclusive; resist the temptation to declare evidence with BF = 1.5 in either direction.
- Bayes factors are not a substitute for predictive performance; pair them with `loo()` when comparing models for prediction, since the two answer different questions.

10.3.9 R Packages Used

`BayesFactor` for default-prior tests and ANOVA / regression Bayes factors, `bridgesampling` for general marginal-likelihood estimation, `bayestestR` for tidy summaries that work with `brms` and `rstanarm` fits, and `polspline` for Savage-Dickey density estimation when the null is nested.

10.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.3.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.4 Bayesian Hypothesis Testing

10.4.1 Introduction

Bayesian decision-making about effects splits into two cultural traditions that complement rather than replace each other. The Bayes-factor school computes a ratio of marginal likelihoods and reads strength of evidence from Jeffreys' scale. The Region of Practical Equivalence (ROPE) school, popularised by Kruschke, defines a range of effects that are practically null and asks how much of the posterior lies inside it. Both produce decisions about hypotheses, but they answer slightly different questions: Bayes factors compare models, while ROPE asks whether the data have ruled out a substantively negligible effect.

10.4.2 Prerequisites

A working understanding of posterior distributions, credible intervals, marginal likelihoods, and the difference between statistical significance and practical importance.

10.4.3 Theory

ROPE. Specify, before looking at the data, a parameter range $[a, b]$ inside which an effect is considered practically zero — for example $\beta \in (-0.1, 0.1)$ on a standardised scale. After fitting, compute the proportion of the 95 % HPD (or the full posterior) inside that range. Conventional rules: $> 95\%$ inside means accept the null for practical purposes, 0% inside means reject it, and intermediate values are inconclusive.

Bayes factor. Ratio of marginal likelihoods under two models; for a point null nested in a continuous alternative, the Savage-Dickey density ratio gives the same answer as the ratio of posterior to prior densities at the null. Both quantities depend on the prior under the alternative, so disclosing the prior is part of the report.

10.4.4 Assumptions

ROPE assumes a domain-relevant equivalence margin set in advance; chasing margins after seeing the posterior is a Bayesian variant of p-hacking. Bayes factors assume proper priors and a likelihood that admits stable marginal-likelihood computation.

10.4.5 R Implementation

```
library(bayestestR); library(brms)

set.seed(2026)
```

```
d <- data.frame(y = rnorm(200), x = rnorm(200))
fit <- brm(y ~ x, data = d, chains = 4, iter = 2000, refresh = 0)

rope(fit, range = c(-0.1, 0.1))
bayesfactor_parameters(fit)
describe_posterior(fit)
```

10.4.6 Output & Results

`rope()` returns the percentage of the HPD (or full posterior, depending on the chosen `ci`) inside the equivalence region for each parameter. `bayesfactor_parameters()` returns the Savage-Dickey Bayes factor against the point null. `describe_posterior()` is a one-stop summary that combines the two with point estimates and credible intervals.

10.4.7 Interpretation

A reporting sentence: "The posterior for β_x had 4 % of its 95 % HPD inside the practical-equivalence region $(-0.1, 0.1)$ and a Savage-Dickey Bayes factor of $\mathrm{BF}_{10} = 7.2$ — substantial evidence for a non-null effect on Jeffreys' scale, with most of the posterior lying outside the practical-null region." When the two methods disagree (ROPE inconclusive but Bayes factor decisive, or vice versa), report both and explain which question your study was designed to answer.

10.4.8 Practical Tips

- Pre-register the ROPE based on what is practically relevant in your field; for standardised effects, ± 0.1 is a common default but is arbitrary in absolute terms.
- For ROPE on raw scales, work out the smallest effect that would change clinical or business behaviour and use that as the half-width.
- Bayes factors are sensitive to the prior under the alternative (Lindley–Bartlett paradox); always run a sensitivity analysis with at least two prior scales.
- Use `bayestestR::describe_posterior()` to deliver a single block of summaries that includes credible intervals, ROPE, Bayes factors, and probability of direction; consistent reporting reduces ambiguity.
- ROPE is easier to communicate to non-statistical audiences ("most of the posterior is outside the negligible-effect range") than Bayes factors are; choose the framing that fits your reader.

10.4.9 R Packages Used

`bayestestR` for `rope()`, `bayesfactor_parameters()`, and `describe_posterior()`; `brms` or `rstanarm` for the underlying Bayesian fit; `bridgesampling` for marginal-

likelihood-based Bayes factors when the Savage-Dickey shortcut does not apply.

10.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.4.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.5 Bayes' Theorem for Parameters

10.5.1 Introduction

Bayesian inference is, mechanically, the application of Bayes' theorem to a parameter rather than to an event. The prior encodes whatever was believed before the data arrived, the likelihood is the data-generating model evaluated at the observed data, and the posterior is the updated belief. The conceptual leap from frequentist statistics is small but consequential: parameters are themselves random quantities whose posterior distribution can be summarised, predicted from, and acted upon, without invoking long-run sampling thought experiments.

10.5.2 Prerequisites

A working understanding of Bayes' theorem for events, the concept of a likelihood function, and the difference between probability density and probability mass.

10.5.3 Theory

For a parameter θ and data y, Bayes' theorem reads

$$p(\theta \mid y) = \frac{p(y \mid \theta)p(\theta)}{p(y)} \propto p(y \mid \theta)p(\theta),$$

where the four quantities are the posterior, the likelihood, the prior, and the marginal likelihood $p(y) = \int p(y \mid \theta)p(\theta)\,\mathrm{d}\theta$. The marginal likelihood appears as a normalisation constant from a single-model perspective but plays the central role in Bayes-factor-based model comparison. For closed-form (conjugate) priors, the posterior comes out in the same family as the prior with updated

hyperparameters; otherwise MCMC produces draws from the un-normalised right-hand side.

10.5.4 Assumptions

The prior reflects genuine pre-data information or a defensible weak-regularisation choice, the likelihood correctly captures the data-generating process, and posterior integrability is established (proper priors guarantee this; improper priors must be checked).

10.5.5 R Implementation

```r
# Beta-Binomial: prior Beta(2, 2), 8 successes in 10 trials
prior_a <- 2; prior_b <- 2
k <- 8; n <- 10
post_a <- prior_a + k
post_b <- prior_b + n - k

theta <- seq(0, 1, length.out = 400)
plot(theta, dbeta(theta, prior_a, prior_b), type = "l",
     main = "Prior -> Posterior", ylab = "Density", col = "grey")
lines(theta, dbeta(theta, k + 1, n - k + 1), col = "orange")  #
 ↪  likelihood scaled
lines(theta, dbeta(theta, post_a, post_b), col = "steelblue", lwd = 2)
legend("topleft", c("Prior", "Likelihood", "Posterior"),
       col = c("grey", "orange", "steelblue"), lty = 1)

# Posterior summaries
c(mean = post_a / (post_a + post_b),
  ci95 = qbeta(c(0.025, 0.975), post_a, post_b))
```

10.5.6 Output & Results

A three-curve plot showing the flat-ish prior, the peaked likelihood, and the posterior between them, plus a summary giving the posterior mean (0.71) and the 95 % equal-tailed credible interval (≈ 0.47 to 0.89). The posterior is closer to the likelihood than to the prior because the data ($n = 10$) carry more precision than the prior ($a + b = 4$ pseudo-observations).

10.5.7 Interpretation

A reporting sentence: "Combining a Beta(2, 2) prior with 8 successes in 10 trials yielded a Beta(10, 4) posterior with mean 0.71 (95 % credible interval 0.47 to 0.89); the prior contributed the equivalent of about four pseudo-observations and the data dominated the posterior shape." Always disclose the prior; the same data with a different prior would lead to a slightly different posterior.

10.5.8 Practical Tips

- For conjugate setups (Beta–Binomial, Normal–Normal, Gamma–Poisson, Dirichlet–Multinomial), the posterior is closed-form and arithmetic alone suffices; reach for MCMC only when conjugacy fails.
- Always report a posterior summary that includes both a central tendency and an uncertainty measure; a posterior mean alone is no more informative than a frequentist point estimate.
- Sensitivity to the prior matters most in small samples; if your conclusion changes with a wider or narrower prior, the data are weak and the analysis is genuinely prior-driven.
- Posterior predictive draws (`rbeta(M,post_a,post_b)` followed by `rbinom(M,n_new,theta)`) deliver direct uncertainty intervals for future outcomes.
- Visualising prior, likelihood, and posterior on the same plot is the single most useful pedagogical artefact when explaining Bayesian inference to non-statistical readers.

10.5.9 R Packages Used

Base R for the conjugate update and plotting, `bayestestR` for tidy posterior summaries, and `brms` or `rstanarm` for non-conjugate problems where MCMC is required to obtain posterior draws.

10.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.5.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.6 Bayesian ANOVA

10.6.1 Introduction

Classical ANOVA reports an F-statistic, a p-value, and — if the omnibus is significant — pairwise comparisons with a multiple-testing correction tacked on. A Bayesian ANOVA replaces all of that with a single posterior distribution over the group means and any contrast you care to compute. There is no omnibus to clear before contrasts become legitimate, and there is no multiple-comparison adjustment to argue about, because every contrast already has its own honest

credible interval reflecting the joint posterior of the parameters that compose it.

10.6.2 Prerequisites

A working understanding of one-way ANOVA, the distinction between fixed and random effects, and Bayesian regression. Familiarity with `emmeans` for marginal means makes the contrast step natural.

10.6.3 Theory

The model is $y_{ij} \sim \mathcal{N}(\mu_i, \sigma^2)$ with $\mu_i = \mu + \alpha_i$. Two parameterisations are common: a fixed-effects parameterisation with weakly informative priors on the contrasts, and a hierarchical parameterisation with $\alpha_i \sim \mathcal{N}(0, \tau^2)$ that pools group estimates toward the grand mean. The hierarchical version is preferable when group counts are large or when groups represent samples from a broader population. Pairwise contrasts and any linear combination of μ_i are computed at the level of MCMC draws, so each draw produces a contrast value and the posterior of the contrast is the empirical distribution of those values.

10.6.4 Assumptions

Approximately Normal residuals within groups, homogeneity of variance unless `sigma` is also modelled by group, exchangeability of groups under the hierarchical version, and proper priors. Independence between observations after group is the same Bayesian assumption as in classical ANOVA.

10.6.5 R Implementation

```
library(brms); library(bayestestR)

data(PlantGrowth)

fit <- brm(weight ~ group, data = PlantGrowth,
           prior = prior(normal(0, 1), class = "b"),
           chains = 4, iter = 2000, refresh = 0)

# Pairwise contrasts
pairs_summary <- pairs(emmeans::emmeans(fit, ~ group))
pairs_summary

# Bayes factors for effect
bayesfactor_parameters(fit)
```

10.6.6 Output & Results

`emmeans()` returns the posterior marginal mean for each group; `pairs()` returns the posterior of every pairwise difference with an HPD or equal-tailed credible interval. `bayesfactor_parameters()` evaluates the Savage-Dickey density ratio at zero for each contrast, returning a Bayes factor for "non-zero contrast" against the null.

10.6.7 Interpretation

A reporting sentence: "Posterior contrasts from a Bayesian one-way ANOVA showed `trt1` − `ctrl` = −0.37 (95 % credible interval −0.95 to 0.21, $\mathrm{BF}_{10} = 1.1$), `trt2` − `ctrl` = 0.49 (95 % CrI 0.04 to 0.94, $\mathrm{BF}_{10} = 4.6$); the data weakly support a treatment-2 effect and are inconclusive about treatment 1." Pair the credible interval with the Bayes factor only when the prior under the alternative is disclosed, since the latter depends sensitively on it.

10.6.8 Practical Tips

- For more than five or six groups, switch to a hierarchical (1 | group) random-effects parameterisation; shrinkage stabilises small-group estimates and gives a principled population-level summary.
- `emmeans()` works directly on `brms` fits and integrates with `pairs()`, `contrast()`, and `confint()` to produce posterior contrasts on whatever scale you need.
- Credible intervals already adjust for the joint posterior; multiple-comparison corrections in the frequentist sense are unnecessary, but you can still apply a region-of-practical-equivalence (ROPE) test if you want a decision rule.
- For unequal variances, fit `bf(weight~group,sigma~group)` so the residual scale is also estimated per group; the contrast machinery generalises without changes.
- Report posteriors graphically (`mcmc_areas`, `stat_halfeye`) alongside the numeric summaries; readers absorb the joint structure faster from a plot than from a contrast table.

10.6.9 R Packages Used

`brms` for model specification and sampling, `emmeans` for posterior marginal means and contrasts, `bayestestR` for Bayes factors and ROPE summaries, and `tidybayes` for tidy posterior draws of contrasts when custom plots are needed.

10.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.6.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.7 Bayesian Hierarchical Models

10.7.1 Introduction

A hierarchical Bayesian model embeds the group-level parameters of a multilevel design inside a higher-level prior, so that the group-level estimates inform each other rather than being computed in isolation. The pay-off is partial pooling: groups with abundant data are estimated almost from their own observations, groups with little data are pulled toward the overall mean, and every estimate carries a credible interval that reflects both the within-group sample size and the between-group heterogeneity. This single mechanism resolves a long list of practical problems — small-sample variance, unbalanced designs, sparse counts — that frequentist mixed-effects models can only address by reweighting or by appealing to asymptotic theory.

10.7.2 Prerequisites

A working understanding of Bayesian inference, the random-effects mixed model, and the role of exchangeability in motivating shared priors.

10.7.3 Theory

A two-level model has

- Level 1: $y_{ij} \mid \theta_i \sim p(y \mid \theta_i)$ — the data given group-specific parameters.
- Level 2: $\theta_i \mid \mu, \tau \sim p(\theta \mid \mu, \tau)$ — group-specific parameters drawn from a population distribution.
- Level 3 (hyperprior): $\mu \sim p(\mu)$, $\tau \sim p(\tau)$ — priors on the population parameters.

The posterior of θ_i is a precision-weighted average of the within-group estimate (driven by the data in group i) and the population mean μ, with weight inversely proportional to the within-group variance and to the cross-group variance τ^2. Smaller groups rely more on the population mean; larger groups stand on their own data. This is the formal expression of "shrinkage" or "borrowing strength."

10.7.4 Assumptions

Exchangeability of groups (group labels carry no information beyond what is in the predictors) and a hyperprior whose support spans plausible values of μ and τ. A poorly chosen hyperprior — particularly an overly tight prior on τ — can dominate the posterior in small samples and look like a strong empirical claim.

10.7.5 R Implementation

```r
library(brms)

# 30 hospitals, events and trials per hospital
set.seed(2026)
n_hosp <- 30
n_per  <- sample(5:50, n_hosp, replace = TRUE)
true_rates <- rbeta(n_hosp, 3, 12)
events <- rbinom(n_hosp, n_per, true_rates)

d <- data.frame(hospital = factor(1:n_hosp),
                events = events, trials = n_per)

fit <- brm(events | trials(trials) ~ 1 + (1 | hospital),
           data = d, family = binomial,
           chains = 4, iter = 2000, refresh = 0)
summary(fit)
```

10.7.6 Output & Results

`summary(fit)` returns the population-level intercept (overall log-odds), the between-hospital SD on the logit scale, and per-hospital posterior summaries via `ranef(fit)`. The shrinkage of small-hospital estimates toward the population mean is most visible when you plot raw event proportions against posterior means alongside hospital sample size.

10.7.7 Interpretation

A reporting sentence: "A Bayesian binomial hierarchical model estimated the population baseline at $\text{logit}^{-1}(-1.4) = 0.20$ (95 % credible interval 0.16 to 0.24); the between-hospital standard deviation on the logit scale was 0.51 (95 % CrI 0.34 to 0.74), and small-hospital posterior means were drawn substantially toward the population baseline." Always report cluster-specific and overall estimates; one without the other misses the central insight of the hierarchical fit.

10.7.8 Practical Tips

- Partial pooling is the headline benefit: do not throw it away by switching to a fixed-effects fit just because cluster counts are large.
- For divergent transitions, switch to non-centred parameterisation; `brms` does this automatically when you specify `prior(...,class="sd",...)`.
- The prior on τ matters enormously when the number of clusters is small; report a sensitivity analysis with at least one wider and one tighter prior.
- For more than two levels (e.g. patient $\subset$ hospital $\subset$ region), nest random effects: `(1|region/hospital)`. The same shrinkage logic applies at each level.
- Do not interpret cluster-specific posterior means as if they came from independent fits; they are *intentionally* shrunk and that shrinkage is the right answer to "how good is hospital i?"

10.7.9 R Packages Used

`brms` for the hierarchical Bayesian fit, `rstanarm` for pre-compiled `stan_glmer` alternatives, `tidybayes` for tidy cluster-level draws when bespoke shrinkage plots are needed, and `bayesplot` for diagnostic visualisation.

10.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.7.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.8 Bayesian Mediation Analysis

10.8.1 Introduction

Mediation analysis decomposes a treatment effect into the part that flows through an intermediate variable (the mediator) and the part that does not. Frequentist approaches build confidence intervals around the product $a_1 b_2$ via Sobel tests or non-parametric bootstrapping; both have to compensate for the fact that the product of two normally distributed estimates is itself non-normal. A Bayesian formulation sidesteps the issue: the joint posterior of (a_1, b_2) is sampled directly, the indirect effect is computed draw-by-draw as the product,

and any quantile of the resulting posterior is a valid credible interval without distributional assumptions.

10.8.2 Prerequisites

A working understanding of multiple regression, the concept of mediation in causal inference, and Bayesian regression. Familiarity with multivariate `brms` formulas accelerates the implementation step.

10.8.3 Theory

The classical Baron-Kenny system has two regressions

$$M = a_0 + a_1 X + \varepsilon_M, \qquad Y = b_0 + b_1 X + b_2 M + \varepsilon_Y,$$

with errors that are typically taken Normal. The direct effect of X on Y is b_1, the indirect effect through M is $a_1 b_2$, and the total effect is $b_1 + a_1 b_2$. In the Bayesian formulation, the two equations are fitted jointly so that the posterior carries the joint uncertainty in (a_1, b_2). The indirect-effect posterior is the empirical distribution of $\{a_1^{(s)} b_2^{(s)}\}_{s=1}^{S}$; the credible interval comes from its quantiles. No symmetry assumption is needed and the posterior captures any skew induced by the product.

10.8.4 Assumptions

Linearity of both equations, Normal residuals (or whatever family you specify), no unmeasured confounders of the $M \to Y$ relation, and correctly ordered causal direction $X \to M \to Y$. Mediation analysis is a causal claim; the Bayesian machinery does not make weak design choices stronger.

10.8.5 R Implementation

```
library(brms)

set.seed(2026)
n <- 300
X <- rnorm(n)
M <- 0.5 * X + rnorm(n)
Y <- 0.3 * X + 0.6 * M + rnorm(n)
d <- data.frame(X, M, Y)

# Joint model using brms multivariate syntax
bf1 <- bf(M ~ X)
bf2 <- bf(Y ~ X + M)
```

```
fit <- brm(bf1 + bf2 + set_rescor(FALSE), data = d,
           chains = 4, iter = 2000, refresh = 0)

post <- as_draws_df(fit)
indirect <- post$b_M_X * post$b_Y_M
quantile(indirect, c(0.025, 0.5, 0.975))
```

10.8.6 Output & Results

A multivariate `brms` fit jointly estimates the two regressions and stores draws for both. Computing the elementwise product of posterior draws of a_1 and b_2 produces the indirect-effect posterior; quantiles of that posterior give the credible interval directly. The total effect, the direct effect, and the proportion mediated all come from the same draw matrix without re-fitting.

10.8.7 Interpretation

A reporting sentence: "Bayesian mediation estimated the indirect effect of X on Y through M at 0.30 (95 % credible interval 0.18 to 0.45) and the direct effect at 0.28 (95 % CrI 0.14 to 0.42); the indirect path explained approximately 52 % of the total effect (95 % CrI 35 % to 68 %)." Always report both direct and indirect effects with credible intervals; reporting only the indirect product hides whether the direct path remains substantive.

10.8.8 Practical Tips

- Use multivariate `brms` formulas (`bf(M~X)+bf(Y~X+M)+set_rescor(FALSE)`) so that the two regressions are fitted jointly and the joint posterior is preserved; running them separately throws away the cross-equation correlation needed for honest indirect-effect intervals.
- Place priors on each regression's coefficients individually; weakly informative defaults ($\mathcal{N}(0, 2.5)$ on standardised scales) are usually appropriate.
- Sensitivity analysis to unmeasured confounding is more important than the choice of estimator; consider `causalsens` or Bayesian instrumental-variable extensions when the assumption is weak.
- For binary or count mediators / outcomes, change the family in the corresponding `bf()` and the indirect-effect computation generalises; just use `posterior_epred()` to translate to the natural-effects scale.
- Reporting the proportion mediated ($a_1 b_2 / (b_1 + a_1 b_2)$) requires care because the denominator can be small or change sign across draws; truncate or report the indirect and total effects separately when the proportion is unstable.

10.8.9 R Packages Used

`brms` for joint multivariate regressions and direct posterior products of coefficients, `bmlm` for purpose-built Bayesian mediation with multilevel structure, `tidybayes` for tidy posterior draws of the indirect effect, and `bayestestR` for credible intervals and ROPE summaries on the derived quantities.

10.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.8.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.9 Bayesian Meta-Analysis

10.9.1 Introduction

Meta-analysis with k studies is awkward for frequentist random-effects estimators when k is small: REML and DerSimonian-Laird both rely on asymptotic approximations that break down when only five or six studies contribute, and the between-study variance τ^2 can collapse to zero or be wildly imprecise. A Bayesian formulation puts an explicit prior on τ and propagates uncertainty into both the pooled effect and the prediction interval for a new study, which is exactly the inferential question that drives most meta-analyses in clinical and biomedical work.

10.9.2 Prerequisites

A working understanding of fixed- and random-effects meta-analysis, exchangeability assumptions, and hierarchical Bayesian models. Familiarity with `brms` formula syntax accelerates the implementation step.

10.9.3 Theory

The hierarchical model is

$$y_i \sim \mathcal{N}(\theta_i, s_i^2), \qquad \theta_i \sim \mathcal{N}(\mu, \tau^2),$$

where y_i and s_i are the observed effect estimate and its standard error from study i, θ_i is the latent study-specific effect, μ is the population effect, and τ is the between-study standard deviation. Typical priors are weakly informative: $\mu \sim \mathcal{N}(0, 2)$ on the standardised-mean-difference scale and $\tau \sim$ half-Normal$(0, 0.5)$ or half-Cauchy$(0, 0.3)$. The half-Cauchy gives the heavier tail favoured when very large heterogeneity is plausible. The posterior delivers μ, τ, every θ_i shrunk toward μ, and a predictive distribution $\mathcal{N}(\mu, \tau^2)$ for the next study.

10.9.4 Assumptions

Exchangeable studies, accurate within-study standard errors s_i, and proper priors. The within-study sampling distribution is taken as known; if s_i are themselves estimated from few patients, an additional level of hierarchy is needed.

10.9.5 R Implementation

```r
library(brms)

studies <- data.frame(
  yi = c(0.2, 0.3, -0.05, 0.4, 0.25),
  sei = c(0.10, 0.15, 0.08, 0.12, 0.20)
)

fit <- brm(yi | se(sei) ~ 1 + (1 | seq_len(nrow(studies))),
           data = studies,
           prior = prior(normal(0, 1), class = "Intercept") +
                   prior(cauchy(0, 0.5), class = "sd"),
           chains = 4, iter = 4000, refresh = 0)

summary(fit)
```

10.9.6 Output & Results

`summary(fit)` returns the pooled effect (`Intercept`) and the between-study SD (`sd(seq_len(...))`) with 95 % credible intervals, $\hat{R}$, and ESS. Per-study posterior means come from `ranef(fit)`; the predictive distribution for a future study is `posterior_predict()` with `re_formula=NA`.

10.9.7 Interpretation

A reporting sentence: "A Bayesian random-effects meta-analysis with a half-Cauchy(0, 0.5) prior on τ pooled the five studies to an effect of 0.23 (95 % credible interval 0.07 to 0.39); the between-study heterogeneity was small ($\tau = 0.09$, 95 % CrI 0.01 to 0.25), and the 95 % predictive interval for a future trial was -0.05 to 0.51." Reporting the predictive interval together with the pooled

estimate avoids the false sense of precision that a credible interval on μ alone can convey.

10.9.8 Practical Tips

- With $k < 10$ studies, the prior on τ matters substantially; a half-$t(3, 0, 1)$ or half-Normal(0, 0.5) is a reasonable compromise between letting the data speak and avoiding implausible values.
- Always report the prior in the methods section and conduct a sensitivity analysis with at least one wider and one tighter prior on τ.
- For binary outcomes, fit on the log-odds-ratio or risk-ratio scale rather than risk differences; the Normal-Normal hierarchy is more reasonable on the log scale.
- The `bayesmeta` package provides closed-form computations for simple models and is faster than MCMC when the prior is conjugate.
- For network meta-analysis with multiple treatments, switch to `gemtc` or `multinma`; the random-effects machinery generalises but the implementation is non-trivial.

10.9.9 R Packages Used

`brms` for the hierarchical Bayesian fit with study-level standard errors, `bayesmeta` for closed-form alternatives, `bayesplot` for diagnostic plots, and `gemtc` / `multinma` for the network-meta extension.

10.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.9.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.10 Bayesian Mixed Models

10.10.1 Introduction

Bayesian mixed models treat the random-effect variance components as parameters with their own priors instead of plug-in estimates from REML. The pay-off

is most visible in small samples and at the boundary: ML and REML routinely return zero variance estimates when only a handful of clusters contribute, but a half-Normal or half-Cauchy prior keeps the variance away from zero in a principled way and propagates the resulting uncertainty into every fixed-effect interval. The cost is computational — sampling instead of optimising — but for any analysis that depends on credible variance components, that cost is a worthwhile investment.

10.10.2 Prerequisites

A working understanding of frequentist mixed-effects models (`lme4`-style formulas), Bayesian regression, and the role of half-distributions as priors on standard deviations.

10.10.3 Theory

A random-intercept-and-slope model has the form

$$y_{ij} = \beta_0 + \beta_1 X_{ij} + u_{0i} + u_{1i} X_{ij} + \varepsilon_{ij},$$

with $(u_{0i}, u_{1i}) \sim \mathcal{N}(0, \Sigma)$ and $\varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$. Priors are placed on every component: weakly informative Normal priors on β, half-Normal or half-Cauchy on the diagonal of Σ, an LKJ prior on the correlation, and an exponential or half-t on σ. The LKJ(η) prior with $\eta > 1$ pulls correlations toward zero gently while still allowing strong correlations when the data demand them. Posterior draws of u_i are *shrunk* toward zero by an amount that depends on within-cluster signal-to-noise — the same partial-pooling behaviour that makes mixed models attractive in the first place.

10.10.4 Assumptions

Conditional independence of observations given random effects, exchangeability of clusters, approximately Normal random effects with the prior structure above, and proper priors on every variance and correlation parameter. Convergence diagnostics ($\widehat{R}$, divergent transitions, ESS) are part of the assumptions in practice.

10.10.5 R Implementation

```
library(brms)

data(sleepstudy, package = "lme4")

fit <- brm(Reaction ~ Days + (Days | Subject), data = sleepstudy,
           prior = prior(normal(0, 30), class = "b") +
```

```
                  prior(normal(250, 50), class = "Intercept") +
                  prior(exponential(0.1), class = "sd") +
                  prior(exponential(0.1), class = "sigma") +
                  prior(lkj(2), class = "cor"),
          chains = 4, iter = 2000, refresh = 0)
summary(fit)

# Visualise random effects
ranef(fit)
```

10.10.6 Output & Results

`summary(fit)` reports the population-level fixed effects, the standard deviations of the random intercept and slope, the correlation between them, and the residual scale, all with credible intervals, $\hat{R}$, and ESS. `ranef(fit)` returns a list of per-cluster posterior summaries — the analogue of `lme4::ranef()` but with full credible intervals.

10.10.7 Interpretation

A reporting sentence: "A Bayesian linear mixed model estimated the population-level slope at 10.5 ms per day of sleep deprivation (95 % credible interval 7.4 to 13.6); between-subject heterogeneity in slope was substantial (SD = 6.0, 95 % CrI 4.0 to 8.8) and weakly negatively correlated with the intercept ($r = -0.13$, 95 % CrI −0.49 to 0.27)." Always report the variance components with their credible intervals; point estimates alone can be misleading when posteriors are skewed.

10.10.8 Practical Tips

- Use `LKJ(2)` for correlation priors as a sensible default; tighten to `LKJ(4)` only when you have strong reason to expect weak correlations.
- For divergent transitions in hierarchical models, switch to non-centred parameterisation (`brms` does this automatically when you set `prior(...,class="sd",...)`).
- Standardise continuous predictors before fitting; weakly informative priors are then meaningful across columns.
- Compare nested random-effect structures with `loo_compare()` rather than likelihood-ratio tests; LOO works correctly when ML-LRT statistics are on the boundary of the parameter space.
- For very small n_{cluster} (under 5), report a sensitivity analysis across at least two priors on the random-effect SD; the choice can dominate the posterior.
- `rstanarm::stan_lmer()` mirrors `lme4` syntax and uses pre-compiled Stan models, useful when fit time matters and the model is standard.

10.10.9 R Packages Used

`brms` for formula-based Bayesian mixed models with full prior control, `rstanarm` for the pre-compiled `stan_lmer` interface, `bayesplot` for diagnostic plots, and `tidybayes` for tidy posterior draws when bespoke shrinkage plots are needed.

10.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.10.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.11 The Beta-Binomial Model

10.11.1 Introduction

The beta-binomial conjugate pair is the textbook entry point to Bayesian inference: a Beta prior on a probability combined with a binomial likelihood yields a Beta posterior whose parameters are obtained by adding successes to one shape parameter and failures to the other. The closed form makes the entire workflow — prior elicitation, posterior, credible interval, posterior predictive — tractable on a calculator. It is also the conceptual backbone for hierarchical models of proportions, which use a Beta prior at the population level to share strength across many groups.

10.11.2 Prerequisites

Familiarity with the Beta distribution as a flexible density on $[0, 1]$, the binomial likelihood for independent Bernoulli trials, and the basic mechanics of Bayesian updating.

10.11.3 Theory

With prior $\theta \sim \text{Beta}(a, b)$ and likelihood $X \sim \text{Binomial}(n, \theta)$, the posterior is

$$\theta \mid x \sim \text{Beta}(a + x, b + n - x).$$

The shape parameters of the prior have a direct interpretation as pseudo-counts: a pseudo-successes and b pseudo-failures observed before the data arrived, so the prior effective sample size is $a + b$. The posterior mean is

$$\mathbb{E}[\theta \mid x] = \frac{a + x}{a + b + n},$$

a weighted average of the prior mean $a/(a+b)$ and the sample proportion x/n, with weights determined by the prior pseudo-count and the actual sample size. Conjugacy also gives closed-form expressions for the posterior variance, credible intervals, and the predictive distribution for future Bernoulli trials.

10.11.4 Assumptions

Independent Bernoulli trials with a constant success probability θ, a Beta prior plausible enough that its tails do not exclude reasonable values, and a sample size that is fixed in advance — informative censoring or stopping rules invalidate the binomial likelihood.

10.11.5 R Implementation

```
# Prior Beta(2, 2) -- weakly informative around 0.5
# Data: 15 successes in 25 trials
a <- 2; b <- 2
x <- 15; n <- 25

post_a <- a + x
post_b <- b + n - x

c(prior_mean = a / (a + b),
  sample_prop = x / n,
  posterior_mean = post_a / (post_a + post_b),
  ci95 = qbeta(c(0.025, 0.975), post_a, post_b))

theta <- seq(0, 1, length.out = 400)
plot(theta, dbeta(theta, a, b), type = "l", col = "grey",
     ylab = "Density", main = "Beta-Binomial update")
lines(theta, dbeta(theta, post_a, post_b), col = "steelblue", lwd = 2)
```

10.11.6 Output & Results

A short numeric vector with the prior mean (0.50), the sample proportion (0.60), the posterior mean (0.59), and the 95 % equal-tailed credible interval (0.40 to 0.76). The prior-versus-posterior plot makes the pull of the prior visible: with only 25 trials the posterior shifts noticeably toward the data but retains the curvature of the prior.

10.11.7 Interpretation

A reporting sentence: "A Beta(2, 2) prior combined with 15 successes in 25 trials yielded a Beta(17, 12) posterior with mean 0.59 (95 % credible interval 0.40 to 0.76); the data have moved the posterior 0.09 above the prior mean while excluding values below 0.40 with 97.5 % probability." Always disclose the prior in the methods section so that readers can see how much shrinkage it imposes.

10.11.8 Practical Tips

- Beta$(1, 1)$ is the uniform reference prior; Beta$(0.5, 0.5)$ is Jeffreys; both keep posterior inference data-driven.
- For genuinely informative priors, set $a + b$ equal to the prior effective sample size you can defend (e.g. results from a prior trial of size 20 with proportion 0.7 give Beta$(14, 6)$).
- Use `qbeta()` for credible intervals and `pbeta()` for posterior tail probabilities; HPD intervals come from `HDInterval::hdi(qbeta(p,post_a,post_b))` evaluated on a grid.
- For multi-category outcomes, generalise to the Dirichlet–multinomial, which inherits the conjugate structure on a probability simplex.
- When the prior is contested, report a sensitivity analysis across at least Beta$(1, 1)$, Beta$(2, 2)$, and one informative pair; if the conclusion holds across all three, the result is robust to prior choice.

10.11.9 R Packages Used

Base R suffices for the conjugate update (`dbeta`, `pbeta`, `qbeta`, `rbeta`); `bayestestR::point_estimate()` and `HDInterval::hdi()` add tidy summaries and HPD intervals when a more polished workflow is needed.

10.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.11.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.12 brms Basics

10.12.1 Introduction

`brms` is the most widely used high-level interface between R and Stan. It accepts the familiar `lme4` formula syntax for fixed and random effects, automatically translates the model to Stan, compiles it, runs the No-U-Turn Sampler (NUTS), and returns a posterior together with diagnostic and predictive helpers. This means you can fit Bayesian generalised linear, mixed, multinomial, ordinal, censored, distributional, or non-linear models without writing Stan by hand, while still being able to inspect or override the generated code when the situation demands it.

10.12.2 Prerequisites

Comfort with classical regression, an understanding of priors and posteriors, and a working `rstan` or `cmdstanr` toolchain (a C++ compiler is required on first use). Familiarity with `lme4` formulas accelerates the learning curve dramatically because almost every random-effects expression carries over unchanged.

10.12.3 Theory

A `brms` model has three pieces: a **formula** (mean structure plus optional distributional terms), a **family** (likelihood and link), and **priors** (per parameter class). Random effects use `(slope|group)` syntax, with `||` for uncorrelated terms. Distributional models, set up through `bf(y~...,sigma~...)`, let any parameter of the family depend on predictors. Priors are attached by class: `b` for population-level slopes, `Intercept` for the centred intercept, `sd` for random-effect standard deviations, `sigma` for residual scale, and `cor` for correlation matrices via the LKJ distribution. Defaults are weakly informative but should be reviewed for any analysis intended for publication.

10.12.4 Assumptions

Inherits the assumptions of the chosen family — for `gaussian` that means linearity in the linear predictor, additivity, and Normal residuals after partial pooling. Convergence diagnostics ($\hat{R} < 1.01$, no divergent transitions, sensible effective sample size) are also a precondition for trusting any posterior summary.

10.12.5 R Implementation

```
library(brms)

data(sleepstudy, package = "lme4")

fit <- brm(Reaction ~ Days + (Days | Subject),
```

```
          data = sleepstudy,
          family = gaussian,
          prior = prior(normal(0, 30), class = "b") +
                  prior(exponential(0.1), class = "sigma") +
                  prior(exponential(0.1), class = "sd") +
                  prior(lkj(2), class = "cor"),
          chains = 4, iter = 2000, refresh = 0)

summary(fit)
pp_check(fit)
```

10.12.6 Output & Results

`summary()` reports posterior means, standard errors, 95 % credible intervals, $\hat{R}$, bulk effective sample size, and tail effective sample size for every population-level coefficient, group-level standard deviation, and the residual scale. `pp_check()` overlays simulated datasets on the observed outcome; if the empirical density falls inside the cloud of replicated densities, the model captures the marginal distribution adequately.

10.12.7 Interpretation

A reporting sentence: "A Bayesian linear mixed model (`brms` 2.21, four chains, 2,000 iterations) estimated reaction time to increase by 10.5 ms per day of sleep deprivation (95 % CrI 7.4 to 13.6), with substantial between-subject heterogeneity in both intercept ($SD = 26$) and slope ($SD = 6.5$); $\hat{R} = 1.00$ for all parameters." Always pair the point estimate with the credible interval and a convergence statistic.

10.12.8 Practical Tips

- Run `get_prior()` on your formula before fitting; it lists every prior class so you can override defaults instead of inheriting them silently.
- Inspect `stancode(fit)` and `standata(fit)` when something behaves oddly; the generated code is readable and reveals exactly how `brms` parameterised the model.
- `conditional_effects()` produces marginal effect plots with credible bands and is the fastest way to communicate non-linear or interaction effects to non-Bayesian audiences.
- Use `add_criterion(fit,"loo")` to attach LOO-CV for later model comparison; storing it on the object avoids re-fitting.
- For very large datasets, switch to `backend="cmdstanr"` and `threads_per_chain=2` to exploit within-chain parallelism.
- When defaults yield divergent transitions, increase `adapt_delta` to 0.95 or 0.99 before reaching for stronger priors.

10.12.9 R Packages Used

`brms` for model specification and compilation, `rstan` or `cmdstanr` as the sampling backend, `bayesplot` for diagnostic plots used internally by `pp_check()`, and `posterior` for handling draw arrays.

10.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.12.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.13 Conjugate Priors

10.13.1 Introduction

Before MCMC made arbitrary Bayesian models tractable, conjugate priors provided the only route to a tractable posterior. A prior is **conjugate** to a likelihood if the resulting posterior belongs to the same distributional family as the prior – the prior and posterior share a form, with updated parameters that have simple closed-form expressions. Conjugate pairs remain pedagogically central because they make the Bayesian update arithmetic, and they remain practically useful as building blocks within larger hierarchical models.

10.13.2 Prerequisites

The reader should know Bayes' theorem and be comfortable with the beta, normal, and gamma distributions. Familiarity with basic R plotting is assumed.

10.13.3 Theory

10.13.3.1 Beta-Binomial

For a binomial observation $Y \sim \text{Binomial}(n, \theta)$ with success probability θ and a beta prior $\theta \sim \text{Beta}(\alpha, \beta)$, the posterior is

$$\theta \mid Y = y \;\sim\; \text{Beta}(\alpha + y, \beta + n - y).$$

The update rule is nothing more than "add successes to the first shape parameter and failures to the second". The posterior mean is $(\alpha+y)/(\alpha+\beta+n)$, a weighted average of the prior mean $\alpha/(\alpha+\beta)$ and the observed proportion y/n, with weights determined by the prior's "effective sample size" $\alpha+\beta$.

10.13.3.2 Normal-Normal with known variance

For observations $Y_1, \dots, Y_n \overset{iid}{\sim} \mathcal{N}(\mu, \sigma^2)$ with σ^2 known, and a normal prior $\mu \sim \mathcal{N}(\mu_0, \tau_0^2)$, the posterior is

$$\mu \mid y \;\sim\; \mathcal{N}(\mu_n, \tau_n^2),$$

where the posterior precision is $1/\tau_n^2 = 1/\tau_0^2 + n/\sigma^2$, and the posterior mean is a precision-weighted average $\mu_n = \tau_n^2(\mu_0/\tau_0^2 + n\bar{y}/\sigma^2)$.

10.13.3.3 Gamma-Poisson

For independent counts $Y_i \sim \text{Poisson}(\lambda)$ and a gamma prior $\lambda \sim \text{Gamma}(\alpha, \beta)$ (shape-rate parameterisation), the posterior is

$$\lambda \mid y \;\sim\; \text{Gamma}\Big(\alpha + \sum y_i,\; \beta + n\Big)\,.$$

Each of these updates can be memorised as "add the data to the prior parameters", with the precise form determined by the sufficient statistics of the likelihood.

10.13.4 Assumptions

Conjugate analyses assume the same independence and distributional conditions as the corresponding likelihoods. In addition, they assume that the prior is genuinely an expression of prior belief, not a mathematical convenience chosen for conjugacy alone. An informative conjugate prior that contradicts the data produces a posterior that is a compromise between the two; if this is not what you intended, the prior was a bad choice.

10.13.5 R Implementation

A visualisation of the beta-binomial update makes the idea concrete:

```
library(ggplot2)

alpha_0 <- 2
beta_0  <- 2
y       <- 8
n       <- 10
```

```r
alpha_n <- alpha_0 + y
beta_n  <- beta_0 + n - y

theta <- seq(0, 1, length.out = 401)
df <- data.frame(
  theta = rep(theta, 3),
  density = c(dbeta(theta, alpha_0, beta_0),
              dbinom(y, n, theta) * n,
              dbeta(theta, alpha_n, beta_n)),
  kind = factor(rep(c("Prior", "Likelihood (scaled)", "Posterior"),
                    each = 401),
                levels = c("Prior", "Likelihood (scaled)", "Posterior"))
)

ggplot(df, aes(x = theta, y = density, colour = kind, linetype = kind)) +
  geom_line(linewidth = 1) +
  labs(x = expression(theta), y = "Density",
       title = "Beta-Binomial update with y = 8 of n = 10") +
  theme_minimal(base_size = 12)
```

A `Beta(2,2)` prior (symmetric, weakly informative around 0.5) combined with 8 successes in 10 trials gives a `Beta(10,4)` posterior concentrated around 0.71.

10.13.6 Output & Results

Posterior summaries from the beta distribution are:

- Mean: $(\alpha + y)/(\alpha + \beta + n) = 10/14 \approx 0.714$
- 95% equal-tailed credible interval: `qbeta(c(0.025,0.975),10,4)` $\approx (0.47, 0.91)$
- Probability that $\theta > 0.5$: `1-pbeta(0.5,10,4)` ≈ 0.95.

10.13.7 Interpretation

The posterior in a conjugate model is not an approximation; it is the exact posterior under the stated prior. A Bayesian reporting this analysis would write: "With a weakly informative Beta$(2, 2)$ prior and data of 8 successes in 10 trials, the posterior for θ is Beta$(10, 4)$, with posterior mean 0.71 (95% CrI 0.47 to 0.91). The posterior probability that θ exceeds 0.5 is 0.95."

10.13.8 Practical Tips

- Report the prior explicitly so readers can judge its influence.
- Use informative priors when justified by domain knowledge; use weakly informative priors as the default for exploratory analyses.

- Conjugate updating extends to multi-parameter cases (normal-inverse-gamma for (μ, σ^2), Dirichlet-multinomial for categorical probabilities), but outside the exponential family it usually does not hold.
- For small-sample binomial problems the beta-binomial is particularly useful: it avoids the ad-hoc continuity corrections of frequentist proportion tests and gives natural credible intervals.
- When your likelihood is non-conjugate, do not force a conjugate prior on it; use MCMC or variational methods instead.

10.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.13.10 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.14 Credible Intervals

10.14.1 Introduction

A credible interval is the Bayesian summary of posterior uncertainty most often reported alongside a point estimate. Unlike a confidence interval, whose probability statement refers to long-run frequency under hypothetical resampling, a credible interval supports the direct claim that "given the data and prior, there is a 95 % probability that the parameter lies inside this interval." That single sentence is the reason students consistently misinterpret confidence intervals — and the reason credible intervals are increasingly preferred in applied biomedical reporting.

10.14.2 Prerequisites

A working understanding of Bayesian inference, posterior distributions, and the distinction between a posterior and a sampling distribution.

10.14.3 Theory

Two definitions dominate practice:

- **Equal-tailed interval (ETI)**: the interval bounded by the $\alpha/2$ and $1-\alpha/2$ quantiles of the posterior. It is invariant under monotone reparameterisation and easy to compute from MCMC draws.
- **Highest posterior density (HPD) interval**: the narrowest set with $1-\alpha$ posterior mass; for unimodal posteriors it is a single interval centred on the region of highest density, and it is always at least as short as the ETI.

For symmetric posteriors the two intervals coincide; for skewed posteriors they can differ markedly. A 95 % level is conventional but arbitrary — some Bayesian methodologists advocate 89 % to discourage misreading the threshold as a hypothesis test.

10.14.4 Assumptions

A proper posterior distribution and either MCMC draws (post-warm-up, with adequate effective sample size) or a closed-form posterior. ETI endpoints are quantiles and tolerate small samples better than HPD endpoints, which depend on density estimation in the tails.

10.14.5 R Implementation

```r
library(HDInterval)

set.seed(2026)
theta <- rbeta(10000, 10, 4)

# Equal-tailed
quantile(theta, c(0.025, 0.975))

# HPD
hdi(theta, credMass = 0.95)
```

10.14.6 Output & Results

`quantile()` returns the ETI; `hdi()` returns the HPD. For a Beta(10, 4) posterior the ETI is approximately $(0.50, 0.90)$ and the HPD is approximately $(0.53, 0.93)$ — slightly shifted toward the right because the posterior is left-skewed and the HPD chases the highest density region rather than splitting tail probability evenly.

10.14.7 Interpretation

A reporting sentence: “The posterior probability had a 95 % equal-tailed credible interval of 0.50 to 0.90; the highest-posterior-density interval was 0.53 to 0.93. There is a 95 % posterior probability that the true proportion falls in either

interval, given the prior and the observed data." Always state which interval is reported; mixing definitions across tables in the same paper is a common reproducibility headache.

10.14.8 Practical Tips

- Report ETIs for symmetric or near-symmetric posteriors; they are more familiar and reparameterisation-invariant.
- Report HPDs for skewed posteriors when the additional precision matters and the density estimate is stable.
- HPD intervals can be disjoint under multimodal posteriors; set `allowSplit=TRUE` in `HDInterval::hdi()` so the structure is preserved.
- Disclose the prior in any reporting sentence; the credible interval is not a property of the data alone.
- Use post-warm-up MCMC draws and check ESS in the tails — credible-interval endpoints depend on quantile estimation, which needs more effective draws than the mean.

10.14.9 R Packages Used

`HDInterval` for the canonical `hdi()` function, `bayestestR::ci()` for an integrated wrapper that selects between ETI and HPD on `brms` and `rstanarm` model objects, and base R (`quantile()`) for the equal-tailed case.

10.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.14.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.15 Divergences and Pairs Plots

10.15.1 Introduction

A divergent transition is HMC's most informative warning: it tells you that the leapfrog integrator could not navigate a region of the posterior accurately, and the chain therefore did *not* explore that region. Unlike a high $\hat{R}$, which can sometimes be cured by running longer chains, a divergent transition signals a

structural problem with the posterior geometry that no amount of additional sampling will fix. Diagnosing where divergences occur — typically through pairs plots that colour divergent draws — is the path to a reparameterisation that lets the sampler do its job.

10.15.2 Prerequisites

A working understanding of Hamiltonian Monte Carlo, leapfrog integration, and the geometric origins of divergent transitions; familiarity with `bayesplot::mcmc_pairs()` for visualising joint posteriors.

10.15.3 Theory

The leapfrog integrator approximates Hamiltonian dynamics with finite step size ϵ. In regions of high curvature (typically funnels in hierarchical models, where a small variance parameter shrinks all dependent effects), even a moderate ϵ produces a Hamiltonian error large enough that the proposal is biased. Stan flags such proposals as divergent and refuses to use them, so the chain effectively avoids the problem region instead of exploring it. The classical remedy is a *non-centred* reparameterisation: replacing $\beta_i \sim \mathcal{N}(\mu, \sigma^2)$ with $\tilde{z}_i \sim \mathcal{N}(0,1)$ and $\beta_i = \mu + \sigma\tilde{z}_i$ decouples the funnel geometry and makes the posterior easier to sample.

10.15.4 Assumptions

HMC or NUTS sampling; the divergent-transition flag is specific to these algorithms. Random-walk samplers do not produce divergences but suffer different pathologies (slow mixing, mode-trapping) that need separate diagnostics.

10.15.5 R Implementation

```
library(brms); library(bayesplot)

data(sleepstudy, package = "lme4")

fit <- brm(Reaction ~ Days + (Days | Subject), data = sleepstudy,
           chains = 4, iter = 2000, refresh = 0,
           control = list(adapt_delta = 0.99))

# Divergence diagnostics
rstan::check_hmc_diagnostics(fit$fit)

# Pairs plot of parameters with divergences coloured
np <- nuts_params(fit)
mcmc_pairs(as.array(fit), pars = c("b_Days", "sd_Subject__Days"),
           np = np, np_style = pairs_style_np(div_color = "red"))
```

10.15.6 Output & Results

`check_hmc_diagnostics()` reports the count of divergent transitions, tree-depth saturation events, and the energy Bayesian fraction of missing information (E-BFMI). `mcmc_pairs()` overlays the joint draws of two parameters with red points marking divergences; clustering of red points along the boundary of the parameter space pinpoints the geometry that broke.

10.15.7 Interpretation

A reporting sentence: "An initial fit produced 12 divergent transitions concentrated near small values of the random-slope standard deviation; switching to a non-centred parameterisation and increasing `adapt_delta` to 0.99 eliminated all divergences and yielded $\hat{R} = 1.00$ for every parameter." Report the diagnostic and the fix together; a published Bayesian analysis with unmentioned divergences is not a credible analysis.

10.15.8 Practical Tips

- Any divergence is worth investigating; even a single one signals a region of the posterior the sampler did not explore correctly.
- The classic funnel — variance parameter near zero with correlated dependent effects — is the most common offender; non-centred parameterisation is almost always the right fix.
- Increasing `adapt_delta` to 0.95 or 0.99 reduces step size and eliminates marginal divergences, but it is a band-aid: if the geometry is genuinely difficult, reparameterise.
- `bayesplot::mcmc_parcoord()` complements pairs plots by showing divergent transitions across many parameters simultaneously, which is helpful in models with dozens of variance components.
- When divergences persist after non-centred parameterisation and high `adapt_delta`, examine the prior — improperly tight priors can pinch the posterior into geometry that no sampler can navigate.

10.15.9 R Packages Used

`bayesplot` for `mcmc_pairs()`, `mcmc_parcoord()`, and `nuts_params()`; `rstan` and `cmdstanr` for the underlying diagnostic outputs (`check_hmc_diagnostics()`, `get_sampler_params()`); `posterior` for tidy access to NUTS metadata when building custom diagnostic summaries.

10.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.15.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.16 The Gamma-Poisson Model

10.16.1 Introduction

The Gamma–Poisson conjugate pair is the count-data analogue of the Beta–Binomial model: a Gamma prior on a Poisson rate yields a Gamma posterior whose parameters are obtained by adding the observed event count and the observed exposure to the prior shape and rate respectively. It is the natural starting point whenever you are inferring an event rate per unit time, area, or population, and it generalises directly to Poisson regression once predictors enter the linear predictor.

10.16.2 Prerequisites

A working understanding of the Poisson distribution, the Gamma distribution, and the concept of conjugacy.

10.16.3 Theory

With prior $\lambda \sim \text{Gamma}(\alpha_0, \beta_0)$ and likelihood $y_i \sim \text{Poisson}(\lambda t_i)$ where t_i is the exposure for observation i, the posterior is

$$\lambda \mid y \sim \text{Gamma}\left(\alpha_0 + \sum_i y_i,\ \beta_0 + \sum_i t_i\right).$$

The prior shape α_0 behaves like a pseudo-count of events and the prior rate β_0 like a pseudo-exposure, so a $\text{Gamma}(2, 1)$ prior is equivalent to two prior events observed in one unit of exposure. The posterior mean is $(\alpha_0 + \sum y_i)/(\beta_0 + \sum t_i)$ and the posterior variance is the posterior mean divided by the posterior rate. The Poisson–Gamma compound distribution is a Negative-Binomial, which gives the posterior predictive distribution in closed form.

10.16.4 Assumptions

Independent Poisson counts conditional on the rate, an exposure t_i that is known and not estimated from the same data, and a constant rate over each observation

window (after conditioning on any covariates). Overdispersion relative to Poisson — variance exceeding the mean — motivates a Negative-Binomial likelihood instead.

10.16.5 R Implementation

```r
# Prior Gamma(2, 1) -- 2 events per 1 unit of exposure
alpha0 <- 2; beta0 <- 1

# Data: 15 events in 10 units of exposure
y <- 15; t <- 10

alpha_n <- alpha0 + y
beta_n  <- beta0 + t

c(posterior_mean = alpha_n / beta_n,
  posterior_sd   = sqrt(alpha_n / beta_n^2),
  ci95 = qgamma(c(0.025, 0.975), alpha_n, beta_n))
```

10.16.6 Output & Results

Closed-form summaries: posterior mean approximately 1.55 events per unit exposure, posterior SD approximately 0.38, and 95 % equal-tailed credible interval roughly $(0.94, 2.28)$. The posterior is right-skewed because Gamma densities have a heavier right tail than left, so report the median or HPD when the asymmetry matters.

10.16.7 Interpretation

A reporting sentence: "A Gamma$(2, 1)$ prior combined with 15 events in 10 units of exposure yielded a Gamma$(17, 11)$ posterior with mean 1.55 events per unit exposure (95 % credible interval 0.94 to 2.28); the prior contributed the equivalent of two pseudo-events and one pseudo-unit of exposure, which the data overwhelmed." Quote both the posterior mean and the credible interval, and note any skew.

10.16.8 Practical Tips

- Weakly informative defaults: Gamma$(0.5, 0.5)$ or Gamma$(1, 1)$; the latter has prior mean 1 and prior variance 1.
- For Poisson regression, place priors on regression coefficients on the log-rate scale (typically $\mathcal{N}(0, 1)$ or $\mathcal{N}(0, 2.5)$) rather than on the rate itself; the linear predictor is more naturally constrained.
- The posterior predictive distribution is a Negative-Binomial: $\tilde{y} \mid y \sim \text{NegBinom}(\alpha_n, t_{\text{new}}/(\beta_n + t_{\text{new}}))$, so prediction intervals come for free without simulation.

- Persistent overdispersion after fitting motivates a Negative-Binomial likelihood with a separate dispersion parameter; the Gamma–Poisson is its conjugate root.
- For zero-inflation (more zeros than the Poisson predicts), augment with a zero-inflation component rather than tightening the prior.

10.16.9 R Packages Used

Base R suffices for the conjugate update (`dgamma`, `qgamma`); `bayestestR` for tidy summaries; `brms` with `family=poisson()` (or `negbinomial()`) when covariate-dependent rates are needed; `rstanarm::stan_glm` with `family=poisson` for a quick GLM-style alternative.

10.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.16.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.17 Gibbs Sampling

10.17.1 Introduction

Gibbs sampling is one of the oldest practical MCMC algorithms and remains an instructive special case of more general samplers. It exploits a structural property of multidimensional posteriors: even when the joint distribution is intractable, the conditional distribution of one parameter given all others can often be derived in closed form. Cycling through the parameters and updating each from its full conditional generates a Markov chain whose stationary distribution is the joint posterior, with no acceptance step and no tuning of proposals.

10.17.2 Prerequisites

A working understanding of MCMC, conditional probability, and the conjugate-prior structure that makes full conditionals tractable for canonical models (Normal–Normal, Beta–Binomial, Gamma–Poisson, Dirichlet–Multinomial).

10.17.3 Theory

For a joint posterior $p(\theta_1, \dots, \theta_K \mid y)$, the algorithm initialises at some $\theta^{(0)}$ and, at iteration t, sweeps through the parameters in order:

$$\theta_k^{(t)} \sim p(\theta_k \mid \theta_1^{(t)}, \dots, \theta_{k-1}^{(t)}, \theta_{k+1}^{(t-1)}, \dots, \theta_K^{(t-1)}, y)$$

for $k = 1, \dots, K$. Under standard regularity conditions, the chain is irreducible and aperiodic and converges to the joint posterior. Each update is a draw from a one-dimensional or low-dimensional conditional, often via standard distribution libraries; no Metropolis acceptance is needed because the proposal *is* the conditional. When some conditionals are not available analytically, the affected blocks are updated with a Metropolis step embedded inside the Gibbs sweep ("Metropolis-within-Gibbs").

10.17.4 Assumptions

Every full conditional must be either a recognisable distribution or a distribution one can sample from with a Metropolis step. Strongly correlated parameters update slowly under a coordinate-wise sweep; this is the central pathology of Gibbs and the reason HMC is now preferred for general continuous models.

10.17.5 R Implementation

```r
set.seed(2026)

# Conjugate normal-normal model: mu and sigma2 both unknown
# Priors: mu ~ N(mu0, tau0^2); 1/sigma2 ~ Gamma(alpha, beta)
y <- rnorm(50, 3, 2)
n <- length(y)
mu0 <- 0; tau0 <- 10
alpha <- 2; beta <- 1

n_iter <- 5000
mu_chain    <- numeric(n_iter)
sig2_chain  <- numeric(n_iter)
sig2_chain[1] <- var(y)

for (t in 2:n_iter) {
  # Sample mu | sigma^2, y
  prec <- 1/tau0^2 + n/sig2_chain[t-1]
  mean_mu <- (mu0/tau0^2 + sum(y)/sig2_chain[t-1]) / prec
  mu_chain[t] <- rnorm(1, mean_mu, sqrt(1/prec))

  # Sample sigma^2 | mu, y  (inverse gamma)
  alpha_n <- alpha + n/2
```

```r
  beta_n  <- beta + sum((y - mu_chain[t])^2)/2
  sig2_chain[t] <- 1/rgamma(1, alpha_n, beta_n)
}

# Summaries
c(mean_mu = mean(mu_chain[-(1:1000)]),
  mean_sigma = mean(sqrt(sig2_chain[-(1:1000)])))
```

10.17.6 Output & Results

The chain produces 5,000 draws of μ and σ^2. Discarding the first 1,000 as burn-in and summarising the rest gives posterior means for μ near 3.1 and for σ near 2.0 — close to the data-generating parameters and within the expected Monte Carlo error.

10.17.7 Interpretation

A reporting sentence: "A Gibbs sampler for the conjugate Normal–Inverse-Gamma model produced 4,000 post-burn-in draws with posterior mean $\mu = 3.10$ (95 % credible interval 2.55 to 3.65) and posterior mean $\sigma = 2.01$ (95 % CrI 1.65 to 2.49)." For pedagogical examples, also report the rejection rate (Gibbs has none) and the autocorrelation function so readers can see the per-iteration efficiency.

10.17.8 Practical Tips

- Block-update correlated parameters jointly when the conditional is available in closed form; this reduces autocorrelation dramatically compared with coordinate-wise updates.
- For non-conjugate components, embed a Metropolis step (Metropolis-within-Gibbs); JAGS does this automatically when it cannot resolve a node analytically.
- Check convergence with $\hat{R}$ across multiple chains and trace plots; a single chain cannot reveal mode-trapping.
- For continuous, smooth posteriors of moderate-to-high dimension, HMC (Stan) typically mixes far better than Gibbs because it follows the gradient instead of cycling through coordinates.
- Gibbs remains the algorithm of choice for discrete-state-space models, latent-allocation samplers, and any setting where conjugacy makes a single sweep almost free.

10.17.9 R Packages Used

Base R for hand-coded Gibbs, `rjags` and `nimble` for declarative model specification with automatic Gibbs–Metropolis dispatch, and `coda` for diagnostic plots

and convergence statistics on the resulting chains.

10.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.17.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.18 Hamiltonian Monte Carlo

10.18.1 Introduction

Hamiltonian Monte Carlo (HMC) is the current state of the art for sampling from continuous Bayesian posteriors of moderate-to-high dimension. Where random-walk Metropolis takes small, undirected steps and Gibbs cycles through coordinates, HMC uses gradient information to propose long-distance moves that almost always land in regions of high posterior density. The No-U-Turn Sampler (NUTS), Stan's default, automates the choice of trajectory length so that users do not need to tune step counts by hand. The result is mixing rates that are orders of magnitude better than coordinate-wise samplers for posteriors with strong correlations.

10.18.2 Prerequisites

A working understanding of MCMC, basic calculus (gradients), and the relationship between log-posterior gradients and the geometry of the posterior surface. Familiarity with Stan or `brms` makes the implementation step trivial.

10.18.3 Theory

HMC introduces auxiliary momentum variables p and samples from the joint distribution $p(\theta, p) = p(\theta \mid y)\mathcal{N}(p \mid 0, M)$, where M is the mass matrix. Hamiltonian dynamics conserve the joint Hamiltonian $H(\theta, p) = -\log p(\theta \mid y) + \frac{1}{2}p^T M^{-1} p$, so trajectories simulated with leapfrog integration drift through level sets of constant log-density and explore the posterior efficiently. After L leapfrog steps with step size ϵ, the proposal is accepted with a Metropolis correction that compensates for integrator error. NUTS replaces the manual choice of L with a "no-U-turn" criterion that terminates the trajectory when

it starts doubling back, eliminating the most error-prone tuning parameter of plain HMC.

10.18.4 Assumptions

A continuous, almost-everywhere differentiable log-posterior. Discrete parameters must be marginalised out. The mass matrix should approximate the posterior covariance — Stan adapts it during warm-up — and the step size must be small enough to keep integrator error in check.

10.18.5 R Implementation

```
library(brms)

set.seed(2026)
d <- data.frame(x = rnorm(100))
d$y <- 2 + 1.5 * d$x + rnorm(100)

fit <- brm(y ~ x, data = d,
           chains = 4, iter = 2000, refresh = 0,
           control = list(adapt_delta = 0.95, max_treedepth = 10))

summary(fit)
# Diagnostics: R-hat, divergences, energy
rstan::check_hmc_diagnostics(fit$fit)
```

10.18.6 Output & Results

`summary(fit)` reports posterior summaries with $\hat{R}$, ESS, and credible intervals. `rstan::check_hmc_diagnostics()` reports divergent transitions, tree-depth saturation, energy diagnostics (E-BFMI), and is the first-line check that the geometry was navigable. Healthy HMC delivers $\hat{R}$ near 1.00, no divergent transitions, and ESS that is a substantial fraction of the post-warm-up draws.

10.18.7 Interpretation

A reporting sentence: "HMC/NUTS produced 4,000 post-warm-up draws across four chains with no divergent transitions and $\hat{R} = 1.00$ for every parameter; the slope posterior was 1.52 (95 % credible interval 1.31 to 1.74)." When divergences occur, report them and describe the remedy (reparameterisation, raising `adapt_delta`); a divergent transition is a missed region of the posterior, not a tuning curiosity.

10.18.8 Practical Tips

- For hierarchical models with funnel geometry, switch to non-centred parameterisation; `brms` does this automatically when you specify `prior(..., class="sd",...)`.
- Raising `adapt_delta` to 0.95 or 0.99 reduces divergences at the cost of smaller step sizes and slower sampling; reparameterisation is usually a better fix.
- Tree-depth saturation (`treedepth__` hitting `max_treedepth`) signals that NUTS could not turn around within the budget — increase `max_treedepth` or simplify the model.
- Reparameterise constrained variables onto unconstrained scales when you can: `log(sigma)` rather than `sigma>0` keeps the leapfrog integrator stable.
- HMC scales to thousands of parameters where Gibbs or random-walk MH would be unworkable, but it cannot rescue an ill-conditioned posterior; geometry matters more than algorithm.

10.18.9 R Packages Used

`rstan` and `cmdstanr` as the HMC/NUTS backends, `brms` and `rstanarm` for high-level interfaces, `bayesplot` for diagnostic plots (`mcmc_pairs`, `mcmc_parcoord`) that visualise divergent transitions, and `posterior` for the underlying draw arrays.

10.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.18.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.19 Highest Posterior Density Interval

10.19.1 Introduction

The highest posterior density (HPD) interval is the narrowest range that contains a chosen probability mass — typically 95 % — under a posterior distribution. Two intervals can both cover 95 % of the posterior, yet the HPD will always be at least as short as the equal-tailed interval (ETI) and will include only those parameter values whose density is above some threshold. For unimodal posteriors the two intervals are similar; for skewed or bimodal posteriors

they can disagree dramatically, and the choice between them shapes how readers perceive the result.

10.19.2 Prerequisites

A working understanding of posterior distributions and credible intervals, and access to MCMC draws (via `brms`, `rstanarm`, raw Stan, or any other sampler).

10.19.3 Theory

For a posterior $p(\theta \mid y)$, the $(1-\alpha)$ HPD region is

$$\{\theta : p(\theta \mid y) \geq h\}, \quad \text{with } h \text{ such that } P(\theta \in \text{region}) = 1-\alpha.$$

For a unimodal posterior this region is a single interval centred near the mode; for a multimodal posterior it can be a union of disjoint intervals, one per mode that exceeds the threshold. The ETI, by contrast, places the same tail probability $\alpha/2$ in each tail, so for a skewed posterior it shifts toward the heavier tail and is wider. HPD therefore favours the shape of the posterior; ETI favours interpretability and invariance to monotone transformations.

10.19.4 Assumptions

Adequate MCMC draws (after warm-up, with $\hat{R}$ near 1.00) or a closed-form posterior density. The HPD is sensitive to noisy density estimates in the tails, so very short chains can yield unstable interval endpoints.

10.19.5 R Implementation

```r
library(HDInterval)

set.seed(2026)
theta <- c(rbeta(5000, 5, 15), rbeta(5000, 15, 5))  # bimodal

# Equal-tailed (may span both modes)
quantile(theta, c(0.025, 0.975))

# HPD (disjoint regions)
hdi(theta, credMass = 0.95, allowSplit = TRUE)
```

10.19.6 Output & Results

`hdi()` returns a numeric vector of two endpoints when the HPD is a single interval, or a matrix of begin / end pairs when `allowSplit=TRUE` and the posterior is multimodal. `quantile()` returns the ETI for direct comparison; the gap

between ETI and HPD widths is the most direct numerical signal of posterior asymmetry.

10.19.7 Interpretation

A reporting sentence: "The 95 % HPD interval for θ was $(0.13, 0.38) \cup (0.60, 0.89)$, reflecting the bimodal posterior; the equal-tailed interval $(0.13, 0.89)$ obscures this structure by including the low-density valley between modes." When the posterior is unimodal and roughly symmetric, the two intervals will agree and the choice does not matter; when they disagree, prefer the one whose definition matches the inferential question — HPD for "where does the posterior concentrate?" and ETI for "what range of values is at most 2.5 % above / below the truth?"

10.19.8 Practical Tips

- Set `allowSplit=TRUE` whenever multimodality is plausible; a single interval reported across modes can be misleading.
- HPD is *not* invariant under non-linear reparameterisation: the HPD for σ^2 is not in general the square of the HPD for σ, while the ETI quantile-based interval is. Decide on the reporting scale before computing the interval.
- For bounded parameters (probabilities, variances) the HPD often touches the boundary; report this explicitly so readers know the posterior is not symmetric there.
- Use post-warm-up MCMC draws and check that the chains have mixed; HPD endpoints are noisier than means, so bigger effective sample sizes help.
- `bayestestR::hdi()` provides a similar interface and integrates with model objects from `brms`, `rstanarm`, and `posterior`.

10.19.9 R Packages Used

`HDInterval` for the canonical `hdi()` function with `allowSplit`, `bayestestR` for an integrated wrapper around model objects, and `posterior` for handling MCMC draw arrays consistently across backends.

10.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.19.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.20 Informative Priors

10.20.1 Introduction

An informative prior encodes genuine pre-data knowledge: a quantitative summary of previous studies, a mechanistic constraint that any plausible parameter must respect, or a structured elicitation from domain experts. Done well, informative priors stabilise inference in small samples, integrate evidence across studies, and let a single analysis quantify how new data update what was already known. Done badly, they paper over weak data and make conclusions look more certain than they should. The discipline is to be explicit about the source, justify the choice quantitatively, and run a sensitivity analysis whenever the prior is doing meaningful work.

10.20.2 Prerequisites

A working understanding of Bayesian inference, the role of weakly informative priors as a default, and basic familiarity with meta-analytic summaries on the relevant effect scale.

10.20.3 Theory

Three classical sources of informative priors:

- **Meta-analytic priors** condense effect estimates from previous trials into a Normal (or log-Normal, or t) prior on the relevant scale: $\beta \sim \mathcal{N}(\bar{\beta}, \tau^2)$ where $\bar{\beta}$ and τ^2 come from a published random-effects meta-analysis.
- **Mechanistic priors** are derived from physical, biological, or chemical constraints; for example, drug half-lives are positive and bounded above by metabolic limits, so a log-Normal or truncated prior is appropriate.
- **Expert-elicited priors** capture structured beliefs from clinicians or subject-matter specialists, typically via the SHELF protocol or a similar formal procedure to reduce the influence of a single dominant expert.

A power prior or commensurate prior down-weights the prior contribution when it conflicts with current data — a principled way to guard against unverified historical assumptions.

10.20.4 Assumptions

The prior information is relevant to the current analysis (same population, same outcome definition, same scale) and is not derived from the same data being

analysed (no double-counting). Mechanistic constraints must be physically defensible, not stylistic.

10.20.5 R Implementation

```r
library(brms)

# Meta-analytic prior: previous studies gave OR of 0.8 (95% CI 0.7 to 0.9)
# On log-OR scale: mean -0.223, SD ~ 0.05 (narrow informative prior)
set.seed(2026)
d <- data.frame(x = rnorm(100), y = rbinom(100, 1, 0.3))

fit_inf <- brm(y ~ x, data = d, family = bernoulli,
               prior = prior(normal(-0.223, 0.05), class = "b", coef =
                ↪  "x"),
               chains = 4, iter = 2000, refresh = 0)

# Compare with weakly informative
fit_weak <- brm(y ~ x, data = d, family = bernoulli,
                prior = prior(normal(0, 2.5), class = "b"),
                chains = 4, iter = 2000, refresh = 0)

# Posterior summaries
fixef(fit_inf); fixef(fit_weak)
```

10.20.6 Output & Results

`fixef(fit_inf)` returns the posterior mean and credible interval under the meta-analytic prior; `fixef(fit_weak)` does the same under a weakly informative prior. The two posteriors differ in proportion to the relative precision of the prior and the data: with $\mathcal{N}(-0.223, 0.05)$ the prior carries the weight of roughly $1/0.05^2 = 400$ "pseudo-observations" on the log-odds scale, dominating the small simulated dataset.

10.20.7 Interpretation

A reporting sentence: "Using a meta-analytic prior on the log-odds ratio derived from three previous trials ($\mathcal{N}(-0.223, 0.05^2)$, equivalent to roughly 400 pseudo-observations), the posterior estimate was -0.19 (95 % credible interval -0.28 to -0.10); under a weakly informative $\mathcal{N}(0, 2.5)$ prior, the posterior was -0.04 (95 % CrI -0.43 to 0.34)." Reporting both posteriors lets readers see exactly how much of the conclusion is driven by the prior.

10.20.8 Practical Tips

- Document the source of every informative prior in the methods section: cite the meta-analysis, describe the elicitation protocol, or state the mechanistic constraint and its justification.
- Sensitivity analyses are mandatory whenever an informative prior is used; report the posterior under at least one weakly informative alternative.
- Power priors with discount factor $a_0 \in [0, 1]$ scale the prior likelihood and let the analyst choose how strongly historical data should influence the current analysis.
- Commensurate priors borrow more aggressively when historical and current studies are similar and back off when they conflict; they are particularly useful in regulatory submissions.
- For elicited priors, use formal tools such as the SHELF protocol; ad-hoc elicitation tends to underestimate uncertainty and overstate consensus.
- In regulatory contexts (FDA, EMA), prior choice often requires prespecification and agency agreement; budget time for that conversation before running the analysis.

10.20.9 R Packages Used

`brms` and `rstanarm` for placing arbitrary priors on regression parameters; `RBesT` for meta-analytic predictive priors and effective sample size calculations; `SHELF` for expert elicitation; `BayesPPD` for power-prior workflows in regulatory settings.

10.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.20.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.21 Jeffreys' Prior

10.21.1 Introduction

Jeffreys' prior is the classical "objective" or "reference" prior in Bayesian inference. Defined as proportional to the square root of the Fisher information, it has the elegant property of being invariant under monotone reparameterisation: the prior on a transformation $\eta = g(\theta)$ derived from the Jeffreys prior

on θ equals the Jeffreys prior derived directly for η. This invariance — absent from a flat prior, which is parameterisation-dependent — is what historically made Jeffreys' rule the default attempt at a non-informative prior. In modern Bayesian practice, however, weakly informative priors have largely displaced it because they regularise small-sample inference more reliably and behave better in hierarchical models.

10.21.2 Prerequisites

A working understanding of Fisher information, the role of reparameterisation in inference, and the difference between proper and improper priors.

10.21.3 Theory

For a scalar parameter θ with likelihood $p(y \mid \theta)$, the Jeffreys prior is

$$p(\theta) \propto \sqrt{I(\theta)}, \qquad I(\theta) = -\mathbb{E}\left[\frac{\partial^2 \log p(y \mid \theta)}{\partial \theta^2}\right].$$

Standard examples:

- Binomial(n, θ): $I(\theta) \propto 1/[\theta(1-\theta)]$, so the Jeffreys prior is Beta$(\frac{1}{2}, \frac{1}{2})$.
- Poisson(λ): $I(\lambda) \propto 1/\lambda$, so $p(\lambda) \propto \lambda^{-1/2}$.
- Normal mean (known variance): flat prior $p(\mu) \propto 1$.
- Normal scale (known mean): $p(\sigma) \propto 1/\sigma$.

Jeffreys priors are often improper, but the posterior is usually proper as long as the sample size is sufficient. For multivariate parameters, $I(\theta)$ is a matrix and the prior is $\sqrt{|I(\theta)|}$.

10.21.4 Assumptions

A regular likelihood (twice differentiable in θ, with finite Fisher information across the parameter space). Boundary problems and parameter-dimension growth complicate the construction; reference priors (Bernardo) generalise Jeffreys' rule for hierarchical and high-dimensional cases.

10.21.5 R Implementation

```r
# Binomial: posterior under Jeffreys is Beta(1/2 + x, 1/2 + n - x)
n <- 10; x <- 8
post_a <- 0.5 + x
post_b <- 0.5 + n - x
c(mean = post_a / (post_a + post_b),
  ci95 = qbeta(c(0.025, 0.975), post_a, post_b))
```

```r
# Compare with uniform Beta(1, 1) and Beta(2, 2)
data.frame(
  prior = c("Uniform(0,1)", "Jeffreys", "Beta(2,2)"),
  post_mean = c((1 + x)/(2 + n), (0.5 + x)/(1 + n), (2 + x)/(4 + n))
)
```

10.21.6 Output & Results

Closed-form summaries under three priors. With 8 successes in 10 trials the Jeffreys posterior mean is approximately 0.77 (vs. 0.75 under the uniform Beta$(1, 1)$ and 0.71 under the more informative Beta$(2, 2)$), and the 95 % credible interval is roughly $(0.47, 0.94)$. The comparison highlights how mildly Jeffreys differs from a flat prior in this setting.

10.21.7 Interpretation

A reporting sentence: "Jeffreys' prior Beta$(0.5, 0.5)$ on the success probability combined with 8 successes in 10 trials yielded a posterior mean of 0.77 (95 % credible interval 0.47 to 0.94); the result is essentially equivalent to a uniform prior at this sample size, justifying the reference-prior framing." When Jeffreys differs materially from a weakly informative prior, the prior is doing real work and you should be explicit about why you chose it.

10.21.8 Practical Tips

- The invariance property is mathematically attractive but in practice rarely affects applied conclusions; weakly informative priors are a more flexible default in modern workflows.
- Jeffreys priors can behave badly in hierarchical models — the multivariate version often produces improper posteriors when variance components are at the boundary. Reach for weakly informative half-Cauchy or half-Normal priors instead.
- Always check that the resulting posterior is proper, especially when the prior itself is improper; a back-of-the-envelope check is to confirm that the posterior integrates to a finite quantity.
- Reference priors (Bernardo) generalise Jeffreys' rule for ordered parameters of interest and nuisance components; they are the modern theoretical successor when an objective prior is genuinely required.
- For pedagogical purposes, Jeffreys' prior is invaluable because it makes invariance under reparameterisation a hands-on exercise; for applied analyses, prefer weakly informative defaults unless invariance is a publication requirement.

10.21.9 R Packages Used

Base R suffices for closed-form Jeffreys updates in conjugate models; `BayesianTools` and `mcmc` allow Jeffreys priors via custom log-prior functions; `bayestestR::point_estimate()` and `HDInterval::hdi()` deliver tidy summaries on the resulting posteriors.

10.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.21.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.22 Bayesian Linear Regression

10.22.1 Introduction

Bayesian linear regression keeps the same linear predictor and Normal residual structure as ordinary least squares but treats every parameter as a random variable with a prior. Inference returns a joint posterior over coefficients and the residual scale rather than a point estimate plus standard errors. The practical gain is direct: credible intervals carry the probability statement that students often (incorrectly) attribute to confidence intervals, predictions inherit propagated uncertainty automatically, and small-sample analyses no longer rely on asymptotic approximations of t distributions.

10.22.2 Prerequisites

A working understanding of multiple linear regression, comfort with the idea of a likelihood, and a basic sense of priors. A Stan-capable backend (`rstan` or `cmdstanr`) is needed because `brms` compiles models on demand; `rstanarm` ships with pre-compiled models if compile time is a concern.

10.22.3 Theory

The model is

$$y_i = X_i\beta + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2),$$

with priors $\beta_j \sim \mathcal{N}(0, \tau_j)$ and $\sigma \sim$ Exponential(λ) or a half-t. With weakly informative priors and reasonable sample size, the posterior mean is close to the OLS estimate while small-n inference is regularised toward the prior. The default sampler is Hamiltonian Monte Carlo with the No-U-Turn termination criterion (NUTS); convergence is assessed through $\hat{R}$, bulk and tail effective sample size, and divergent-transition counts.

10.22.4 Assumptions

Linearity in the linear predictor, additive Normal residuals with constant variance, independence of observations, and proper priors. Convergence diagnostics are part of the model assumptions in practice — a chain that has not mixed cannot deliver a credible posterior summary.

10.22.5 R Implementation

```r
library(brms)

set.seed(2026)
d <- data.frame(x = rnorm(100), y = 2 + 1.5 * rnorm(100))

fit <- brm(y ~ x, data = d,
           prior = prior(normal(0, 2.5), class = "b") +
                   prior(normal(0, 5),   class = "Intercept") +
                   prior(exponential(1), class = "sigma"),
           chains = 4, iter = 2000, refresh = 0)

summary(fit)
plot(fit)
```

10.22.6 Output & Results

`summary(fit)` returns posterior mean, standard error of the mean, posterior standard deviation, 95 % credible interval, $\hat{R}$, and bulk and tail effective sample size for every parameter. `plot(fit)` shows trace plots (one line per chain) alongside marginal density plots; well-mixed chains overlap throughout warm-up and produce indistinguishable density plots.

10.22.7 Interpretation

A reporting sentence: "A Bayesian linear regression with weakly informative priors estimated the slope at 1.53 (95 % CrI 1.23 to 1.84) and the residual scale at 1.02 (95 % CrI 0.90 to 1.17); $\hat{R} = 1.00$ for all parameters, with $n_{\text{eff}} > 3{,}500$." Always report a convergence statistic alongside the interval — without it, a reader cannot tell whether the posterior summary is trustworthy.

10.22.8 Practical Tips

- Standardise predictors before fitting so a single $\mathcal{N}(0, 2.5)$ prior on slopes is sensible across columns.
- Run a prior predictive check (`sample_prior="only"`) and look at the implied range of `y`; if it spans implausible values, tighten the prior before fitting to the data.
- For interactions or polynomials, use `bs()` or `poly()` inside the formula but check that the prior scale still makes sense after the transformation.
- Use `bayesplot::pp_check(fit)` to overlay replicated datasets on the observed outcome; systematic deviation in the tails signals the wrong family.
- Compare nested models with `loo_compare(loo(fit1),loo(fit2))` rather than F-tests; the difference in expected log predictive density and its standard error is a richer summary than a p-value.
- For non-Gaussian outcomes, change `family=gaussian` to `bernoulli()`, `poisson()`, `student()`, or any other supported family — the prior structure carries over unchanged.

10.22.9 R Packages Used

`brms` for formula-based Bayesian regression, `rstan` or `cmdstanr` as the sampling backend, `bayesplot` for diagnostic and posterior predictive plots, and `loo` for cross-validation.

10.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.22.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.23 Bayesian Logistic Regression

10.23.1 Introduction

Bayesian logistic regression keeps the same Bernoulli likelihood and logit link as the maximum-likelihood version but replaces the point estimate of each coefficient with a full posterior distribution. The biggest practical pay-off is robustness in small samples or in the presence of complete or quasi-complete separation, where MLE drifts to $\pm\infty$ but a weakly informative prior pulls the estimate to a

finite, defensible value. The posterior also feeds directly into propagated uncertainty for predicted probabilities, an awkward computation in the frequentist world.

10.23.2 Prerequisites

Familiarity with classical logistic regression and odds ratios, an idea of what a prior and a posterior are, and access to a Stan-capable backend (`rstan` or `cmdstanr`).

10.23.3 Theory

The model is

$$y_i \sim \text{Bernoulli}(\text{logit}^{-1}(X_i\beta)),$$

with priors $\beta \sim \mathcal{N}(0, \sigma_\beta^2)$ on the regression coefficients and a separate, often wider prior on the intercept because the average log-odds of success can lie outside the typical predictor scale. A $\mathcal{N}(0, 2.5)$ prior on standardised slopes is the default in `rstanarm` and a sensible starting point in `brms`. Under separation the likelihood becomes monotone in β, and any proper prior — even a very wide one — yields a well-defined posterior with finite credible intervals; this is the Bayesian analogue of Firth's penalised likelihood.

10.23.4 Assumptions

Independent Bernoulli observations conditional on covariates, a logit link that is a reasonable approximation to the true response surface, and valid (proper) priors. As in MLE logistic regression, the linear predictor should not contain redundant columns.

10.23.5 R Implementation

```
library(brms)

set.seed(2026)
d <- data.frame(x = rnorm(200))
d$y <- rbinom(200, 1, plogis(-1 + 1.2 * d$x))

fit <- brm(y ~ x, data = d, family = bernoulli,
           prior = prior(normal(0, 2.5), class = "b") +
                   prior(normal(0, 5), class = "Intercept"),
           chains = 4, iter = 2000, refresh = 0)
summary(fit)
```

```
# Odds ratios
exp(fixef(fit))
```

10.23.6 Output & Results

`summary(fit)` reports posterior mean, standard error, and 95 % CrI on the log-odds scale together with $\hat{R}$ and effective sample size; `exp(fixef(fit))` returns the same summaries on the odds-ratio scale. `posterior_predict()` delivers Bernoulli replicates that feed `pp_check()` to test calibration.

10.23.7 Interpretation

A reporting sentence: "A Bayesian logistic regression with weakly informative $\mathcal{N}(0, 2.5)$ priors estimated the odds ratio for x at 3.4 (95 % CrI 2.5 to 4.5); under the model, a one-standard-deviation increase in x triples the odds of success." When predicted probabilities are required, summarise `posterior_epred()` rather than transforming `fixef()` by hand — only the former propagates the joint uncertainty in intercept and slope.

10.23.8 Practical Tips

- For binary regression with rare outcomes, prefer a $\mathcal{N}(0, 1)$ prior on standardised slopes; the default $\mathcal{N}(0, 2.5)$ allows odds ratios up to about $\exp(5) \approx 150$, which can be implausibly extreme.
- Compare predictive performance with `loo()` rather than likelihood-ratio tests; LOO is well-defined under proper priors and gives Pareto-k diagnostics for influential points.
- Hierarchical extensions are a one-line addition: `y~x+(1|site)` lets baseline log-odds vary across sites and shrinks small-cluster estimates toward the population mean.
- For severe separation, fit with `prior(normal(0,1.5),class="b")` rather than ML's Firth correction; the Bayesian solution is a single line of code and propagates uncertainty consistently.
- Always inspect `pp_check(fit,type="bars")` for binary outcomes; deviations between observed and replicated proportions reveal calibration problems that summary tables hide.

10.23.9 R Packages Used

`brms` for model specification, `bayesplot` for posterior predictive plots, `loo` for cross-validation and model comparison, and `tidybayes` if you want tidy tibbles of posterior draws for downstream plotting.

10.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.23.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.24 Leave-One-Out Cross-Validation

10.24.1 Introduction

Leave-one-out cross-validation (LOO-CV) is the gold standard for estimating out-of-sample predictive performance, but exact LOO requires refitting the model once per observation — prohibitive for any Bayesian model that takes more than a few seconds to sample. Pareto-Smoothed Importance Sampling LOO (PSIS-LOO), introduced by Vehtari, Gelman, and Gabry, approximates exact LOO from a single full-data posterior using importance weights, while providing per-observation diagnostics that flag where the approximation breaks down. It has become the standard model-comparison tool for `brms`, `rstanarm`, and raw Stan workflows.

10.24.2 Prerequisites

A working understanding of cross-validation, importance sampling, and the role of the log pointwise predictive density (elpd) in measuring predictive performance.

10.24.3 Theory

The expected log predictive density under leave-one-out is

$$\mathrm{elpd}_{\mathrm{loo}} = \sum_{i=1}^{n} \log p(y_i \mid y_{-i}),$$

where y_{-i} is the data with observation i removed. PSIS approximates each $p(y_i \mid y_{-i})$ via importance weights $w_i^{(s)} \propto 1/p(y_i \mid \theta^{(s)})$ applied to the full-data posterior draws $\theta^{(s)}$. To stabilise the right tail of the importance weights, PSIS fits a generalised Pareto distribution to the largest weights and replaces them

with order-statistic estimates. The shape parameter $\hat{k}$ of that fit acts as a per-observation diagnostic: $\hat{k} < 0.5$ is excellent, $0.5 \leq \hat{k} < 0.7$ acceptable, and $\hat{k} > 0.7$ indicates that PSIS-LOO is unreliable for that point.

10.24.4 Assumptions

Exchangeable observations and a posterior with sufficient effective sample size to estimate the importance-weight tail; for hierarchical or temporal data, replace standard LOO with leave-one-cluster-out or leave-future-out variants whose theoretical properties match the dependence structure.

10.24.5 R Implementation

```
library(loo); library(brms)

data(mtcars)
fit1 <- brm(mpg ~ wt, data = mtcars, chains = 2, iter = 2000, refresh = 0)
fit2 <- brm(mpg ~ wt + hp, data = mtcars, chains = 2, iter = 2000,
 ↪  refresh = 0)

loo1 <- loo(fit1)
loo2 <- loo(fit2)

loo_compare(loo1, loo2)

# k-hat diagnostic
plot(loo2)
```

10.24.6 Output & Results

`loo()` returns the elpd, the effective parameter count p_{loo}, and a vector of per-observation $\hat{k}$ values. `loo_compare()` aligns two `loo` objects and returns the elpd difference with its standard error, ordered from best to worst. `plot(loo)` displays $\hat{k}$ values against observation index, immediately revealing influential points.

10.24.7 Interpretation

A reporting sentence: "PSIS-LOO favoured the two-predictor model with $\Delta\text{elpd} = 6$ (SE 2.3); all $\hat{k} < 0.5$, so the importance-sampling approximation is reliable, and no observation needs an exact refit." When some $\hat{k}$ values exceed 0.7, rerun with `loo(fit,reloo=TRUE)` to refit the model on the affected leave-out folds; the elpd estimate is then a hybrid of PSIS for safe folds and exact fits for the troublesome ones.

10.24.8 Practical Tips

- Check $\hat{k}$ values *before* interpreting an elpd comparison; a single high-leverage observation can dominate the ranking.
- For time-series, use leave-future-out CV via `loo::loo_subsample()` or `loo_moment_match()`; standard LOO leaks future information into past folds and biases elpd downward.
- For hierarchical models with few clusters, leave-one-cluster-out CV is the appropriate generalisation: it answers the prediction question that matters in practice ("how would the model perform on a new cluster?").
- Differences in elpd are interpretable in absolute terms: $\Delta\text{elpd} > 4 \times \text{SE}$ is decisive; $|\Delta\text{elpd}| < 2 \times \text{SE}$ is inconclusive.
- LOO is well-defined for non-nested models, unlike likelihood-ratio tests, which makes it the natural tool when comparing models that differ in family, link, or predictor set.

10.24.9 R Packages Used

`loo` for `loo()`, `loo_compare()`, `loo_subsample()`, and `loo_moment_match()`, `brms` and `rstanarm` for fits that expose pointwise log-likelihoods automatically, and `bayesplot` for posterior predictive plots that complement the elpd-based comparison.

10.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.24.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.25 Metropolis-Hastings

10.25.1 Introduction

The Metropolis-Hastings algorithm is the original general-purpose MCMC sampler and remains the conceptual reference against which all subsequent algorithms are explained. Given only an unnormalised target density, it produces a Markov chain whose stationary distribution is that target by repeatedly proposing a candidate value and accepting or rejecting it based on a simple density-ratio rule. The trade-off is universality versus efficiency: MH works for any

target one can evaluate, but tuning proposals for high-dimensional or strongly correlated posteriors becomes painful, which is why HMC has displaced random-walk MH for most modern Bayesian software.

10.25.2 Prerequisites

A working understanding of Markov chains, conditional probability, and the role of the unnormalised posterior in Bayesian inference. Familiarity with R's distribution functions (`rnorm`, `dunif`) is needed for the example.

10.25.3 Theory

Given the current state θ, one iteration of MH proceeds as:

1. Propose θ^{*} from a proposal kernel $q(\theta^{*} \mid \theta)$.
2. Compute the acceptance probability $\alpha = \min\left(1, \dfrac{p(\theta^{*})q(\theta \mid \theta^{*})}{p(\theta)q(\theta^{*} \mid \theta)}\right)$.
3. With probability α set the next state to θ^{*}; otherwise keep θ.

For a symmetric proposal (Gaussian random walk centred at the current state) the proposal terms cancel and the acceptance ratio is simply the ratio of target densities. Detailed balance ensures the chain has the target as its invariant distribution; ergodicity (irreducibility plus aperiodicity) ensures convergence regardless of the starting value.

10.25.4 Assumptions

The unnormalised target density is computable everywhere in the support. The proposal kernel must give the chain a positive probability of reaching every part of the support; otherwise the chain is reducible and never explores the full posterior.

10.25.5 R Implementation

```
set.seed(2026)

# Target: Beta(4, 8) (unnormalised density is fine)
log_target <- function(x) {
  if (x <= 0 || x >= 1) return(-Inf)
  (4 - 1) * log(x) + (8 - 1) * log(1 - x)
}

n_iter <- 10000
theta <- numeric(n_iter)
theta[1] <- 0.5
sigma_prop <- 0.1
```

```r
for (i in 2:n_iter) {
  prop <- rnorm(1, theta[i-1], sigma_prop)
  log_ratio <- log_target(prop) - log_target(theta[i-1])
  if (log(runif(1)) < log_ratio) theta[i] <- prop
  else theta[i] <- theta[i-1]
}

# Diagnostic: trace + histogram
par(mfrow = c(1, 2))
plot(theta, type = "l")
hist(theta[-(1:1000)], breaks = 50, freq = FALSE, main = "Posterior
 ↪  samples")
curve(dbeta(x, 4, 8), add = TRUE, col = "red", lwd = 2)
par(mfrow = c(1, 1))

mean(theta[-(1:1000)])     # posterior mean
```

10.25.6 Output & Results

A 10,000-step chain whose post-burn-in histogram aligns with the analytic Beta(4, 8) density. The trace plot, the histogram, and the posterior mean (around $4/(4+8) = 0.33$) confirm convergence; the autocorrelation function tells you how many independent draws the chain is worth.

10.25.7 Interpretation

A reporting sentence: "A random-walk Metropolis-Hastings sampler with proposal $\mathcal{N}(\theta^{(t)}, 0.1^2)$ produced 9,000 post-burn-in draws; the histogram tracked the analytic Beta(4, 8) target, and the posterior mean of 0.33 matched the closed-form value to within Monte Carlo error." Quote the acceptance rate so readers can judge whether the proposal was tuned reasonably.

10.25.8 Practical Tips

- Tune the proposal scale for an acceptance rate near 25–40 % for one-dimensional random walks (Roberts–Gelman–Gilks rule); for higher dimensions, lower rates are optimal.
- Discard a burn-in of 1,000–5,000 iterations and inspect the trace plot for visible drift before computing summaries.
- When parameters are correlated, a coordinate-wise random walk has poor mixing; switch to block updates with a multivariate Normal proposal whose covariance approximates the posterior, or move to HMC.
- Use $\hat{R}$ across multiple chains rather than relying on a single trace; a single chain cannot reveal mode-trapping.
- For tall posteriors with many parameters, modern HMC samplers (Stan,

NUTS) almost always mix dramatically better than basic MH; reach for MH only when the target is hard to differentiate or has discrete components.

10.25.9 R Packages Used

Base R is enough for hand-coded MH; `MCMCpack` and `mcmc` provide tuned implementations for canonical models, and `coda` contributes diagnostic plots and convergence statistics applicable to MH chains from any source.

10.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.25.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.26 The Normal-Normal Model

10.26.1 Introduction

The Normal-Normal model is the simplest example of conjugate Bayesian updating for a continuous parameter: a Normal prior on the mean, a Normal likelihood with known variance, and a Normal posterior obtained by a transparent precision-weighted average. It is the classical pedagogical example, and despite its simplicity it sits at the heart of every hierarchical model that contains a Normal level — random-effects meta-analysis, partial pooling in mixed models, and the mean structure of Gaussian processes all rely on the same closed-form update.

10.26.2 Prerequisites

A working understanding of the Normal distribution, the concept of conjugacy, and the difference between variance and precision (inverse variance). Basic algebra is enough to follow the derivation.

10.26.3 Theory

With prior $\mu \sim \mathcal{N}(\mu_0, \tau_0^2)$ and likelihood $y_i \sim \mathcal{N}(\mu, \sigma^2)$ for $i = 1, \ldots, n$ with known σ, the posterior is

$$\mu \mid y \sim \mathcal{N}(\mu_n, \tau_n^2),$$

where the posterior precision is the sum of the prior precision and the data precision,

$$\tau_n^{-2} = \tau_0^{-2} + n/\sigma^2,$$

and the posterior mean is the precision-weighted average

$$\mu_n = \tau_n^2 \left(\tau_0^{-2}\mu_0 + n\bar{y}/\sigma^2\right).$$

The posterior mean lies between the prior mean and the sample mean, with weight that depends on relative precisions. The precision-additive structure makes the prior's "pseudo-sample-size" interpretable: τ_0^2 corresponds to a prior sample of size σ^2/τ_0^2.

10.26.4 Assumptions

Known σ^2 — so the only unknown is the mean — and a Normal prior. If the variance is also unknown, the conjugate setup uses a Normal-Inverse-Gamma prior and the posterior on μ is a t distribution.

10.26.5 R Implementation

```r
# Prior: mu ~ N(100, 10^2); data: xbar = 105, sigma = 15, n = 20
mu0 <- 100; tau0 <- 10
xbar <- 105; sigma <- 15; n <- 20

prec_post <- 1/tau0^2 + n/sigma^2
tau_post  <- 1/sqrt(prec_post)
mu_post   <- tau_post^2 * (mu0/tau0^2 + n*xbar/sigma^2)

c(posterior_mean = mu_post,
  posterior_sd   = tau_post,
  ci95 = mu_post + c(-1, 1) * 1.96 * tau_post)
```

10.26.6 Output & Results

Posterior mean approximately 104.2 with posterior standard deviation 2.4 and 95 % credible interval roughly $(99.6, 108.9)$. The shrinkage from the sample mean of 105 toward the prior mean of 100 is small because the data precision ($n/\sigma^2 \approx 0.089$) is much larger than the prior precision ($1/\tau_0^2 = 0.01$).

10.26.7 Interpretation

A reporting sentence: "Combining a $\mathcal{N}(100, 10^2)$ prior with 20 observations of sample mean 105 (known $\sigma = 15$) yielded a posterior mean of 104.2 (95 % credible interval 99.6 to 108.9); the prior contributed the equivalent of approximately 2.25 additional observations." The "equivalent observations" framing is one of the most intuitive ways to communicate prior strength to non-Bayesians.

10.26.8 Practical Tips

- As $n \to \infty$, the posterior mean tends to the sample mean and the posterior variance tends to σ^2/n; the prior's influence vanishes asymptotically but can dominate in small samples.
- When σ is unknown, switch to the Normal–Inverse-Gamma conjugate pair; the marginal posterior on μ then has t tails.
- The precision-form update generalises directly to multivariate Normals with covariance matrices in place of variances; everything that was scalar becomes matrix.
- Use this model as a sanity check for software: any general-purpose Bayesian engine (Stan, JAGS, `brms`) should reproduce the closed-form posterior to within Monte Carlo error.
- Hierarchical mean structures are built from this update at every level, which is why understanding it deeply pays dividends in mixed models and meta-analysis.

10.26.9 R Packages Used

Base R is sufficient because the update is closed-form (`dnorm`, `pnorm`, `qnorm`); for didactic plotting compare prior, likelihood, and posterior with `ggplot2`, and for general Bayesian alternatives any of `brms`, `rstanarm`, or `BayesFactor` reproduce the same answer.

10.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.26.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.27 Posterior Mean and Variance

10.27.1 Introduction

A Bayesian analysis returns a full posterior distribution, but applied work eventually demands a number to put in a sentence: a coefficient, a probability, an effect size. The posterior mean, median, and mode are the three most common point summaries, each motivated by a different decision-theoretic loss. The posterior variance and standard deviation quantify the uncertainty around the chosen point. Treating the posterior as the headline and the point estimate as a one-line summary derived from it is a more honest reporting style than collapsing the posterior to a single number first and worrying about uncertainty later.

10.27.2 Prerequisites

A working understanding of Bayes' theorem and the difference between a posterior distribution and a frequentist sampling distribution; basic familiarity with MCMC draws or a closed-form posterior.

10.27.3 Theory

For a posterior $p(\theta \mid y)$, the standard summaries are:

- **Posterior mean**: $\mathbb{E}[\theta \mid y]$ minimises expected squared-error loss; it is the Bayes estimator under quadratic loss and the natural target of MCMC averaging.
- **Posterior median**: minimises expected absolute-error loss; preferred for skewed posteriors because it does not chase a long tail.
- **Posterior mode (MAP)**: maximises $p(\theta \mid y)$ and often coincides with a penalised maximum-likelihood estimate when the prior is interpreted as a penalty.
- **Posterior variance**: $\mathrm{Var}(\theta \mid y)$ measures spread; the standard deviation $\sqrt{\mathrm{Var}}$ is on the original scale of θ.

The mean and variance always exist for proper integrable posteriors; the mode may not be unique under multimodal posteriors and the median is unique only for continuous distributions.

10.27.4 Assumptions

A proper posterior with finite mean and variance, and either closed-form expressions or sufficient MCMC draws. For heavy-tailed posteriors, the mean and variance can be unstable estimators even when they exist mathematically, so report effective sample sizes alongside the summaries.

10.27.5 R Implementation

```r
set.seed(2026)

# Simulated samples from a posterior (e.g., from MCMC)
theta <- rbeta(5000, shape1 = 10, shape2 = 4)

c(mean = mean(theta),
  median = median(theta),
  mode = density(theta)$x[which.max(density(theta)$y)],
  sd = sd(theta),
  var = var(theta))

# Credible interval
quantile(theta, c(0.025, 0.975))
```

10.27.6 Output & Results

Five scalar summaries and a 95 % equal-tailed credible interval. For a Beta(10, 4) posterior the mean is approximately $10/14 \approx 0.71$, the standard deviation is around 0.11, and the 95 % interval excludes the lower decile of the distribution, communicating both the central tendency and the spread.

10.27.7 Interpretation

A reporting sentence: "The posterior probability had mean 0.71 (posterior SD = 0.11, 95 % credible interval 0.47 to 0.89), placing high credibility on values above 0.5 and effectively excluding the prior-favoured region near 0.3." When the posterior is roughly symmetric the choice of mean versus median rarely matters; when it is skewed, report both and let readers see the asymmetry.

10.27.8 Practical Tips

- For symmetric posteriors, mean and median are essentially equal; choose whichever is conventional in your field.
- For posteriors over bounded parameters (probabilities, variances, r-coefficients), the mode can fall on the boundary; the mean stays in the interior and is a safer default.
- Posterior standard deviation is *not* the same as Monte Carlo standard error — the former is genuine uncertainty in θ, the latter is sampling noise from finite MCMC iterations. Report MCSE separately when it is appreciable relative to the posterior SD.
- Avoid reporting a "Bayesian point estimate" without specifying which one; default conventions differ across software and journals.
- For predictions, summarise the posterior predictive distribution rather

than transforming a point estimate of θ; only the former propagates parameter uncertainty.

10.27.9 R Packages Used

Base R suffices for raw summaries (`mean`, `median`, `var`, `sd`, `quantile`), `posterior::summarise_draws()` provides tidy tibbles with Monte Carlo standard errors, and `bayestestR::point_estimate()` exposes mean / median / MAP through one consistent interface for `brms` and `rstanarm` model objects.

10.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.27.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.28 Posterior Predictive Checks

10.28.1 Introduction

A posterior predictive check is the Bayesian answer to "does this model produce data that look like ours?" After fitting, you draw replicated datasets from the model — each one using a different posterior parameter draw — and compare them to the observed data through summary statistics, density overlays, or grouped subsets. Systematic discrepancies between observed and replicated quantities are a model criticism: they tell you which feature of the data the current specification fails to capture, which is far more actionable than a single goodness-of-fit p-value.

10.28.2 Prerequisites

A working understanding of the posterior predictive distribution and access to MCMC draws or any object that exposes a `posterior_predict()` method (`brms`, `rstanarm`, raw Stan).

10.28.3 Theory

The posterior predictive distribution is

$$p(\tilde{y} \mid y) = \int p(\tilde{y} \mid \theta) p(\theta \mid y)\, \mathrm{d}\theta,$$

approximated by drawing θ from the posterior and then $\tilde{y}$ from the likelihood given that θ. A posterior predictive check chooses a test statistic $T(y)$ — the mean, the standard deviation, the proportion of zeros, the maximum, the skew, or any function of the data — and compares its observed value to the distribution of $T(\tilde{y})$ across replicates. If the observed statistic falls in the bulk of the replicate distribution, the model captures that aspect of the data; if it falls in a tail, the model misses it.

10.28.4 Assumptions

A converged posterior with adequate effective sample size, since each replicate dataset depends on a posterior draw. PPCs do not require held-out data and are therefore not a measure of generalisation; they are a within-sample diagnostic.

10.28.5 R Implementation

```r
library(brms); library(bayesplot)

set.seed(2026)
d <- data.frame(y = rnorm(200, 3, 1.5))

fit <- brm(y ~ 1, data = d, family = gaussian,
           chains = 2, iter = 2000, refresh = 0)

# Density overlay
pp_check(fit)

# Test-statistic check: SD of simulated vs observed
pp_check(fit, type = "stat", stat = "sd")

# Interval check
pp_check(fit, type = "intervals")
```

10.28.6 Output & Results

`pp_check(fit)` returns a `ggplot` overlaying the empirical density of `y` on top of densities for several replicate datasets. `type="stat"` plots the distribution of the chosen statistic across replicates with the observed value as a vertical line. `type="intervals"` shows the 50 % and 90 % posterior predictive intervals point-by-point and is the most informative check for time-series and ordered data.

10.28.7 Interpretation

A reporting sentence: "Posterior predictive density overlays showed the observed distribution falling well within the cloud of replicates; the standard-deviation check placed the observed value near the centre of the replicate distribution ($p_{\text{ppc}} = 0.48$), indicating that the Gaussian model captures both location and dispersion adequately." When a check fails, name the feature: "skew of replicates was systematically lower than the observed skew", which directly suggests a heavy-tailed family or a transformation.

10.28.8 Practical Tips

- A density overlay is the broadest check; supplement it with tests on specific statistics (skew, kurtosis, proportion of zeros) that target likely misspecifications.
- Group-wise PPCs (`pp_check(fit,type="stat_grouped",group="site")`) catch problems concentrated in specific subgroups that pooled checks would mask.
- For count outcomes, `type="rootogram"` is more informative than density overlays because it highlights overdispersion and zero-inflation directly.
- A passing PPC does not guarantee good predictions on new data; pair PPCs with `loo()` or held-out evaluation when generalisation matters.
- When PPCs reveal misspecification, the fix is almost always a richer family (heavy tails, dispersion, mixtures) before a richer mean structure.

10.28.9 R Packages Used

`bayesplot` for the `pp_check()` and `ppc_*` graphics, `brms` (or `rstanarm`) as a backend that exposes `posterior_predict()` directly, and `posterior` for handling draw arrays when building custom PPCs by hand.

10.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.28.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.29 The Posterior Predictive Distribution

10.29.1 Introduction

The posterior predictive distribution (PPD) is the Bayesian answer to the question "what range of values can a future observation plausibly take?" It accounts for two sources of uncertainty simultaneously: the sampling variability that would persist even if the parameters were known, and the residual parameter uncertainty quantified by the posterior. A plug-in prediction that uses only the posterior mean throws away the second source and produces intervals that are too narrow — a frequent and consequential mistake in applied work.

10.29.2 Prerequisites

A working understanding of posterior distributions, the difference between parameter inference and observation prediction, and Monte Carlo sampling from MCMC draws.

10.29.3 Theory

For data y and a prospective new observation $\tilde{y}$, the posterior predictive distribution is

$$p(\tilde{y} \mid y) = \int p(\tilde{y} \mid \theta) p(\theta \mid y) \, \mathrm{d}\theta.$$

The Monte Carlo approximation is direct: for each posterior draw $\theta^{(s)}$, sample $\tilde{y}^{(s)} \sim p(\tilde{y} \mid \theta^{(s)})$. The collection $\{\tilde{y}^{(s)}\}_{s=1}^{S}$ is itself a sample from the PPD, and any quantile, interval, or expectation of interest comes from it. Because the integral over θ is automatic in this formulation, the resulting prediction interval is wider than the corresponding interval based on a plug-in $\hat{\theta}$, and that extra width is the propagated parameter uncertainty.

10.29.4 Assumptions

Valid posterior draws (converged chains, adequate ESS) and the assumed data-generating model $p(\tilde{y} \mid \theta)$ for the new observation. If the new observation is from a different population or covariate range, the PPD inherits the model's extrapolation behaviour and may be unrealistic.

10.29.5 R Implementation

```
library(brms)

set.seed(2026)
n <- 100
```

```
d <- data.frame(x = rnorm(n), y = NA)
d$y <- 2 + 1.5 * d$x + rnorm(n)

fit <- brm(y ~ x, data = d, chains = 2, iter = 2000, refresh = 0)

# Posterior predictive samples
pp <- posterior_predict(fit, newdata = data.frame(x = 0:3))
apply(pp, 2, quantile, c(0.025, 0.5, 0.975))
```

10.29.6 Output & Results

`posterior_predict()` returns a matrix with one row per posterior draw and one column per `newdata` row. Summarising columnwise gives the posterior predictive median and 95 % interval at each new x. The resulting interval combines the residual standard deviation with the parameter uncertainty inherited from the posterior of β_0 and β_1.

10.29.7 Interpretation

A reporting sentence: "At $x = 2$ the posterior predictive 95 % interval for a new observation is 3.1 to 6.8, with median 5.0; the corresponding interval for the conditional mean is 4.7 to 5.3, illustrating that residual variability dominates the predictive uncertainty in this model." Distinguishing the predictive interval from the credible interval on the mean is essential — they answer different questions and the former is always wider.

10.29.8 Practical Tips

- Use `posterior_epred()` for the conditional-mean credible interval (no residual variability), `posterior_predict()` for the full predictive distribution including residual variation, and `posterior_linpred()` if you need the linear predictor before any link transformation.
- Pair PPD draws with `pp_check(fit)` to confirm that the model captures the marginal distribution of the data; a misspecified family will produce predictive intervals that are too tight or too wide where they matter.
- For binary, count, or ordinal outcomes, the PPD is a discrete distribution; visualise with rootograms or bar charts rather than density curves.
- When predicting beyond the observed predictor range, report posterior predictive intervals separately for in-sample and out-of-sample x; readers should see where extrapolation begins.
- Always summarise the PPD on the original scale of the outcome; transforming a credible interval on the linear predictor and adding back-of-the-envelope residual SD is rarely correct.

10.29.9 R Packages Used

`brms` and `rstanarm` for `posterior_predict()`, `posterior_epred()`, and `posterior_linpred()`; `tidybayes::predicted_draws()` for tidy long-format predictive draws; `bayesplot` for `pp_check()` and other PPD diagnostic plots.

10.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.29.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.30 Prior Specification

10.30.1 Introduction

Choosing a prior is the most consequential modelling decision in any Bayesian analysis after the choice of likelihood. The prior shapes the posterior strongly when data are sparse, almost imperceptibly when data are abundant, and unpredictably in the middle ground that most applied work occupies. Three categories cover most needs: flat or improper priors, weakly informative priors that regularise without imposing substantive content, and informative priors that incorporate genuine pre-data knowledge. Modern Bayesian practice has converged on weakly informative priors as the default because they stabilise sampling, regularise small-sample inference, and remain agnostic enough that the data dominate the posterior whenever they are informative.

10.30.2 Prerequisites

A working understanding of Bayes' theorem, the role of the prior in posterior updating, and basic familiarity with parameter scales (standardised vs. raw, log-scale vs. linear).

10.30.3 Theory

Flat / improper priors assign constant density across the support. They sometimes appear intuitive but can yield improper posteriors in hierarchical models or near boundary points, and they are not invariant under reparameterisation.

Weakly informative priors are proper distributions broad enough to admit any plausible value while ruling out absurd ones. Standard choices are $\mathcal{N}(0, 2.5)$ for standardised slopes, $\mathcal{N}(0, 10)$ for intercepts, half-Cauchy$(0, 5)$ or half-Normal$(0, 2.5)$ for variance components, and LKJ(2) for correlation matrices. They regularise estimates without injecting domain claims.

Informative priors carry substantive content from previous studies, theoretical bounds, or expert elicitation. They are appropriate when the information is defensible and reproducible, and inappropriate when chosen post-hoc to push the posterior somewhere convenient.

The **prior predictive check** — simulating from the prior, propagating through the likelihood, and inspecting the implied range of `y` — is the single most useful sanity check. If the prior implies blood pressures of 1,000 mm Hg or odds ratios of 10^6, it is too wide.

10.30.4 Assumptions

The prior reflects pre-data information honestly, the likelihood is correctly specified, and the prior is proper unless the resulting posterior has been verified to be proper analytically.

10.30.5 R Implementation

```
library(brms)

# Logistic regression with weakly informative priors
priors <- prior(normal(0, 2.5), class = "b") +
          prior(normal(0, 10),  class = "Intercept")

# brms: inspect priors before fitting
get_prior(y ~ x1 + x2, data = data.frame(y = rbinom(100, 1, 0.5),
                                         x1 = rnorm(100), x2 = rnorm(100)),
          family = bernoulli)
```

10.30.6 Output & Results

`get_prior()` lists every parameter class with its default prior, ready to be overridden. After fitting, `prior_summary(fit)` confirms exactly which priors entered the model — a step worth running for any analysis intended for publication.

10.30.7 Interpretation

A reporting sentence: "We placed weakly informative priors on all coefficients ($\beta \sim \mathcal{N}(0, 2.5)$ on standardised predictors, intercept $\sim \mathcal{N}(0, 10)$) and a half-Cauchy$(0, 2.5)$ prior on the random-effect standard deviation; a sensitivity

analysis under tighter $\mathcal{N}(0, 1)$ slopes produced indistinguishable posterior summaries." Always disclose priors and at least one sensitivity check; reviewers increasingly expect both.

10.30.8 Practical Tips

- Defaults from `brms` and `rstanarm` are weakly informative and well-engineered; override them only when you have a reason.
- Flat priors are not "neutral"; they can dominate hierarchical posteriors at the boundary and make sampling unstable.
- For variance components, use half-Cauchy or half-Normal priors with scales matched to the natural variability of the data; never use a flat prior on a SD.
- For unstandardised predictors, multiply prior scales by the empirical SD of the predictor so that the implied effect range is plausible.
- Run prior predictive simulations (`sample_prior="only"` in `brms`) before fitting; the implied range of `y` is the simplest check that the prior is not absurd.
- Sensitivity analysis is mandatory for small-sample studies; report the posterior under at least two prior scales differing by an order of magnitude.

10.30.9 R Packages Used

`brms` for prior specification with `get_prior()` / `prior_summary()`, `rstanarm` for an alternative interface with auto-scaled defaults, `bayesplot` for prior predictive visualisation, and `bayestestR` for tidy summaries that report priors alongside posteriors.

10.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.30.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.31 R-hat and ESS Diagnostics

10.31.1 Introduction

An MCMC posterior summary is only as good as the chain that produced it. Two questions decide whether the draws are usable: have the chains converged on the same distribution, and how many independent observations is the autocorrelated chain actually worth? The first is answered by $\hat{R}$, the second by effective sample size (ESS). These two numbers, reported alongside every parameter in modern Bayesian output, are the minimum diagnostic checklist before any posterior interval is interpreted.

10.31.2 Prerequisites

A working understanding of Markov chain Monte Carlo, the role of warm-up versus sampling iterations, and the autocorrelation structure of MCMC draws.

10.31.3 Theory

$\hat{R}$ compares the variance between chains to the variance within chains. If chains have converged on the same target, the two variance estimates agree and $\hat{R} \to 1$; persistent disagreement inflates $\hat{R}$. The Vehtari et al. (2021) rank-normalised variant handles heavy-tailed posteriors and is the default in `posterior::summarise_draws()`. ESS expresses the variance of the Monte Carlo estimator as if the draws were independent: $n_{\text{eff}} = N / (1 + 2\sum_{k} \rho_k)$ for autocorrelations ρ_k. Bulk ESS quantifies precision of the posterior mean and median; tail ESS quantifies precision of the 5 % and 95 % quantiles, which require many more draws because they are estimated from a thin part of the distribution.

10.31.4 Assumptions

At least four chains started from dispersed initial values, with adequate warm-up so that the sampler has tuned step size and mass matrix. A single chain cannot diagnose convergence, regardless of how long it is — between-chain variance is undefined.

10.31.5 R Implementation

```
library(brms); library(posterior)

data(sleepstudy, package = "lme4")
fit <- brm(Reaction ~ Days + (Days | Subject), data = sleepstudy,
           chains = 4, iter = 2000, refresh = 0)

# Summary with R-hat and ESS
summary(fit)
```

```
# Per-parameter
post_draws <- as_draws_df(fit)
summarise_draws(post_draws)
```

10.31.6 Output & Results

`summary(fit)` reports `Rhat`, `Bulk_ESS`, and `Tail_ESS` for every parameter. `summarise_draws()` returns the same diagnostics in a tidy tibble, ready for `dplyr` filtering when a model has hundreds of parameters and you want to know which one needs attention.

10.31.7 Interpretation

A reporting sentence: "All parameters had $\hat{R} \leq 1.00$, bulk ESS $> 2{,}000$, and tail ESS $> 1{,}500$; sampling is adequate for posterior intervals." Conventional thresholds: $\hat{R} < 1.01$ indicates convergence, $\hat{R} > 1.05$ is a clear failure; bulk and tail ESS should each exceed 400 for credible interval estimation to be reliable.

10.31.8 Practical Tips

- If $\hat{R}$ exceeds 1.01, double the iteration count first; if the problem persists, reparameterise (e.g. non-centred random effects) rather than throwing more iterations at it.
- Low ESS relative to total draws (ratios under 0.1) signals strong autocorrelation; raise `adapt_delta` or examine the model for funnel-shaped geometry.
- Divergent transitions in HMC are a separate, more serious diagnostic — they indicate that the sampler missed regions of the posterior, not just that it explored slowly. Always inspect them with `bayesplot::mcmc_parcoord()` or `pairs()`.
- Trace plots remain a useful supplement: a "hairy caterpillar" with chains overlapping is what convergence looks like; a chain wandering away or stuck in a valley is visible long before $\hat{R}$ catches it for short runs.
- For very high-dimensional models, monitor $\hat{R}$ and ESS for *derived* quantities of interest (e.g. predictions, contrasts), not just raw parameters; convergence of one does not imply the other.

10.31.9 R Packages Used

`posterior` for the rank-normalised $\hat{R}$, bulk and tail ESS, and tidy-draw structures, `bayesplot` for diagnostic plots, and `brms` (or `rstanarm`, `cmdstanr`) as the modelling backend that produces the draws in the first place.

10.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.31.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.32 rstanarm

10.32.1 Introduction

`rstanarm` is a sister package to `brms` from the Stan team. Where `brms` translates each formula into a fresh Stan program and compiles it on the fly, `rstanarm` ships with a fixed catalogue of pre-compiled models — `stan_lm`, `stan_glm`, `stan_glmer`, `stan_polr`, `stan_betareg`, and a handful of others — that match the most common R regression interfaces. The trade-off is straightforward: no compilation latency at fit time, weakly informative defaults curated by experts, and a learning curve identical to base-R modelling functions, in exchange for less flexibility when the model deviates from the standard catalogue.

10.32.2 Prerequisites

A working knowledge of generalised linear models, an idea of what a prior and a posterior are, and comfort with R's `lm()` / `glm()` / `lmer()` syntax. No C++ toolchain is needed beyond what installing the package itself requires, since the Stan models are pre-compiled.

10.32.3 Theory

Each `stan_*` function is a thin R wrapper around a Stan program that has been compiled when the package was built. The program already includes weakly informative default priors centred at zero with scales determined by the standard deviation of each predictor. Override defaults through the `prior`, `prior_intercept`, `prior_aux`, and `prior_covariance` arguments, each accepting one of `normal()`, `student_t()`, `cauchy()`, `laplace()`, `exponential()`, or `decov()`. Because the priors are scaled to the predictors, the same numerical scale (e.g. `normal(0,2.5)`) is sensible across very different predictor ranges.

10.32.4 Assumptions

The assumptions of the underlying GLM or GLMM apply: linearity in the linear predictor, independence after grouping, the appropriate link / variance relationship for the family. Convergence ($\hat{R} < 1.01$, no divergent transitions) is a precondition for trusting the posterior summary that `rstanarm` returns.

10.32.5 R Implementation

```r
library(rstanarm)

set.seed(2026)
d <- data.frame(y = rnorm(100), x = rnorm(100))

# Same syntax as lm() but Bayesian
fit <- stan_glm(y ~ x, data = d,
                prior = normal(0, 2.5),
                prior_intercept = normal(0, 5),
                prior_aux = exponential(1),
                chains = 4, iter = 2000, refresh = 0)
summary(fit)
posterior_interval(fit, prob = 0.95)
```

10.32.6 Output & Results

`summary(fit)` lists the posterior mean, standard deviation, 10 / 50 / 90 % quantiles, mean posterior predictive checks (PPP), $\hat{R}$, and effective sample size for each parameter. `posterior_interval()` returns the requested credible interval. Because the same Stan program is reused, fits typically take seconds rather than minutes.

10.32.7 Interpretation

A reporting sentence: "A Bayesian linear regression (`stan_glm`, weakly informative priors, four chains $\times$ 2,000 iterations) estimated the slope at 0.47 (95 % CrI 0.27 to 0.68); the posterior overlaps the OLS point estimate but quantifies the residual uncertainty rather than just standard errors." Compared with frequentist output, the credible interval has the direct probability interpretation that students often (incorrectly) assign to confidence intervals.

10.32.8 Practical Tips

- Use `prior_summary(fit)` to verify exactly which priors were applied — the auto-scaling can surprise you when predictors are on unusual scales.
- For mixed models, `stan_glmer()` mirrors `lme4::glmer()` syntax including correlated random slopes.

- `pp_check(fit)` provides graphical posterior predictive checks; `bayes_R2()` returns the Bayesian counterpart of R^2 as a posterior distribution.
- `loo(fit)` and `waic(fit)` attach information criteria with Pareto-k diagnostics, ready for `loo_compare()` between models.
- When the model you need is not in the catalogue (for example, custom likelihoods, censored data, ordinal-with-arbitrary-link), reach for `brms` or raw Stan; `rstanarm` is deliberately scoped.
- For very small datasets, prefer `student_t(df=3)` priors over `normal()` to reduce posterior sensitivity to a single influential observation.

10.32.9 R Packages Used

`rstanarm` for the pre-compiled Stan models, `bayesplot` for diagnostic plots underlying `pp_check()`, `loo` for cross-validation, and `posterior` for working with the underlying draws.

10.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.32.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.33 The Savage-Dickey Density Ratio

10.33.1 Introduction

The Savage-Dickey density ratio is a small but powerful identity: when a point-null hypothesis is nested inside a continuous alternative, the Bayes factor in favour of the null can be read off as the ratio of the marginal posterior to the marginal prior, both evaluated at the null value. This sidesteps the notoriously hard computation of the marginal likelihood under each model and turns Bayes-factor calculation into something you can do from MCMC draws and a one-line evaluation of the prior density. It is the Bayesian counterpart of asking "did the data move probability mass away from the null?" and quantifying the answer.

10.33.2 Prerequisites

A working understanding of Bayes factors, posterior densities, and the difference between point-null and interval-null hypotheses. Familiarity with density

estimation from MCMC samples is also useful.

10.33.3 Theory

For a point null $H_0 : \theta = \theta_0$ nested in a continuous alternative $H_1 : \theta \in \Theta$, with the prior under H_1 taken as the marginal prior $p(\theta)$ on the parameter of interest, the Bayes factor in favour of the null is

$$\mathrm{BF}_{01} = \frac{p(\theta_0 \mid y)}{p(\theta_0)}.$$

If the posterior puts more density at the null than the prior did, the data support H_0; if it puts less, they support H_1. The quantity is interpretable on Jeffreys' evidence scale and is invariant under monotone reparameterisations of nuisance parameters because the alternative's prior must be set up consistently. The identity assumes the prior under H_1 reduces to the prior under H_0 when $\theta = \theta_0$; this is automatic for nested-null setups but can fail under improper or hierarchical priors.

10.33.4 Assumptions

A continuous parameter, the null nested inside the alternative, and a proper prior whose density at θ_0 is finite and known. Improper priors invalidate the construction because the Bayes factor depends on a normalisation constant that has been thrown away.

10.33.5 R Implementation

```
library(polspline)

set.seed(2026)
# Posterior samples (e.g., from MCMC)
theta_post <- rnorm(5000, mean = 0.2, sd = 0.3)
theta_prior_density <- dnorm(0, mean = 0, sd = 1)   # N(0, 1) prior at 0

# Posterior density at 0 via logspline fit
logspl <- logspline(theta_post)
post_density_at_0 <- dlogspline(0, logspl)

BF01 <- post_density_at_0 / theta_prior_density
BF01; 1/BF01
```

10.33.6 Output & Results

The ratio of two scalars, ideally accompanied by its inverse so both directions are visible. A logspline density fit to the posterior draws is a robust choice: it

captures skew and tail behaviour better than a normal approximation and is less noisy than a kernel density estimate at the boundary.

10.33.7 Interpretation

A reporting sentence: "The Savage-Dickey Bayes factor was $\mathrm{BF}_{10} = 3.1$ under a $\mathcal{N}(0, 1)$ prior on the alternative — moderate evidence for a non-zero effect — versus $\mathrm{BF}_{01} = 0.32$." On Jeffreys' scale, $1 < \mathrm{BF} < 3$ is anecdotal, 3–10 is moderate, 10–30 strong, and beyond 30 very strong; the symmetric thresholds apply to BF_{01} for evidence in favour of the null.

10.33.8 Practical Tips

- Always disclose the prior under the alternative; the Bayes factor is sensitive to its scale, particularly via the Lindley-Bartlett paradox where a very wide prior favours the null mechanically.
- Use a logspline or kernel density estimate that is robust at the boundary — naive Gaussian approximations break down when the posterior is skewed.
- Confirm enough posterior draws fall near θ_0 for the density estimate to be stable; if the data strongly support H_1, the posterior density at the null is small and noisy. Increase chain length when reporting strong evidence against H_0.
- For interval nulls or composite hypotheses, the Savage-Dickey identity does not apply; use bridge sampling (`bridgesampling::bridge_sampler()`) instead.
- Prefer a sensitivity analysis: report the Bayes factor under at least two prior scales, so readers can judge how prior-driven the conclusion is.

10.33.9 R Packages Used

`polspline` for the logspline posterior density estimate at the null, `bayestestR` for an integrated Savage-Dickey wrapper that works directly with `brms` and `rstanarm` fits, and `bridgesampling` for the more general marginal-likelihood approach when the Savage-Dickey shortcut does not apply.

10.33.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.33.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error

- brms / Stan, LOO, hierarchical models

10.34 Stan: Introduction

10.34.1 Introduction

Stan is a probabilistic programming language with an R-like syntax that compiles to highly optimised C++. Writing a model directly in Stan gives you full control over the likelihood, the prior, and any reparameterisation that improves sampling — control that the higher-level wrappers `brms` and `rstanarm` deliberately abstract away. The cost is verbosity, but the payoff is a transparent specification: every line in the `model` block contributes a term to the log-posterior, and a Stan program can be archived as the canonical statement of an analysis.

10.34.2 Prerequisites

A working understanding of Bayesian inference, the difference between a prior and a likelihood, and the basics of Hamiltonian Monte Carlo. You also need a working C++ toolchain (Rtools on Windows, build-essential on Linux) so that `rstan` or `cmdstanr` can compile the model.

10.34.3 Theory

A Stan program is organised into blocks evaluated in a specific order:

- `data`: variables that come from the user, dimensioned and constrained.
- `transformeddata` (optional): deterministic transformations of the data computed once at compile time.
- `parameters`: continuous unknowns to be inferred. Constraints (`<lower=0>`) are reparameterised internally, which is why discrete parameters have to be marginalised out.
- `transformedparameters` (optional): deterministic functions of parameters, recomputed at every leapfrog step.
- `model`: contributions to the log-posterior — both priors on parameters and the likelihood given the data.
- `generatedquantities` (optional): posterior predictive draws, log-likelihoods for LOO-CV, or any deterministic quantity to monitor.

The default sampler is the No-U-Turn variant of HMC, which adapts step size and mass matrix during a warm-up phase before producing draws.

10.34.4 Assumptions

Continuous parameters or correctly marginalised discrete latent states. The posterior must be log-concave only locally, but pathological geometry (funnels,

ridges, multimodality) often shows up as divergent transitions and demands reparameterisation.

10.34.5 R Implementation

```
// Stan model: linear regression
data {
  int<lower=1> N;
  vector[N] x;
  vector[N] y;
}
parameters {
  real alpha;
  real beta;
  real<lower=0> sigma;
}
model {
  alpha ~ normal(0, 10);
  beta  ~ normal(0, 2.5);
  sigma ~ exponential(1);
  y ~ normal(alpha + beta * x, sigma);
}
```

```
library(rstan)

set.seed(2026)
d <- list(N = 100, x = rnorm(100), y = rnorm(100, 2 + 1.5 * rnorm(100),
 ↪  1))

# Assume the Stan code is saved to "model.stan"
# fit <- stan("model.stan", data = d, chains = 4, iter = 2000)
# print(fit, pars = c("alpha", "beta", "sigma"))
```

10.34.6 Output & Results

`print(fit)` returns posterior mean, standard error of the mean, posterior standard deviation, 2.5 / 25 / 50 / 75 / 97.5 % quantiles, effective sample size, and $\hat{R}$ for every monitored parameter. Diagnostics also include divergent-transition counts, maximum tree depth saturation, and energy diagnostics — any of which, if non-zero, signals geometry trouble.

10.34.7 Interpretation

Compared with `lm()` or `glmer()` style output, Stan output is more verbose but more explicit: there is no implicit prior, no black-box optimisation, and

the credible interval has a direct probability interpretation conditional on the model. A reporting sentence: "A Stan-compiled linear regression with weakly informative priors estimated the slope at 1.49 (95 % CrI 1.30 to 1.69), with $\hat{R} = 1.00$ and $n_{\text{eff}} > 3{,}500$ for every parameter."

10.34.8 Practical Tips

- Use `brms::stancode()` or `rstanarm::stan_glm` source as templates — they are well-engineered models you can copy and modify.
- Reparameterise hierarchical models in non-centred form ($\theta_j = \mu + \tau \cdot \tilde{\theta}_j$ with $\tilde{\theta}_j \sim \mathcal{N}(0, 1)$) to eliminate funnel-shaped divergences.
- Prefer `cmdstanr` over `rstan` for new projects: it tracks upstream Stan releases, has a cleaner API, and avoids the binary-compatibility surprises that plague R-package builds.
- Vectorise statements (`y~normal(mu,sigma)` over `for` loops); the compiler optimises vectorised forms more aggressively.
- Place posterior predictive draws in `generatedquantities`, not the `model` block, so they do not enter the log-density evaluation and slow sampling.

10.34.9 R Packages Used

`rstan` or `cmdstanr` to compile and sample from Stan programs, `posterior` for working with draw arrays, `bayesplot` for diagnostic and posterior plots, and `loo` for leave-one-out cross-validation.

10.34.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.34.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.35 tidybayes Workflow

10.35.1 Introduction

`tidybayes` is the bridge between Bayesian model objects and the rest of the tidyverse. After fitting a model with `brms`, `rstanarm`, or raw Stan, you typically want to compare posteriors across parameters, summarise interval estimates by

subgroup, or layer credible bands on top of raw observations. Doing this with the matrices that samplers natively produce is awkward; `tidybayes` reshapes them into long-format tibbles in which every draw of every parameter occupies its own row, making the familiar `dplyr` verbs and `ggplot2` geoms work without modification.

10.35.2 Prerequisites

A working understanding of Bayesian inference (priors, likelihoods, posteriors), comfort with `dplyr` pipelines, and a fitted MCMC model object. You do not need to write Stan by hand — `brm()` or `stan_glm()` is enough.

10.35.3 Theory

A posterior is a joint distribution over parameters represented as S MCMC draws. `tidybayes::spread_draws()` returns one row per draw per parameter, so a model with K regression coefficients yields $S \times K$ rows. `gather_draws()` produces an even longer format with a `.variable` column, useful when faceting. The package adds geoms — `stat_halfeye`, `stat_pointinterval`, `stat_lineribbon` — that compute density and interval summaries on the fly, so the data frame stays untransformed. Predictive draws come from `epred_draws()` (expectation of the posterior predictive) and `predicted_draws()` (full predictive distribution including residual variation).

10.35.4 Assumptions

The model has already converged: chains have mixed, $\hat{R}$ is below 1.01, and effective sample size is adequate. `tidybayes` does not diagnose convergence; it visualises whatever draws you give it, so a poorly mixed chain produces a misleading posterior summary.

10.35.5 R Implementation

```
library(brms); library(tidybayes); library(ggplot2)

data(mtcars)
fit <- brm(mpg ~ wt + hp, data = mtcars, chains = 4, iter = 2000, refresh
 ↪  = 0)

# Tidy draws
draws <- fit |>
  spread_draws(b_wt, b_hp)

head(draws)

# Visualise posterior
```

```
draws |>
  pivot_longer(c(b_wt, b_hp), names_to = "parameter", values_to =
   ↪  "value") |>
  ggplot(aes(x = value, y = parameter)) +
  stat_halfeye(fill = "#2A9D8F") +
  theme_minimal()

# Conditional effects
fit |>
  epred_draws(newdata = data.frame(wt = 2.5, hp = seq(50, 300, 10))) |>
  ggplot(aes(x = hp, y = .epred)) +
  stat_lineribbon(alpha = 0.3) +
  theme_minimal()
```

10.35.6 Output & Results

`spread_draws()` returns a tibble with `.chain`, `.iteration`, `.draw`, and one column per parameter. The half-eye plot shows the posterior density together with a black point at the median and bars at the 66 % and 95 % credible intervals. `stat_lineribbon` produces a shaded band that widens at horsepower values poorly supported by the data, communicating where extrapolation is risky.

10.35.7 Interpretation

A reporting sentence: "The posterior median effect of weight on miles-per-gallon was −3.9 (95 % CrI −5.4 to −2.4), while horsepower contributed −0.03 (95 % CrI −0.05 to −0.01); both intervals exclude zero, consistent with each predictor having a non-trivial association after the other is controlled for." Pair narrative numbers with the half-eye plot so readers can see the full posterior shape and not just an interval.

10.35.8 Practical Tips

- Use `spread_draws()` when you want one column per parameter and `gather_draws()` when you plan to facet by parameter name.
- Regular expressions are accepted: `spread_draws(fit,b_[a-z]+)` extracts every regression coefficient at once.
- `add_epred_draws()` and `add_predicted_draws()` attach predictive draws directly to a `newdata` frame, which is convenient when overlaying credible bands on observed points.
- Use `mean_qi()`, `median_qi()`, or `mode_hdi()` to collapse draws into interval summaries; `point_interval=mean_qi` is the default in most geoms but is worth setting explicitly for reproducibility.
- The package is engine-agnostic: the same code path supports `rstan`, `cmdstanr`, `rstanarm`, and `brms`, so refactoring between them rarely affects

downstream plotting code.
- Avoid coercing draws to wide format prematurely; `ggplot2` is happiest with the long tibble that `tidybayes` returns.

10.35.9 R Packages Used

`brms` for model fitting, `tidybayes` for posterior reshaping and stat geoms, `ggplot2` and `dplyr` for plotting and manipulation, plus `posterior` (loaded transitively) for low-level draw arrays.

10.35.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.35.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.36 WAIC

10.36.1 Introduction

The widely applicable information criterion (WAIC), introduced by Watanabe, estimates the expected out-of-sample prediction score from posterior samples without holding out data. It is the Bayesian successor to AIC and DIC: AIC requires a single point estimate (and assumes regularity that fails for hierarchical models); DIC plug-ins a posterior mean of the deviance and is unstable in mixed models. WAIC instead averages the pointwise log-posterior predictive density across posterior draws and penalises by an effective-parameter term, making it directly applicable to any model with a well-defined pointwise likelihood.

10.36.2 Prerequisites

A working understanding of information criteria, posterior predictive distributions, and the relationship between in-sample fit and out-of-sample prediction.

10.36.3 Theory

For data $y_1, \dots, y_n$ and posterior draws $\theta^{(s)}$, the log pointwise predictive density is

$$\widehat{\text{lpd}} = \sum_{i=1}^{n} \log\left(\frac{1}{S}\sum_{s=1}^{S} p(y_i \mid \theta^{(s)})\right).$$

The effective parameter count p_{waic} is the sum of pointwise posterior variances of the log-likelihood. Then

$$\text{WAIC} = -2(\widehat{\text{lpd}} - p_{\text{waic}}).$$

Lower is better. Watanabe's asymptotic theory shows that for regular models WAIC is asymptotically equivalent to leave-one-out cross-validation. The standard error of a WAIC difference between two models comes from the sample standard deviation of the pointwise differences scaled by $\sqrt{n}$, providing a principled threshold for "meaningful" differences.

10.36.4 Assumptions

A pointwise likelihood is well-defined (so each observation contributes one term to the sum) and the posterior is well-mixed. WAIC inherits the exchangeability assumption of standard cross-validation — for clustered or temporal data, replace it with leave-one-cluster-out or leave-future-out variants.

10.36.5 R Implementation

```
library(brms); library(loo)

data(mtcars)
fit1 <- brm(mpg ~ wt, data = mtcars, chains = 2, iter = 2000, refresh = 0)
fit2 <- brm(mpg ~ wt + hp, data = mtcars, chains = 2, iter = 2000,
 ↪  refresh = 0)

w1 <- waic(fit1)
w2 <- waic(fit2)

loo_compare(w1, w2)
```

10.36.6 Output & Results

`waic()` returns the WAIC value, p_{waic}, and a vector of pointwise contributions. `loo_compare()` aligns two WAIC objects, reports their differences and standard errors, and orders models from best (top) to worst (bottom).

10.36.7 Interpretation

A reporting sentence: "WAIC favoured the two-predictor model with $\Delta\text{WAIC} = -12 \pm 4$ (standard error of the difference); under the rule-of-thumb that differences exceeding $2 \times \text{SE}$ are meaningful, the data clearly support including horsepower in addition to weight." Always quote both the difference and its standard error — a "12-point improvement" without context is uninformative.

10.36.8 Practical Tips

- Use PSIS-LOO from the `loo` package whenever possible; it is more numerically stable than WAIC and provides Pareto-k diagnostics that flag influential observations one at a time.
- WAIC can be unstable when a single observation contributes a large fraction of the variance of the log-likelihood; if you see p_{waic} comparable to the number of observations, do not trust the criterion and switch to LOO.
- Differences in WAIC are noisy; only invest belief in differences that exceed 2 standard errors.
- WAIC is well-defined for non-nested models, unlike the likelihood-ratio test, which makes it natural for comparing models with different families or link functions.
- For hierarchical models, choose the leave-out unit deliberately: leaving out a single observation may understate predictive uncertainty if all other observations from the same cluster remain in the training fold.

10.36.9 R Packages Used

`loo` for the canonical `waic()` and `loo()` implementations and `loo_compare()`, `brms` and `rstanarm` for fits that expose pointwise log-likelihoods automatically, and `posterior` for the underlying draw arrays.

10.36.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.36.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.37 Weakly Informative Priors

10.37.1 Introduction

Weakly informative priors are the modern default in applied Bayesian work. They assign non-trivial probability to a plausible range of parameter values while remaining broad enough that the data dominate the posterior. They differ from "non-informative" priors (which can be improper or pathological) and from informative priors (which encode substantive knowledge): the goal is to regularise the inference, rule out absurd values, and stabilise sampling without injecting domain claims that the data cannot challenge. The result is a posterior that looks much like a frequentist estimate when data are abundant, but that behaves sensibly under separation, sparsity, and high-dimensional regression where likelihood-only methods misbehave.

10.37.2 Prerequisites

A working understanding of priors, the role of the prior in Bayesian updating, and basic familiarity with regression on standardised predictors.

10.37.3 Theory

A prior is *weakly informative* if it gives small but non-zero probability to extreme values, has a finite second moment when possible, and is centred at a value that does not bias the analysis (zero, in most regression contexts). Standard choices for standardised predictors:

- Regression slopes: $\mathcal{N}(0, 2.5)$ or $\mathcal{N}(0, 5)$ on standardised covariates.
- Intercept (after centring y): $\mathcal{N}(0, 10)$.
- Standard deviations of random effects or residuals: half-Normal$(0, 2.5)$, half-Cauchy$(0, 5)$, or Exponential(λ) with rate matched to the SD scale.
- Correlation matrices: LKJ prior with shape $\eta \geq 2$, which gently regularises toward weakly correlated structures.

A useful calibration test: simulate from the prior predictive distribution and check that the implied range of `y` is plausible. If the prior implies effects of $\pm 1{,}000$ on a quantity bounded by physiology, it is too wide.

10.37.4 Assumptions

Predictors are on a sensible scale (typically standardised), priors are proper (so the posterior is integrable), and the prior predictive range covers the plausible space without dominating it.

10.37.5 R Implementation

```
library(brms)

# Default brms priors are weakly informative
data_ex <- data.frame(y = rnorm(100), x = rnorm(100))
get_prior(y ~ x, data = data_ex)

# Explicit weakly informative
fit <- brm(y ~ x, data = data_ex,
           prior = prior(normal(0, 2.5), class = "b") +
                   prior(normal(0, 5),   class = "Intercept") +
                   prior(exponential(1), class = "sigma"),
           iter = 2000, chains = 4, refresh = 0)
```

10.37.6 Output & Results

`get_prior()` lists every parameter class together with the package's default prior; explicit overrides replace them by class. After fitting, `prior_summary(fit)` confirms exactly which priors entered the model — a step worth running for any analysis intended for publication.

10.37.7 Interpretation

A reporting sentence: "Weakly informative priors $\mathcal{N}(0, 2.5)$ on standardised slopes and Exponential(1) on the residual scale yielded a posterior centred close to the OLS estimate ($\beta = 0.51$, 95 % credible interval 0.31 to 0.71), with negligible prior pull at this sample size; a sensitivity analysis under $\mathcal{N}(0, 1)$ produced essentially identical posterior summaries." Disclosing both the prior and a sensitivity check is the gold-standard report.

10.37.8 Practical Tips

- Gelman's heuristic: half-Cauchy with scale 2.5–5 for variance components — heavy enough to allow large variances when warranted, regularising enough to prevent boundary-zero estimates.
- For unstandardised predictors, multiply the prior scale by the predictor's empirical SD to keep the implied effect range plausible.
- LKJ(2) for correlation matrices is a sensible default; tighten only when you have reason to expect weak correlations.
- Always run prior predictive checks (`sample_prior="only"`); they reveal absurd implied ranges before the data can mask them.
- For non-standardised log-odds problems where the baseline log-odds is far from zero, widen the prior on the intercept (e.g. $\mathcal{N}(0, 10)$) so the prior mode does not exclude the data-implied baseline.

10.37.9 R Packages Used

`brms` for formula-based prior specification with `get_prior()` / `prior_summary()`, `rstanarm` for an alternative interface with auto-scaled defaults, `bayestestR` for posterior summaries that report the prior alongside the posterior, and `bayesplot` for prior predictive visualisation.

10.37.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

10.37.11 See also — labs in this chapter

- Bayesian thinking; α control vs decision error
- brms / Stan, LOO, hierarchical models

10.38 Week 3, Session 1 — Bayesian thinking; α control vs decision error

Testing labs use the main template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

10.38.1 Learning objectives

- Compute a beta-binomial posterior and interpret its credible intervals in terms of a decision.
- Contrast frequentist α-control (type I error rate across repeated experiments) with Bayesian decision error for a specific experiment.
- Show how prior sensitivity translates into decision sensitivity.

10.38.2 Prerequisites

Probability basics and conjugacy.

10.38.3 Background

Frequentist testing controls the long-run rate of falsely rejecting a true null across repeated experiments; the $\alpha = 0.05$ convention is a statement about procedure, not about the current dataset. Bayesian inference instead returns a posterior distribution of the unknown and turns that into a decision by weighing

costs. For a binomial proportion with a beta prior, the posterior is beta again, so the arithmetic is clean enough to follow carefully.

The two frameworks answer different questions. "Given these data and this model, what is the probability that the treatment is beneficial?" is a Bayesian question. "Across many repetitions of this design, how often would I reject a true null?" is a frequentist question. In applied biomedicine, both questions are relevant; knowing which one you are answering prevents confused write-ups.

Prior sensitivity is a useful discipline regardless of philosophy. A posterior that shifts materially when the prior is varied across reasonable alternatives is less trustworthy than one that does not, and sharing that sensitivity is the Bayesian analogue of transparent model-building.

10.38.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

10.38.5 1. Hypothesis

A small trial observes $k = 7$ successes in $n = 20$ patients of a new response-rate biomarker. Is the underlying response rate meaningfully above 0.2?

10.38.6 2. Visualise

Prior vs posterior for three priors.

```
k <- 7; n <- 20
theta <- seq(0, 1, length.out = 400)
priors <- tibble(
  name = c("uniform (Beta(1,1))", "sceptical (Beta(2,8))", "flat
  ↪  informative (Beta(4,4))"),
  a0   = c(1, 2, 4),
  b0   = c(1, 8, 4)
)
post <- priors |>
  rowwise() |>
  mutate(data = list(tibble(
    theta = theta,
    prior = dbeta(theta, a0, b0),
    posterior = dbeta(theta, a0 + k, b0 + n - k)
  ))) |>
  unnest(data) |>
  pivot_longer(c(prior, posterior), names_to = "density")
```

```
ggplot(post, aes(theta, value, colour = density)) +
  geom_line() + facet_wrap(~ name) +
  labs(x = expression(theta), y = "density")
```

10.38.7 3. Assumptions

Bernoulli trials, exchangeable, conjugate beta prior. With no prior data on the response rate in this population, we carry all three priors through and compare decisions.

10.38.8 4. Conduct

Posterior summaries.

```
summ <- priors |>
  rowwise() |>
  mutate(
    post_mean = (a0 + k) / (a0 + b0 + n),
    ci_lo = qbeta(0.025, a0 + k, b0 + n - k),
    ci_hi = qbeta(0.975, a0 + k, b0 + n - k),
    p_gt_0_2 = 1 - pbeta(0.2, a0 + k, b0 + n - k)
  ) |>
  dplyr::select(name, post_mean, ci_lo, ci_hi, p_gt_0_2)
summ
```

A frequentist analogue for comparison.

```
bt <- binom.test(k, n, p = 0.2, alternative = "greater")
c(estimate = bt$estimate, p_value = bt$p.value,
  ci_lo = bt$conf.int[1], ci_hi = bt$conf.int[2])
```

10.38.9 5. Conclude

> With 7 responders out of 20, the posterior probability that the response rate exceeds 0.2 is `round(summ$p_gt_0_2[1],2)` under a flat prior and `round(summ$p_gt_0_2[2],2)` under a sceptical prior. A one-sided exact binomial test against $p_0 = 0.2$ gives $p =$ `signif(bt$p.value,2)`.

All three priors point the same direction, but the strength of the conclusion depends on the prior. For a go/no-go decision, reporting the posterior probability of exceeding a decision threshold — along with its prior-sensitivity — is more decision-relevant than a p-value.

10.38.10 Common pitfalls

- Treating a 95% credible interval as a 95% confidence interval; they are calibrated for different inferences.
- Choosing a prior that pushes the posterior toward the desired decision and not reporting the sensitivity.
- Comparing a one-sided Bayesian tail probability to a two-sided p-value.

10.38.11 Further reading

- Spiegelhalter DJ, Abrams KR, Myles JP, *Bayesian Approaches to Clinical Trials and Health-Care Evaluation*.
- McElreath R, *Statistical Rethinking*, ch. 2–3.

10.38.12 Session info

```
sessionInfo()
```

10.38.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

10.39 Week 3, Session 2 — brms / Stan, LOO, hierarchical models

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

10.39.1 Learning objectives

- Write a simple multilevel model in `brms` syntax and describe each prior.
- Compare two candidate models using leave-one-out (LOO) cross- validation.
- Perform a prior predictive check and interpret it.

10.39.2 Prerequisites

Beta-binomial Bayesian intuition; linear mixed-effects models.

10.39.3 Background

`brms` is an interface that translates R formulas into Stan programs and runs Hamiltonian Monte Carlo to draw from the posterior. It is particularly strong for multilevel (hierarchical) models, where the benefits of Bayesian computation

— full posterior uncertainty, proper propagation into predictions — show up most clearly.

LOO cross-validation, implemented via the `loo` package, estimates out-of-sample predictive accuracy from the posterior without refitting, using Pareto-smoothed importance sampling. It is a modern replacement for AIC/BIC in Bayesian model comparison, and comes with diagnostics that warn when the approximation is unreliable.

Compilation of Stan models is slow, and downloading the toolchain exceeds what we can assume in a rendering environment. We therefore show the `brms` code with `#|eval:false` and walk through prior predictive simulation and a small frequentist counterpart that runs end-to-end.

10.39.4 Setup

```
library(tidyverse)
library(lme4)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

10.39.5 1. Goal

Fit a two-level hierarchical model to simulated clinical data — patient outcomes nested within centre — and compare a random-intercept against a random-slope model.

10.39.6 2. Approach

Simulate 20 centres with varying mean outcomes and varying dose effects.

```
n_centre <- 20; per_centre <- 15
centre <- rep(seq_len(n_centre), each = per_centre)
alpha0 <- rnorm(n_centre, 0, 1)
beta0  <- rnorm(n_centre, 0.5, 0.3)
dose <- rnorm(n_centre * per_centre)
y <- alpha0[centre] + beta0[centre] * dose + rnorm(n_centre * per_centre,
 ↪  0, 0.7)
d <- tibble(y = y, dose = dose, centre = factor(centre))
ggplot(d, aes(dose, y, colour = centre)) +
  geom_point(alpha = 0.5) + geom_smooth(method = "lm", se = FALSE) +
  theme(legend.position = "none")
```

10.39.7 3. Execution

A classical `lmer` random-intercept-and-slope fit as a stand-in we can run.

```
fit_lmer_ri  <- lmer(y ~ dose + (1 | centre), data = d)
fit_lmer_rs  <- lmer(y ~ dose + (dose | centre), data = d)
summary(fit_lmer_rs)$varcor
anova(fit_lmer_ri, fit_lmer_rs)
```

Prior predictive simulation (frequentist-style) for the Bayesian counterpart we are about to sketch.

```
prior_sim <- replicate(20, {
  a <- rnorm(1, 0, 1); b <- rnorm(1, 0, 1); sd_a <- abs(rnorm(1, 0, 1))
  a + b * dose + rnorm(length(dose), 0, sd_a)
})
ggplot(tibble(y = as.numeric(prior_sim)), aes(y)) +
  geom_histogram(bins = 40, fill = "grey60", colour = "white") +
  labs(x = "y drawn under priors only", y = "count")
```

The `brms` counterpart.

```
library(brms)
priors <- c(
  prior(normal(0, 2), class = "Intercept"),
  prior(normal(0, 1), class = "b"),
  prior(exponential(1), class = "sd"),
  prior(exponential(1), class = "sigma")
)
m_ri <- brm(y ~ dose + (1 | centre), data = d,
            prior = priors, chains = 2, iter = 2000, refresh = 0)
m_rs <- brm(y ~ dose + (dose | centre), data = d,
            prior = priors, chains = 2, iter = 2000, refresh = 0)
loo_compare(loo(m_ri), loo(m_rs))
```

10.39.8 4. Check

Compare to `lmer` and report pooled vs unpooled estimates.

```
fixef_lmer <- fixef(fit_lmer_rs)
fixef_lmer
```

10.39.9 5. Report

> On simulated hierarchical data (20 centres × 15 patients), a random-slope `lmer` model recovered an average dose effect of `round(fixef_lmer["dose"],2)` (SE from `lmer`), with substantial centre-to-centre variation. The `brms` sketch shown in the accompanying code would fit the same structure with explicit priors, and `loo_compare` would rank the random-slope model against the random-intercept model.

In practice, the Bayesian and frequentist fits should agree closely when priors are weakly informative and n is large; the Bayesian route adds explicit prior sensitivity and full posterior uncertainty for downstream prediction.

10.39.10 Common pitfalls

- Fitting a flat prior in a small-sample hierarchical model and getting a divergent posterior on variance components.
- Comparing LOO elpd differences smaller than about 4 SE without noting the uncertainty.
- Forgetting to scale continuous predictors; default priors assume normalised inputs.

10.39.11 Further reading

- Bürkner P-C (2017), *brms: An R package for Bayesian multilevel models using Stan.*
- Vehtari A, Gelman A, Gabry J (2017), *Practical Bayesian model evaluation using leave-one-out cross-validation.*

10.39.12 Session info

```
sessionInfo()
```

10.39.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 11

Survival Analysis

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Time-to-event methods: Kaplan-Meier, Cox regression, time-varying covariates, competing risks, multistate models, and survival-targeted machine learning. Censoring is treated as a first-class concept, not a footnote.

This chapter contains **30 method pages** and **5 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

11.1 Method pages

Method	Source slug
Accelerated Failure Time Models	`accelerated-failure-time`
Censoring Types	`censoring-types`
Checking Cox Assumptions	`cox-assumptions-check`
Competing Risks: Cumulative Incidence	`competing-risks-cif`
Concordance and the C-Index	`concordance-c-index`
Conditional Survival	`conditional-survival`
Cox Proportional Hazards Regression	`cox-proportional-hazards`
Fine-Gray Subdistribution Hazards	`fine-gray-regression`
Frailty Models	`frailty-models`
Hazard, Survival, and Cumulative Hazard	`hazard-survival-cumulative`
Interval-Censored Data	`interval-censored-data`
Joint Longitudinal-Survival Models	`joint-longitudinal-survival`
Kaplan-Meier Estimation	`kaplan-meier`
Landmark Analysis	`landmark-analysis`

Method	Source slug
Left Truncation	`left-truncation`
Log-Rank Test: Details	`log-rank-test-details`
Multi-State Models	`multi-state-models`
Parametric Survival: Log-Normal	`parametric-survival-lognormal`
Parametric Survival: Weibull	`parametric-survival-weibull`
Recurrent Events	`recurrent-events`
Restricted Mean Survival Time (RMST)	`restricted-mean-survival-time`
Royston-Parmar Flexible Parametric Models	`flexible-parametric-royston-parmar`
Schoenfeld Residuals	`schoenfeld-residuals`
Simulating Survival Data	`survival-simulation`
Stratified Cox Models	`stratified-cox`
The Brier Score for Survival	`brier-score-survival`
The Nelson-Aalen Estimator	`nelson-aalen-estimator`
Time-Dependent ROC	`time-dependent-roc`
Time-Varying Covariates	`time-varying-covariates`
Weighted Log-Rank Tests	`weighted-log-rank`

11.2 Labs

Lab
Survival primer: KM, log-rank, Cox PH
Time-varying covariates and immortal-time bias
Competing risks and multistate models
Survival ML (random survival forest, DeepSurv)
Time-dependent Brier, IPA, external validation

11.3 Accelerated Failure Time Models

11.3.1 Introduction

Accelerated failure time (AFT) models express covariate effects directly on the *time scale*: each covariate scales the expected survival time by a multiplicative factor. The model is $\log T = x^\top \beta + \sigma W$ with W a standardised noise (extreme-value, Normal, or logistic, depending on the family). The exponentiated coefficient $\exp(-\beta_j)$ is an "acceleration factor" — the multiplier on the time scale per unit increase in x_j. AFT is the natural framework when communicating effects to clinical audiences who reason in survival times rather than hazards, and it remains valid under non-proportional hazards where Cox regression fails.

11.3.2 Prerequisites

A working understanding of parametric survival distributions (Weibull, log-Normal, log-logistic) and the difference between hazard-ratio and acceleration-factor interpretations.

11.3.3 Theory

Common AFT families:

- **Weibull**: log-time errors are extreme-value distributed; this is the unique family with both PH and AFT interpretations, with $\beta_{\text{PH}} = -k\beta_{\text{AFT}}$ where k is the Weibull shape.
- **Log-Normal**: errors are Normal on the log-time scale; produces non-monotone hazards.
- **Log-logistic**: errors are logistic on the log-time scale; produces non-monotone hazards with heavier tails than log-Normal and a closed-form quantile function.
- **Generalised Gamma**: a three-parameter family that nests Weibull, log-Normal, and exponential; useful for selecting among nested AFT alternatives via likelihood-ratio tests.

The acceleration-factor interpretation is direct: $\exp(\beta_j) = 0.7$ means survival times are 30 % shorter for each unit increase in x_j; $\exp(\beta_j) = 1.3$ means 30 % longer.

11.3.4 Assumptions

The chosen parametric family is correct (verifiable through residual diagnostics and AIC comparisons), independent non-informative censoring, and the absence of unmodelled time-varying covariate effects.

11.3.5 R Implementation

```
library(survival); library(flexsurv)

data(lung)

# Weibull AFT
aft_w <- survreg(Surv(time, status) ~ age + sex, data = lung,
                 dist = "weibull")
summary(aft_w)

# Log-normal AFT
aft_ln <- survreg(Surv(time, status) ~ age + sex, data = lung,
                  dist = "lognormal")
```

```
# flexsurv alternative
flex_w <- flexsurvreg(Surv(time, status) ~ age + sex, data = lung, dist =
 ↪  "weibull")
flex_w

# AFT interpretation
exp(coef(aft_w)[-1])   # acceleration factors (time ratio)
```

11.3.6 Output & Results

`survreg()` returns AFT coefficients on the log-time scale plus the scale parameter; `flexsurvreg()` provides equivalent estimates with cleaner prediction interfaces. Exponentiating the slope coefficients gives acceleration factors interpretable as time ratios.

11.3.7 Interpretation

A reporting sentence: "A Weibull AFT model estimated that each additional year of age accelerated the event by a factor of 0.98 (95 % CI 0.97 to 1.00) — equivalently, a 2 % shorter expected time per year — and females had 30 % longer expected survival than males (acceleration factor 1.30, 95 % CI 1.06 to 1.59)." Choose the AFT-vs-PH framing based on the audience: clinicians who think in survival times prefer acceleration factors, while those used to hazard ratios prefer PH.

11.3.8 Practical Tips

- Weibull AFT is equivalent to Weibull PH up to a fixed transformation; choose the parameterisation that matches the audience.
- Use AIC across distributions (exponential, Weibull, log-Normal, log-logistic, generalised gamma) to select the family; AICs differ substantially when the hazard shape varies.
- Check goodness of fit with Cox-Snell residuals (`flexsurv::resid()`) — a Cox-Snell residual against exp(1) should follow the unit exponential.
- The log–log survival plot is the standard diagnostic for Weibull suitability; clear non-linearity rules it out and motivates a different family.
- For non-monotone hazards (log-Normal, log-logistic), AFT is mandatory — there is no PH analogue without a time-varying coefficient.
- For very small samples, AFT models are more efficient than Cox because the parametric structure adds information; for large samples with PH, Cox is competitive without distributional assumptions.

11.3.9 R Packages Used

`survival` for `survreg()` with multiple AFT families; `flexsurv` for `flexsurvreg()` with cleaner prediction and richer family support including generalised gamma; `eha` for additional parametric extensions; `rstpm2` for spline-based parametric models that bridge AFT and Royston-Parmar.

11.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.3.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.4 The Brier Score for Survival

11.4.1 Introduction

The Brier score for survival is a prediction-loss metric that combines calibration and discrimination into one number, evaluated at a chosen time horizon t. It generalises the binary Brier score to right-censored data via inverse-probability-of-censoring weights, so a model that predicts a survival probability $\hat{S}_i(t)$ for each subject can be assessed against the actual observed status at t. Unlike the C-index, which captures only ranking, the Brier score penalises miscalibrated probabilities — a model that ranks well but consistently overpredicts survival will score worse on Brier than on concordance.

11.4.2 Prerequisites

A working understanding of survival predictions, calibration vs discrimination, censoring, and the binary Brier score $(\hat{p} - y)^2$.

11.4.3 Theory

At time t, the censoring-weighted Brier score is

$$\text{BS}(t) = \frac{1}{n}\sum_{i=1}^{n} w_i(t)(Y_i(t) - \hat{S}_i(t))^2,$$

where $Y_i(t) = \mathbf{1}\{T_i > t\}$ is the observed survival indicator (when defined) and $w_i(t)$ is an inverse-probability-of-censoring weight that adjusts for the missing $Y_i(t)$ when subject i is censored before t. The integrated Brier score (IBS, also called the continuous ranked probability score for survival) averages $\text{BS}(t)$ over a follow-up window and gives a single-number summary suited to model selection.

A useful reference is the Brier score of the marginal Kaplan-Meier estimator (no covariates); a model that delivers a lower IBS than the null KM has produced informative predictions.

11.4.4 Assumptions

The censoring weights $w_i(t)$ are correctly estimated, typically via the Kaplan-Meier of the censoring distribution, which assumes non-informative censoring conditional on covariates.

11.4.5 R Implementation

```r
library(pec); library(survival)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex, data = lung, x = TRUE, y =
 ↪  TRUE)
pec_out <- pec(list(fit), data = lung,
               formula = Surv(time, status) ~ 1,
               times = seq(100, 800, by = 100))

plot(pec_out)
crps(pec_out)
```

11.4.6 Output & Results

`pec()` returns time-dependent Brier scores at the requested horizons together with the null KM benchmark. `crps()` returns the integrated Brier score (continuous ranked probability score) over the requested window. The plot overlays the model and reference Brier curves; a model curve below the KM curve at every t represents predictive improvement throughout follow-up.

11.4.7 Interpretation

A reporting sentence: "The Cox model achieved an integrated Brier score of 0.16 over 100–800 days, compared to 0.21 for the null Kaplan-Meier; predictive

improvement was largest between 200 and 500 days, where the model's time-specific Brier was about 30 % below the reference." Pair the IBS with a C-index; together they cover both calibration and discrimination.

11.4.8 Practical Tips

- Compare the model to the null KM at every time point you report; without that reference, an absolute Brier score is hard to interpret.
- Time-dependent Brier curves reveal when the model is most and least informative; this often identifies follow-up windows where extra covariates would help.
- For honest assessment, use cross-validation: `pec(...,splitMethod="cv10")` performs 10-fold CV and adjusts for optimism.
- Integrated Brier (IBS) is a single-number summary suited to model ranking; report it alongside per-time-point Brier curves for transparency.
- Complement with calibration plots (predicted vs observed survival probabilities at t) and the C-index; the three together give a full predictive-performance picture.
- Brier-score weights depend on a censoring model; if censoring depends on covariates, fit a Cox model for the censoring time and use it instead of the marginal KM.

11.4.9 R Packages Used

`pec` for `pec()`, `crps()`, and prediction-error curves; `riskRegression` for unified discrimination, calibration, and Brier metrics with bootstrap CIs; `survival` for the Cox fit; `timeROC` for time-dependent AUCs that complement the Brier-score view.

11.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.4.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.5 Censoring Types

11.5.1 Introduction

Censoring is the defining feature of survival data: the exact event time is unknown but partially observed, because follow-up ended, the event was screened periodically, or the subject was lost to observation. Handling censoring correctly is what separates survival analysis from ordinary regression on time-to-event endpoints. The wrong censoring assumption is one of the most consequential analytic errors in clinical studies, so identifying the censoring type and choosing methods that respect it is the first step of any time-to-event workflow.

11.5.2 Prerequisites

A working understanding of time-to-event concepts, follow-up windows, and the difference between an event and an end-of-follow-up timestamp.

11.5.3 Theory

The main censoring categories:

- **Right censoring**: the event had not yet occurred when observation ended. The most common type, the default in clinical trials.
- **Left censoring**: the event occurred before the first observation; the exact time is unknown but is in $[0, t_{\text{first observation}}]$.
- **Interval censoring**: the event occurred within a known interval $(L, R]$, typical in studies with periodic check-ups.

Two design-based subtypes of right censoring:

- **Type I**: fixed follow-up time; censoring time is identical for all administratively censored subjects.
- **Type II**: follow until a pre-specified number of events occur; censoring time depends on the data realisation.

Informative censoring is the most dangerous failure mode: the censoring mechanism is related to the underlying event risk (e.g., subjects drop out because they are doing badly). It violates the independence assumption that nearly every standard method requires and can severely bias estimates without warning.

11.5.4 Assumptions

Non-informative censoring conditional on covariates included in the model. If censoring depends on prognosis through unobserved variables, sensitivity analyses (informative-censoring bounds, multiple imputation) are needed.

11.5.5 R Implementation

```r
library(survival)

# Right-censored data: time and status
Surv(time = c(5, 8, 12, 6), event = c(1, 1, 0, 1))

# Interval-censored: event occurred between left and right
Surv(time = c(5, 8), time2 = c(10, 12), type = "interval2")

# Counting-process style (time-varying covariates): start, stop, status
Surv(time = c(0, 5, 10), time2 = c(5, 10, 15), event = c(0, 0, 1),
     type = "counting")
```

11.5.6 Output & Results

`Surv` objects display censoring through a `+` suffix on right-censored times, brackets for interval-censored intervals, and pairs for counting-process intervals. Every downstream survival function (`survfit`, `coxph`, `survreg`) takes a `Surv` object as the response and dispatches on its censoring type.

11.5.7 Interpretation

A reporting sentence: "Of 228 patients, 165 (72 %) experienced the event and 63 (28 %) were administratively right-censored at study end (median follow-up 310 days, range 5 to 1,022). No interval-censored or left-censored events were present. Censoring is assumed non-informative; a sensitivity analysis under tipping-point assumptions confirmed conclusions are robust to plausible departures." Always quantify censoring and discuss its assumed mechanism.

11.5.8 Practical Tips

- Right censoring is the default; specifying other types requires the `type` argument to `Surv()`.
- Interval-censored data require `icenReg` or `survival` with `Surv(L,R,type="interval2")`; treating intervals as exact event times biases estimates.
- Check for informative censoring by comparing baseline characteristics of censored vs event subjects; large differences are a warning sign.
- Distinguish administrative censoring (study end) from loss to follow-up (subject left); the two have different mechanisms and warrant separate sensitivity analyses.
- Plot censoring patterns: a histogram of censoring times alongside event times often reveals administrative cut-offs and dropout clusters that affect interpretation.
- For competing risks, treating competing events as censoring is itself a censoring choice; cumulative incidence analysis avoids it.

11.5.9 R Packages Used

`survival` for `Surv()` with all censoring types and the standard estimators; `icenReg` for interval-censored regression; `cmprsk` and `tidycmprsk` for competing-risks alternatives; `epi` for tools targeted at epidemiological censoring scenarios.

11.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.5.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.6 Competing Risks: Cumulative Incidence

11.6.1 Introduction

A competing risk is an event whose occurrence prevents the event of interest from happening — death from a non-disease cause prevents disease-specific death, retirement prevents return-to-work, and so on. In this setting the standard Kaplan-Meier estimator overestimates the cumulative incidence of the event of interest, because it treats competing events as censoring rather than as the absorbing states they actually are. The cumulative incidence function (CIF), estimated non-parametrically by Aalen-Johansen, is the correct summary: it gives the probability of having experienced each cause by time t in the presence of all other competing causes.

11.6.2 Prerequisites

A working understanding of survival analysis, censoring, and the Kaplan-Meier estimator; familiarity with multi-state thinking ("alive → cause-1 event" vs "alive → cause-2 event") helps.

11.6.3 Theory

The CIF for cause k is

$$\mathrm{CIF}_k(t) = \mathrm{P}(T \le t, \text{cause} = k) = \int_0^t S(u^-) h_k(u)\, \mathrm{d}u,$$

where h_k is the cause-specific hazard and S is the overall (any-cause) survival. Summing CIFs across causes recovers $1 - S(t)$. The Aalen-Johansen estimator generalises Nelson-Aalen for multi-state processes; Gray's test compares CIFs across groups, with the null that the CIF for the event of interest is equal across groups. For regression, the Fine-Gray subdistribution-hazards model targets covariate effects on the CIF directly, whereas cause-specific Cox models target effects on the cause-specific hazard — the two answer different scientific questions.

11.6.4 Assumptions

Non-informative censoring (independent of competing causes), only one event per subject (the first event ends follow-up), and accurate cause classification.

11.6.5 R Implementation

```
library(cmprsk)

# Simulated: event causes 1 (event of interest), 2 (competing), 0
 ↪  (censored)
set.seed(2026)
t_time <- rexp(300, 0.3)
cause <- sample(0:2, 300, replace = TRUE, prob = c(0.3, 0.5, 0.2))
group <- factor(sample(c("A", "B"), 300, replace = TRUE))

fit <- cuminc(t_time, cause, group)
plot(fit, xlab = "Time", ylab = "Cumulative incidence",
     color = c("red", "blue", "green", "orange"))

# Gray's test for equality of CIFs
print(fit)
```

11.6.6 Output & Results

`cuminc()` returns CIF estimates for each combination of cause and group, plus Gray's test χ^2 statistics for equality across groups within each cause. The plot overlays the four CIF curves; cumulative incidences for the two causes within each group sum to less than the corresponding $1 - S$ at every time.

11.6.7 Interpretation

A reporting sentence: "The five-year cumulative incidence of event 1 was 25 % in group A and 40 % in group B (Gray's test $p = 0.01$); event 2 incidences were 20 % vs 18 % ($p = 0.65$). KM analysis would have estimated event-1 incidence in group B at 52 %, illustrating the upward bias of ignoring the competing cause." Always present CIFs for every competing cause and explain why KM is inappropriate.

11.6.8 Practical Tips

- Never report $1 - \widehat{S}(t)$ from Kaplan-Meier as "cumulative incidence" in competing-risks settings; it overestimates the true cumulative incidence.
- Plot the CIF for every competing cause so readers see the partition of the overall event probability across causes.
- For regression effects on the CIF (e.g., "this covariate increases the cumulative incidence of cause 1"), use Fine-Gray subdistribution hazards (`cmprsk::crr` or `riskRegression::FGR`).
- For mechanistic effects on cause-specific hazards (e.g., "this drug accelerates cause-1 events but the drop in event-1 cumulative incidence is driven by competing event 2"), use cause-specific Cox models.
- Gray's test compares CIFs; the log-rank test on cause-1 events with cause-2 events censored compares cause-specific hazards. Report both when they disagree.
- For visualisation in `ggplot2`, `tidycmprsk` and `ggsurvfit` provide tidy interfaces with cleaner output than base `cmprsk::plot.cuminc()`.

11.6.9 R Packages Used

`cmprsk` for `cuminc()` and `crr()` (Fine-Gray); `tidycmprsk` for tidy-style competing-risks workflows; `riskRegression` for `FGR()`, `CSC()` (cause-specific Cox), and unified prediction interfaces; `ggsurvfit` for ggplot-friendly CIF plots.

11.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.6.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models

- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.7 Concordance and the C-Index

11.7.1 Introduction

Harrell's concordance index (C-index) generalises the ROC AUC to censored survival data. It is the probability that, for a randomly chosen pair of subjects, the one with the higher model-predicted risk actually fails first. The C-index is therefore the headline measure of *discrimination* for any survival model — Cox regression, parametric survival, machine-learning predictors — and the default reporting standard alongside the model's coefficients in clinical prediction-model papers.

11.7.2 Prerequisites

A working understanding of survival analysis, Cox regression, censoring, and the AUC for binary classifiers; familiarity with the distinction between calibration and discrimination.

11.7.3 Theory

Define a pair (i, j) as *comparable* if it can be determined which subject failed first — at least one observed event with the other subject either failing later or surviving past the first failure. The C-index is the proportion of comparable pairs for which the higher predicted risk corresponds to the earlier observed event:

$$C = \mathrm{P}(\hat{\eta}_i > \hat{\eta}_j \mid T_i < T_j, \delta_i = 1).$$

$C = 0.5$ corresponds to random ranking and $C = 1.0$ to perfect discrimination. Variants address specific limitations of Harrell's original index: Uno's C-statistic uses inverse-probability-of-censoring weights for unbiased estimation under heavy or non-uniform censoring; time-dependent AUCs (Heagerty, Uno) describe how discrimination changes across the follow-up period; the integrated Brier score is a complementary calibration-discrimination compound.

11.7.4 Assumptions

Risk scores come from a survival model whose linear predictor $\hat{\eta} = X\hat{\beta}$ is a meaningful ranking; the censoring mechanism is non-informative for Harrell's C and may be informative for Uno's C up to its weighting.

11.7.5 R Implementation

```r
library(survival)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
summary(fit)$concordance

# Time-dependent c-index via survcomp or timeROC
```

11.7.6 Output & Results

`summary(fit)$concordance` returns Harrell's C-index together with its standard error. For richer diagnostics, `survcomp::concordance.index()` adds bootstrap confidence intervals and `timeROC::timeROC()` returns time-dependent AUCs at user-specified landmarks.

11.7.7 Interpretation

A reporting sentence: "The Cox model achieved a Harrell's C-index of 0.63 (95 % CI 0.56 to 0.70) on the development data, indicating modest discrimination; on the held-out validation cohort the C-index dropped to 0.59, suggesting some optimism that the bootstrap procedure did not fully correct." Always pair the development-set C with an external or cross-validated estimate; in-sample concordance is optimistic.

11.7.8 Practical Tips

- $C > 0.7$ is conventionally considered acceptable for clinical use, $C > 0.8$ good; thresholds depend on the application and on what alternative classifiers achieve.
- Use Uno's C-statistic (`survAUC::UnoC()`) when censoring is heavy or non-uniform; Harrell's C can be biased upward in those settings.
- Report bootstrap confidence intervals via `survcomp::concordance.index()`; the analytical SE in `coxph` output is conservative.
- Time-dependent AUCs (`timeROC::timeROC()`) complement the overall C-index by showing when discrimination is highest and lowest across follow-up.
- For external validation, compute the C-index on an independent dataset; a development-set C without validation should never be the only discrimination measure reported.
- Pair discrimination (C-index) with calibration (calibration-in-the-large, calibration slope, Brier score); a model can rank well but predict poorly calibrated absolute probabilities.

11.7.9 R Packages Used

`survival` for `coxph()` with built-in concordance computation; `survcomp` for `concordance.index()` and bootstrap CIs; `timeROC` for time-dependent AUCs; `pec` for prediction-error curves and integrated Brier scores; `riskRegression` for unified discrimination and calibration metrics.

11.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.7.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.8 Conditional Survival

11.8.1 Introduction

Conditional survival is the probability of surviving an additional time t given that a patient has already survived to time s. It answers the clinically relevant question that absolute survival cannot: a five-year survival probability of 30 % at diagnosis says little to a patient who has already lived three years. The conditional five-year-from-now estimate may be 70 % or higher, dramatically updating the prognosis. Survivorship clinics, oncology long-term follow-up, and patient counselling all rely on conditional rather than unconditional survival.

11.8.2 Prerequisites

A working understanding of the survival function $S(t)$, conditional probability, and the Kaplan-Meier estimator.

11.8.3 Theory

By the definition of conditional probability,

$$\mathrm{P}(T > t + s \mid T > s) = \frac{\mathrm{P}(T > t + s)}{\mathrm{P}(T > s)} = \frac{S(t + s)}{S(s)}.$$

A patient who has survived s years is now in the risk set at s; their next-t-year survival is the ratio of survival probabilities. For Kaplan-Meier estimates, this is computed from $\widehat{S}(s)$ and $\widehat{S}(t+s)$ directly. Confidence intervals require variance propagation through the ratio; the log or log-log transformation gives intervals that respect the $[0, 1]$ bounds.

11.8.4 Assumptions

A reliable Kaplan-Meier or parametric survival estimate at both s and $t+s$; in practice this means sufficient follow-up so that $\widehat{S}(t+s)$ is not based on a tiny remaining cohort.

11.8.5 R Implementation

```r
library(survival)

data(lung)
fit <- survfit(Surv(time, status) ~ 1, data = lung)

# Unconditional 2-year survival
summary(fit, times = c(500, 1000))$surv   # approximate 2- and 3-year

# Conditional 1-year survival given 1-year survivorship
times <- summary(fit, times = c(365, 730))
cond_surv <- times$surv[2] / times$surv[1]
cond_surv
```

11.8.6 Output & Results

The ratio of two KM survival estimates at chosen landmark times. For confidence intervals, transform onto the log-log scale, compute SEs by the delta method, and back-transform. The `condsurv` package automates the computation including bootstrap intervals.

11.8.7 Interpretation

A reporting sentence: "Unconditional one-year survival was 48 %; conditional on having survived one year, the probability of surviving an additional year was 66 % (95 % CI 56 % to 75 %); conditional on having survived two years, the next-year survival rose to 78 %." Always report the landmark time s explicitly; conditional survival is meaningless without it.

11.8.8 Practical Tips

- Report conditional survival at clinically meaningful landmarks: one-year, five-year, ten-year survivor checkpoints in oncology, post-transplant 90-

day survival in transplantation.
- `condsurv` automates the computation and produces clean tables and plots; it handles confidence intervals via the log-log transformation by default.
- Confidence intervals must respect the $[0, 1]$ bounds; the log-log transformation is the standard choice and is preferable to the naive Wald interval.
- For dynamic prediction (conditional survival as a function of s), landmark analyses or joint longitudinal-survival models give more refined estimates that incorporate post-baseline information.
- Conditional survival is particularly useful in patient counselling because it answers the question patients actually ask: "Given that I'm still alive, how much longer can I expect to live?" rather than the population-level survival reported at diagnosis.
- Pair conditional-survival reports with the underlying KM curve so readers can see how the conditional estimate evolves across the cohort's experience.

11.8.9 R Packages Used

`survival` for the Kaplan-Meier baseline; `condsurv` for tabular conditional survival with CIs; `survminer::ggsurvplot()` for KM curves to display alongside conditional summaries; `dynpred` and `JM` for dynamic prediction beyond the simple ratio of survival probabilities.

11.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.8.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.9 Checking Cox Assumptions

11.9.1 Introduction

A fitted Cox model is only as good as its assumptions: proportional hazards across covariate levels, the right functional form for continuous predictors, and

the absence of one or two highly influential observations driving the coefficient estimates. Three residual families address these in turn — Schoenfeld for proportional hazards, martingale for functional form, and DFBETA-style influence diagnostics for unduly leveraged observations. Reporting all three is now standard for any clinical-trial Cox analysis.

11.9.2 Prerequisites

A working understanding of Cox proportional-hazards regression, residual-based diagnostics, and the difference between the proportional-hazards assumption and the linearity-of-the-linear-predictor assumption.

11.9.3 Theory

- **Schoenfeld residuals** are computed at each event time as the difference between the observed covariate value and the expected value across the risk set. Under proportional hazards, they should have no systematic relationship with time. `cox.zph()` tests the correlation between scaled residuals and a time transform (default $g(t) = \log t$).
- **Martingale residuals** are observed minus expected events for each subject. Plotting them against a continuous covariate exposes non-linearity: a smooth trend indicates the linear-predictor assumption fails for that covariate.
- **Deviance residuals** normalise martingale residuals to make outlying observations visible on a roughly symmetric scale.
- **DFBETA** measures how much each coefficient changes when each individual observation is removed; large absolute values flag influential observations whose exclusion changes the conclusions.

11.9.4 Assumptions

A Cox model has been fitted, with sufficient events to estimate residuals reliably (typically at least 10 events per covariate, and substantially more for stable diagnostic plots).

11.9.5 R Implementation

```
library(survival); library(survminer)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)

# PH test
zph <- cox.zph(fit)
print(zph)
ggcoxzph(zph)
```

```r
# Linearity check for continuous age
ggcoxfunctional(Surv(time, status) ~ age,
  data = na.omit(lung[, c("time", "status", "age")]))

# Influential observations
ggcoxdiagnostics(fit, type = "dfbeta")
```

11.9.6 Output & Results

`cox.zph()` returns per-covariate χ^2 tests and a global test for proportional hazards. `ggcoxzph()` plots the smoothed scaled Schoenfeld residuals with a horizontal reference at the constant-HR null. `ggcoxfunctional()` plots martingale residuals against the continuous covariate with a smoother. `ggcoxdiagnostics()` displays DFBETA for each coefficient.

11.9.7 Interpretation

A reporting sentence: "Diagnostics for the Cox model showed no PH violations (Schoenfeld global $p = 0.08$, per-covariate $p > 0.10$), approximately linear effects for age (martingale residuals showed no systematic trend), and three influential observations (DFBETA > 0.1 in absolute value) that were investigated and retained after sensitivity analysis showed coefficients changed by less than 5 %." Always report the global PH test, the per-covariate tests, and any sensitivity analysis you ran.

11.9.8 Practical Tips

- If PH fails for a covariate, choose between stratification (`+strata(var)`) and a time-varying coefficient via `tt()` or `survSplit()`; the choice depends on whether the stratifier's effect is itself of interest.
- If linearity fails for a continuous covariate, add a restricted cubic spline (`rms::rcs()` or `splines::ns()`) instead of dichotomising; categorisation discards information.
- Influential observations identified by DFBETA should be inspected for data errors first; if real, rerun without them as a sensitivity analysis and report any material changes.
- The global Schoenfeld test aggregates per-covariate tests; a global non-significant result with one significant per-covariate test still warrants action on that covariate.
- For continuous covariates with non-linear effects, the martingale-residual plot against the variable is often more informative than any formal test.
- All three diagnostics should appear in a model-checking section; reporting only the model output without diagnostics is no longer acceptable in clinical-trial publications.

11.9.9 R Packages Used

`survival` for `cox.zph()`, `residuals()`, and `coxph()` with strata; `survminer` for `ggcoxzph()`, `ggcoxfunctional()`, `ggcoxdiagnostics()`, and ggplot-flavoured Cox diagnostics; `rms` for `cph()` with built-in spline and validation tools; `flexsurv` and `timereg` for parametric and additive-hazards alternatives when assumptions fail.

11.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.9.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.10 Cox Proportional Hazards Regression

11.10.1 Introduction

Cox proportional hazards regression is the most widely used survival regression model in clinical and epidemiological research. Its central insight is that hazard *ratios* between subjects can be estimated without specifying or estimating the baseline hazard $h_0(t)$. The partial likelihood, introduced by D. R. Cox in 1972, conditions on the risk set at each event time and depends only on the regression coefficients, leaving the baseline hazard unspecified. The result is a semi-parametric model that combines the interpretability of regression with the flexibility of non-parametric survival analysis.

11.10.2 Prerequisites

A working understanding of hazard functions, the proportional-hazards assumption, and the partial-likelihood concept.

11.10.3 Theory

The Cox model is

$$h(t \mid x) = h_0(t) \exp(x^\top \beta).$$

The partial likelihood over event times is

$$L_p(\beta) = \prod_{i:\delta_i=1} \frac{\exp(x_i^\top \beta)}{\sum_{j \in R(t_i)} \exp(x_j^\top \beta)},$$

where $R(t_i)$ is the risk set just before t_i. Maximising L_p produces $\hat{\beta}$ whose exponentiated components are hazard ratios. Tied event times require a handling rule: Breslow's approximation (default) is fast and consistent under no ties, Efron's is more accurate when ties are common, and the exact-marginal method is the gold standard for discrete-time data.

The model assumes the hazard ratio between any two covariate patterns is constant over time — proportional hazards. When this fails, options include stratification, time-varying coefficients, or switching to a parametric AFT model.

11.10.4 Assumptions

Proportional hazards across covariate levels, independent non-informative censoring, and the absence of unmodelled time-varying covariate effects. Adequate events-per-variable (EPV) ratios — at least 10–15 events per regression coefficient — for stable inference.

11.10.5 R Implementation

```r
library(survival); library(broom)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
tidy(fit, conf.int = TRUE, exponentiate = TRUE)

# Check PH assumption
cox.zph(fit)
```

11.10.6 Output & Results

`tidy(fit,exponentiate=TRUE)` returns a tibble of hazard ratios, confidence intervals, and Wald p-values for each covariate. `summary(fit)` adds the likelihood-ratio test, score test, and Wald global tests for the joint null. `cox.zph()` reports the proportional-hazards diagnostic for each covariate and globally.

11.10.7 Interpretation

A reporting sentence: "A multivariable Cox proportional-hazards model adjusted for age, sex, and ECOG performance status estimated that each additional year of age increased the hazard of death by 2 % (HR 1.02, 95 % CI 1.00

to 1.04, $p = 0.04$); female sex was associated with a 40 % lower hazard (HR 0.60, 95 % CI 0.42 to 0.85, $p = 0.004$). The proportional-hazards assumption was supported (global Schoenfeld test $p = 0.55$)." Always pair coefficient reporting with PH diagnostics.

11.10.8 Practical Tips

- Always check proportional hazards via `cox.zph()` after fitting; reporting Cox HRs without PH diagnostics is no longer acceptable in clinical-trial publications.
- Use Efron's tie-handling method (`ties="efron"`) for tied event times; the default in older software is Breslow, which is less accurate.
- Report HRs with 95 % confidence intervals; p-values alone are inadequate, and a CI that includes 1.0 is more interpretable than a $p > 0.05$.
- Small events-per-variable (EPV) ratios inflate standard errors and lead to unstable estimates; report the EPV alongside the model.
- For continuous covariates, check linearity in the log-hazard via martingale residuals or restricted cubic splines; categorising a continuous covariate to "fix" non-linearity discards information.
- For non-PH covariates, choose between stratification (if the stratifier is not of interest) and time-varying coefficients (if the time-varying effect is itself the question).

11.10.9 R Packages Used

`survival` for `coxph()`, `cox.zph()`, and `Surv()`; `broom` for `tidy()` and `glance()` summaries; `survminer` for ggplot-style diagnostic plots; `rms` for `cph()` with built-in spline and bootstrap-validation tools.

11.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.10.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.11 Fine-Gray Subdistribution Hazards

11.11.1 Introduction

In competing-risks data, two questions often co-exist: "what is this covariate's mechanistic effect on the event of interest" and "what is its effect on the population-level probability of experiencing that event by time t?" Cause-specific Cox regression answers the first; Fine-Gray subdistribution-hazards regression answers the second. The two questions can disagree dramatically — a covariate can leave the cause-specific hazard unchanged while still raising or lowering the cumulative incidence because it changes the competing-event hazard. Reporting only one when both are relevant is a common analytical mistake.

11.11.2 Prerequisites

A working understanding of competing-risks data, cumulative incidence functions, and the cause-specific vs subdistribution-hazards distinction.

11.11.3 Theory

The subdistribution hazard for cause 1 is

$$h_1^{\text{sub}}(t \mid x) = \lim_{\Delta t \to 0} \frac{\mathrm{P}(t \le T < t + \Delta t,\ C = 1 \mid T \ge t \text{ or } (T < t \text{ and } C \ne 1),\ x)}{\Delta t}.$$

The risk set keeps subjects who have already experienced a competing event, with diminishing weights. Under the Fine-Gray model

$$h_1^{\text{sub}}(t \mid x) = h_{10}(t) \exp(x^\top \beta),$$

the exponentiated coefficient $\exp(\beta_j)$ is a *subdistribution hazard ratio* and maps directly onto changes in cumulative incidence: a sHR > 1 means higher cumulative incidence of cause 1 per unit of covariate. The model is fitted by a modified partial likelihood with inverse-probability-of-censoring weights.

11.11.4 Assumptions

Proportional subdistribution hazards (an analogue of proportional cause-specific hazards), independent censoring (or correctly weighted), and accurate cause classification.

11.11.5 R Implementation

```r
library(cmprsk)

set.seed(2026)
t_time <- rexp(300, 0.3)
cause  <- sample(0:2, 300, replace = TRUE, prob = c(0.3, 0.5, 0.2))
x1     <- rnorm(300); x2 <- rbinom(300, 1, 0.5)

fg <- crr(ftime = t_time, fstatus = cause, cov1 = cbind(x1, x2),
          failcode = 1, cencode = 0)
summary(fg)
```

11.11.6 Output & Results

`crr()` returns subdistribution coefficients on the log-sHR scale, standard errors, p-values, and (after `summary()`) exponentiated subdistribution hazard ratios with confidence intervals. The output structure matches Cox regression but the interpretation refers to cumulative incidence rather than cause-specific hazard.

11.11.7 Interpretation

A reporting sentence: "Fine-Gray subdistribution-hazards regression for cause 1 estimated sHR = 1.45 for x_1 (95 % CI 1.10 to 1.90), meaning a 45 % higher cumulative incidence of cause 1 per unit increase in x_1; a cause-specific Cox model on the same data gave HR 1.32, indicating that part of the effect on cumulative incidence comes via the competing-event hazard." Always report both subdistribution and cause-specific HRs when they differ; they answer different questions.

11.11.8 Practical Tips

- Subdistribution hazard ratios do *not* equal cause-specific hazard ratios; treating them as interchangeable is a common error.
- Use cause-specific Cox for *etiologic* questions ("what affects the disease mechanism?"); use Fine-Gray for *prognostic* / prediction questions ("what affects the probability of disease over time?").
- For prediction tools (nomograms, risk calculators), Fine-Gray is the correct framework because absolute risk depends on cumulative incidence.
- Report both approaches when both questions matter; transparency about the difference prevents readers from misinterpreting one as the other.
- `riskRegression::FGR()` offers a modern alternative with cleaner diagnostics and better small-sample handling.
- Check proportional subdistribution hazards via `crskdiag` or simulated residual plots; the assumption can fail even when proportional cause-specific hazards holds.

11.11.9 R Packages Used

`cmprsk` for `crr()` and `cuminc()`; `riskRegression` for `FGR()`, `CSC()` (cause-specific Cox), and unified prediction interfaces; `tidycmprsk` for tidy-style competing-risks workflows; `survival` for Cox regression on cause-specific hazards as the complementary fit.

11.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.11.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.12 Royston-Parmar Flexible Parametric Models

11.12.1 Introduction

Royston-Parmar models bridge the gap between rigid parametric survival families (exponential, Weibull, log-normal) and the assumption-light Cox semi-parametric model. They place restricted cubic splines on the log cumulative hazard (or on related transforms) so the baseline can take essentially any shape the data demand, while keeping the full parametric likelihood that delivers smooth survival, hazard, and quantile predictions for any covariate pattern. In epidemiological work — particularly population-based cancer studies — they have become the preferred default because they combine Cox-like flexibility with parametric extrapolation.

11.12.2 Prerequisites

A working understanding of Cox regression, parametric survival families, and restricted cubic splines for smooth functional forms.

11.12.3 Theory

The Royston-Parmar model writes

$$\log H(t \mid x) = s(\log t; \gamma) + x^\top \beta,$$

where s is a restricted cubic spline in $\log t$ with knots placed at quantiles of the log event-time distribution. With k internal knots, the spline has $k + 1$ parameters; common choices are 1–4 knots. Setting `scale="hazard"` produces a proportional-hazards model; `scale="odds"` gives a proportional-odds parameterisation; `scale="normal"` gives a probit parameterisation, equivalent to a generalised log-normal model. Time-dependent effects can be modelled by interacting covariates with spline basis functions.

Predictions on the natural time scale are smooth and available at any horizon, including extrapolation beyond the maximum observed event time, where Cox predictions are undefined.

11.12.4 Assumptions

The chosen scale (PH, PO, or probit) is appropriate; the number of knots is not so large that the baseline is overfitted nor so small that the underlying shape is missed. AIC is the conventional knot-selection criterion.

11.12.5 R Implementation

```r
library(flexsurv)

data(lung)

# 2 internal knots (default placement at quantiles of log-event times)
fit <- flexsurvspline(Surv(time, status) ~ age + sex, data = lung,
                      k = 2, scale = "hazard")
fit
plot(fit, type = "hazard")
plot(fit, type = "survival")
```

11.12.6 Output & Results

`flexsurvspline()` returns spline coefficients $\gamma_0, \gamma_1, \ldots$, covariate effects $\hat{\beta}$ on the log-hazard scale (under PH), and a fully smooth model object. `plot(fit,type="hazard")` overlays the model-implied hazard on a non-parametric reference, and `plot(fit,type="survival")` does the same for the survival function. The smoothness is the headline visual difference from a Cox-based survfit step function.

11.12.7 Interpretation

A reporting sentence: "A Royston-Parmar PH model with two internal knots estimated the hazard ratio for female sex at 0.60 (95 % CI 0.42 to 0.85), close to the Cox estimate of 0.59 (0.42 to 0.83); the spline baseline captured a hazard peak around 9 months that the Weibull model missed entirely (AIC: Royston-Parmar 1{,}890, Weibull 1{,}908)." Reporting AIC across knots and against simpler parametric alternatives makes the choice transparent.

11.12.8 Practical Tips

- Start with 2–3 knots and compare AIC across 1–4; more than 4 internal knots tends to overfit small datasets.
- Use `scale="hazard"` (PH) when the goal matches a Cox interpretation; `"odds"` for a proportional-odds analogue; `"normal"` for a probit / log-normal-like fit.
- Coefficient estimates should be close to those from a Cox model under PH; large discrepancies indicate either Cox tied-event handling differences or an issue with the spline baseline.
- Time-dependent covariate effects come from `gamma1~x` interactions inside `flexsurvspline()`; this lets you test PH non-parametrically within the model.
- For predictions on a fine time grid, `summary(fit,type="survival",t=seq(0,1000,10))` produces smooth survival probabilities ready for plotting.
- In population-based cancer epidemiology, Royston-Parmar models have largely replaced Cox for relative-survival and excess-mortality work; the parametric baseline gives interpretable hazard shapes and clean extrapolation.

11.12.9 R Packages Used

`flexsurv` for `flexsurvspline()` with PH, PO, and probit scales; `rstpm2` for the related Stata-compatible Royston-Parmar implementation with extended time-dependent effects; `survival` for the Cox baseline against which to compare; `ggflexsurv` for ggplot-style plotting of `flexsurvspline` objects.

11.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.12.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.13 Frailty Models

11.13.1 Introduction

A frailty model adds an unobserved multiplicative effect — the frailty — to the hazard, allowing the survival analogue of a random effect. Frailty captures two things: unmeasured heterogeneity at the individual level (the population mixture of "robust" and "fragile" subjects with different baseline hazards), and clustered survival data where observations within a unit (family, hospital, patient with recurrent events) share an unobserved common factor. Without it, standard errors are wrong for clustered data, and population-level hazard ratios are biased toward the null because of the selection-of-survivors effect.

11.13.2 Prerequisites

A working understanding of Cox regression, random-effects models in linear and generalised linear settings, and the concept of unobserved heterogeneity.

11.13.3 Theory

The shared-frailty Cox model is

$$h_{ij}(t) = h_0(t) u_i \exp(x_{ij}^\top \beta),$$

where u_i is a positive random variable (frailty) shared across observations j within cluster i and conventionally constrained to have mean 1. The Gamma distribution is the textbook default — its conjugate structure with the partial likelihood makes estimation tractable — though log-Normal frailties are increasingly used when the Laplace-approximation likelihood is implemented.

The frailty variance $\theta = \mathrm{Var}(u_i)$ measures the strength of within-cluster correlation; $\theta = 0$ corresponds to no clustering and reduces to the standard Cox model. The likelihood-ratio test for $\theta = 0$ is on the boundary of the parameter space, so the conventional χ^2_1 reference is replaced by a 50-50 mixture of χ^2_0 and χ^2_1.

11.13.4 Assumptions

The frailty distribution (Gamma or log-Normal) is correctly specified, frailties are independent across clusters, and the standard Cox assumptions (PH within

cluster) hold.

11.13.5 R Implementation

```r
library(survival)

data(kidney)   # two events per patient (recurrent catheter infections)

# Shared frailty
fit <- coxph(Surv(time, status) ~ age + sex + frailty(id), data = kidney)
summary(fit)
```

11.13.6 Output & Results

`summary(fit)` returns the fixed-effect Cox coefficients (interpreted as conditional-on-frailty hazard ratios), the frailty variance estimate, and a likelihood-ratio test for the frailty term against the no-frailty null. The variance estimate is the headline measure of unobserved heterogeneity.

11.13.7 Interpretation

A reporting sentence: "A shared-Gamma frailty Cox model on 38 patients with two recurrent catheter-infection times each estimated a frailty variance of 0.41 (LRT $\chi^2_{0.5} = 6.8$, $p = 0.005$); after accounting for within-patient correlation, female sex was strongly protective (HR 0.36, 95 % CI 0.21 to 0.62)." Always report frailty variance with its significance test; a small variance suggests the frailty term is unnecessary.

11.13.8 Practical Tips

- For frailty variance close to zero, drop the frailty term and refit with the standard Cox model; the simpler fit is more efficient when no clustering is present.
- Use shared-frailty models for naturally clustered data (families, hospitals, repeated measures); individual frailties require recurrent events or external longitudinal information.
- Gamma frailty is the default choice for partial-likelihood implementations; log-Normal requires Laplace approximation and is implemented in `frailtypack` and `coxme`.
- Reported hazard ratios are *conditional* on the frailty (within-cluster); for marginal effects, integrate over the frailty distribution or use generalised estimating equations as an alternative.
- The LRT for $\theta = 0$ is on the parameter-space boundary; use the half-half mixture reference (or simulation-based p-values) to avoid conservative tests.

- For high-dimensional clustering or complex frailty structures, `coxme` provides a mixed-model interface compatible with `lme4`-style formulas.

11.13.9 R Packages Used

`survival` for `coxph()` with `frailty()`; `coxme` for mixed-effects Cox models with multiple frailties and complex random-effect structures; `frailtypack` for Gamma, log-Normal, and joint frailty models including recurrent events; `frailtyEM` for EM-based frailty estimation with cleaner SE handling.

11.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.13.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.14 Hazard, Survival, and Cumulative Hazard

11.14.1 Introduction

A time-to-event distribution admits three mutually equivalent representations: the survival function $S(t)$, the hazard function $h(t)$, and the cumulative hazard $H(t)$. They contain the same information but emphasise different aspects of the process — surviving probability, instantaneous risk, and accumulated risk respectively. Switching freely among them is one of the foundational fluencies of survival analysis, because each form has natural plotting and inferential conventions and most software exposes all three.

11.14.2 Prerequisites

A working understanding of survival analysis basics, the difference between probability and rate, and integration of non-negative functions.

11.14.3 Theory

For a non-negative continuous time-to-event T with density $f(t)$:

- **Survival function**: $S(t) = \mathrm{P}(T > t)$. The probability of surviving past time t.
- **Hazard function**: $h(t) = f(t)/S(t)$. The instantaneous event rate at time t given survival up to t. Hazard is a rate (per unit time), not a probability.
- **Cumulative hazard**: $H(t) = \int_0^t h(u)\,\mathrm{d}u = -\log S(t)$. Accumulated risk by time t.

Any one of the three determines the other two: $S(t) = e^{-H(t)}$, $h(t) = -\mathrm{d}\log S(t)/\mathrm{d}t$. The Kaplan-Meier estimator targets $S(t)$ directly; the Nelson-Aalen estimator targets $H(t)$ and is preferable in the right tail where the KM step function is unstable. The hazard itself is never estimated directly without smoothing, because instantaneous rates require kernel methods or parametric assumptions.

11.14.4 Assumptions

A non-negative, almost surely continuous time-to-event with proper density (or, for discrete time, a discrete analogue with the same mutual recoverability).

11.14.5 R Implementation

```r
library(survival)

data(lung)
fit <- survfit(Surv(time, status) ~ 1, data = lung)

# Survival curve
plot(fit, conf.int = TRUE, main = "S(t)")

# Cumulative hazard
plot(fit, fun = "cumhaz", main = "H(t)")
```

11.14.6 Output & Results

Two complementary curves: the Kaplan-Meier survival function with point-wise confidence bands, and the cumulative hazard estimated via $-\log \widehat{S}(t)$ (which agrees with Nelson-Aalen up to small-sample corrections). Reading both together makes the underlying hazard pattern visible — a steep $S(t)$ corresponds to a rising $H(t)$, and the slope of $H(t)$ at any time is the local hazard rate.

11.14.7 Interpretation

A reporting sentence: “Median survival was 310 days (95 % CI 280 to 360); cumulative hazard at one year was approximately 0.85, corresponding to $S(365) =$

$\exp(-0.85) = 0.43$, consistent with the KM estimate of 42 % at one year." Quoting both representations communicates the same fact in two complementary forms readers may prefer.

11.14.8 Practical Tips

- Hazard is a rate, not a probability; never describe a hazard ratio as "doubled probability" without converting through $S(t)$ or absolute differences.
- Compare cumulative hazards across groups when discriminating risk over follow-up; the log-rank test is essentially a comparison of cumulative hazards.
- Constant hazard implies exponential survival and a linear cumulative hazard; deviations from linearity in $H(t)$ are the easiest visual diagnostic for non-exponentiality.
- Kaplan-Meier and Nelson-Aalen estimators agree closely; prefer Nelson-Aalen when you specifically need $\hat{H}(t)$ in the right tail.
- Hazard ratios are interpretable only under proportional hazards; if the assumption fails, report cumulative incidence, restricted-mean survival time, or stratify the comparison.
- For modelling a smooth hazard, use parametric (Weibull, log-normal) or flexible (Royston-Parmar splines) models; don't try to estimate $h(t)$ nonparametrically without bandwidth selection.

11.14.9 R Packages Used

`survival` for `survfit()`, `Surv()`, and the standard estimators; `flexsurv` for parametric and Royston-Parmar spline-based models that yield smooth hazards; `muhaz` for kernel-smoothed hazard estimation; `survminer::ggsurvplot()` and `survminer::ggsurvplot_combine()` for ggplot-style display of the curves.

11.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.14.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.15 Interval-Censored Data

11.15.1 Introduction

Interval censoring arises whenever subjects are observed at discrete intervals and the exact event time is therefore unknown — only the interval that contains it is known. Cavity detected at a routine dental check-up, HIV seroconversion identified at a quarterly clinic visit, breakdown of a part discovered at periodic inspection: in each case the event time lies between the last "no event" inspection and the first "event observed" inspection. Treating these as exact event times biases standard survival methods; the Turnbull non-parametric maximum-likelihood estimator (NPMLE) and parametric or semi-parametric extensions handle the interval structure correctly.

11.15.2 Prerequisites

A working understanding of survival analysis, the distinction between right, left, and interval censoring, and basic Kaplan-Meier estimation as the gold-standard right-censored analogue.

11.15.3 Theory

Each subject contributes an interval $(L_i, R_i]$ within which the event occurred; right-censored subjects have $R_i = \infty$, exact events have $L_i = R_i$. Turnbull's algorithm finds the NPMLE survival function on a grid of unique left and right endpoints via an EM iteration: at each step it apportions probability mass across the candidate intervals proportional to current estimates and updates the survival estimate from those weighted contributions. Convergence yields the NPMLE up to indeterminacy in regions where the data provide no information (the so-called Turnbull intervals).

For regression, parametric models (Weibull, log-normal) and semi-parametric extensions (PH or PO with monotone splines on the baseline) accept interval-censored input directly. The likelihood is $\prod_i [S(L_i) - S(R_i)]$, which reduces to the standard product in the right-censored special case.

11.15.4 Assumptions

Independent interval censoring (the inspection schedule does not depend on the subject's underlying time-to-event), and accurate interval boundaries.

11.15.5 R Implementation

```
library(icenReg); library(survival)

data(miceData)
```

```r
head(miceData)

# NPMLE
fit_np <- ic_np(cbind(l, u) ~ grp, data = miceData)
plot(fit_np)

# Parametric: Weibull PH
fit_w <- ic_par(cbind(l, u) ~ grp, data = miceData, model = "ph", dist =
 ↪  "weibull")
summary(fit_w)
```

11.15.6 Output & Results

`ic_np()` returns the Turnbull NPMLE for each group with confidence bands and a step-function-style plot. `ic_par()` fits a parametric or semi-parametric regression with hazard-ratio interpretation under PH. The two complement each other: NPMLE for visual exploration, regression for covariate effects.

11.15.7 Interpretation

A reporting sentence: "Turnbull NPMLE showed lower survival in the contaminated group throughout follow-up (12-month survival 28 % vs 52 %); a Weibull PH model adjusted for sex estimated a hazard ratio of 2.1 (95 % CI 1.4 to 3.2)." When the NPMLE flatlines in regions with no information, describe these gaps in the caption rather than letting readers misinterpret the apparent constant survival.

11.15.8 Practical Tips

- `icenReg` is the most actively maintained interval-censored regression package in R; `ic_np()` for non-parametric, `ic_par()` for parametric, `ic_sp()` for semi-parametric with monotone splines.
- Exact event times in interval notation are encoded as $L = R$; right-censored subjects as $R = \infty$ or `Inf`. Most software accepts both encodings.
- The NPMLE has indeterminate intervals (regions of constant survival between Turnbull intervals); plot the NPMLE as a band rather than a single curve when visualising.
- For Cox-style proportional-hazards regression with interval-censored data, `survival::survreg(dist="weibull")` accepts a `Surv(L,R,type="interval2")` object directly.
- For very heavy interval censoring (long inspection intervals), consider sensitivity analyses with imputed event times at the midpoint or right endpoint and check that conclusions are robust.
- Standard Cox-Snell residuals do not directly apply; use modified residuals from `icenReg::diag_baseline()` for goodness-of-fit checks.

11.15.9 R Packages Used

`icenReg` for `ic_np()`, `ic_par()`, and `ic_sp()` interval-censored regression; `survival` for `Surv(...,type="interval2")` and `survreg()` interval-censored support; `Icens` for additional NPMLE algorithms; `intccr` for interval-censored competing risks.

11.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.15.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.16 Joint Longitudinal-Survival Models

11.16.1 Introduction

Joint longitudinal-survival models bring together two submodels — a mixed model for a longitudinally measured biomarker and a survival model for time to an event — via shared random effects so that the evolving biomarker is itself part of the hazard. This solves two problems simultaneously: the time-varying-covariate Cox model that uses last-observation-carried-forward biomarker values ignores measurement error and timing irregularity, and informative dropout (subjects censored because their biomarker is deteriorating) biases naive survival estimates. The joint formulation models both processes together and integrates over the shared random effects, yielding consistent hazard ratios for the underlying biomarker trajectory.

11.16.2 Prerequisites

A working understanding of linear mixed models, Cox regression, time-varying covariates, and the concept of informative dropout.

11.16.3 Theory

The longitudinal submodel is

$$y_{ij} = \mu_i(t_{ij}) + \varepsilon_{ij}, \qquad \mu_i(t) = X_{ij}^\top \gamma + Z_{ij}^\top b_i,$$

with $b_i \sim \mathcal{N}(0, D)$ subject-specific random effects. The survival submodel uses the latent trajectory:

$$h_i(t) = h_0(t) \exp\!\big(\alpha \mu_i(t) + x_i^\top \beta\big).$$

The association parameter α measures how strongly the underlying biomarker drives the hazard. The joint likelihood integrates the random effects out of both submodels simultaneously, typically via Gauss-Hermite quadrature; this is what makes joint models computationally heavier than fitting the two submodels separately.

11.16.4 Assumptions

Multivariate Normal random effects, correctly specified mean and covariance structures, no model misspecification in either submodel, and a proper proportional-hazards form (or a flexible alternative) for the survival component.

11.16.5 R Implementation

```
library(JM)

data(pbc2)
lm_fit <- lme(log(serBilir) ~ year + drug, random = ~ year | id,
              data = pbc2)
surv_fit <- coxph(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)

joint_fit <- jointModel(lm_fit, surv_fit, timeVar = "year",
                        method = "piecewise-PH-aGH")
summary(joint_fit)
```

11.16.6 Output & Results

`summary(joint_fit)` returns the longitudinal-submodel coefficients (fixed effects, random-effect covariance), the survival-submodel coefficients (β), and the association parameter α with its standard error and 95 % confidence interval. Diagnostics include the Gauss-Hermite quadrature convergence criterion and the integrated likelihood.

11.16.7 Interpretation

A reporting sentence: “A joint longitudinal-survival model linking log-bilirubin trajectory to time to death (PBC dataset, 312 subjects) estimated the association parameter at $\alpha = 0.74$ (SE $= 0.10$, $p < 0.001$); each unit increase in the

underlying log-bilirubin level corresponded to a 2.1-fold higher hazard of death, after accounting for measurement error and informative dropout." Always describe both submodels in the methods so the joint structure is reproducible.

11.16.8 Practical Tips

- Use `JM` for classical maximum-likelihood joint models; `JMbayes2` for Bayesian alternatives that handle complex random-effect structures and missing data more flexibly.
- Joint models handle informative dropout — subjects censored because their biomarker is deteriorating — that biases naive Cox time-varying-covariate analyses.
- Computational cost scales rapidly with the number of random effects; start with random intercept and slope, escalate only if diagnostics demand it.
- Dynamic predictions ("given the biomarker history to time s, what is the predicted survival from s?") come from `survfitJM()` and are the headline clinical use case.
- Convergence diagnostics matter: check `joint_fit$convergence` and Gauss-Hermite node counts; raise `iter.qN` and `nQuad` if the model fails to converge.
- A simpler alternative is the time-varying Cox model with LOCF imputation; it loses efficiency and ignores measurement error but is easier to fit and explain to a non-statistical audience.

11.16.9 R Packages Used

`JM` for `jointModel()` with maximum-likelihood estimation; `JMbayes2` for Bayesian joint models with richer random-effect structures; `joineRML` for multivariate longitudinal-survival models with multiple biomarkers; `dynpred` for dynamic predictions when joint modelling is computationally infeasible.

11.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.16.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.17 Kaplan-Meier Estimation

11.17.1 Introduction

The Kaplan-Meier (KM) estimator is the non-parametric maximum-likelihood estimator of the survival function from right-censored time-to-event data. Its signature step-function plot is the single most recognisable figure in clinical research. The estimator makes no distributional assumption about the time-to-event variable; it simply accumulates the conditional probability of surviving each observed event time.

11.17.2 Prerequisites

The reader should understand that a survival analysis studies time until an event, and that "right censoring" means some subjects are observed only up to a point before their event has occurred. Familiarity with R's `survival` package is not required but helps.

11.17.3 Theory

Let $t_1 < t_2 < \ldots < t_k$ denote the distinct event times observed in the data. At time t_j, let d_j be the number of events and n_j the number of subjects still at risk (not yet having had the event and not yet censored). The Kaplan-Meier estimator of the survival function is

$$\hat{S}(t) = \prod_{j : t_j \le t} \left(1 - \frac{d_j}{n_j}\right).$$

The estimator is a step function that drops at each event time by the fraction of the at-risk set experiencing the event. Censored subjects contribute to the risk set up to their censoring time but do not cause drops.

The standard-error estimator due to Greenwood is

$$\widehat{\mathrm{Var}}[\hat{S}(t)] = \hat{S}(t)^2 \sum_{j : t_j \le t} \frac{d_j}{n_j (n_j - d_j)}.$$

Pointwise confidence intervals are typically constructed on the complementary log-log scale to ensure they lie in $[0, 1]$.

The **log-rank test** compares survival between two or more groups. It sums the observed-minus-expected events at each event time across groups, yielding a test statistic that follows a χ^2_{k-1} distribution under the null hypothesis of equal survival functions.

11.17.4 Assumptions

Kaplan-Meier estimation assumes:

1. Censoring is **non-informative** – subjects who are censored at time t have the same future risk as those who remain under observation. Violated when, for example, patients drop out because they feel worse.
2. Subjects are independent.
3. Risk of event does not depend on calendar time beyond what covariates capture (important when accrual spans a long period).

11.17.5 R Implementation

```r
library(survival)
library(survminer)

data(lung, package = "survival")

fit <- survfit(Surv(time, status) ~ sex, data = lung)

print(fit)

ggsurvplot(fit,
           data = lung,
           conf.int = TRUE,
           pval = TRUE,
           risk.table = TRUE,
           legend.labs = c("Male", "Female"),
           xlab = "Days from diagnosis",
           ylab = "Survival probability",
           palette = "Set2")
```

The `Surv()` call packages the time and event-indicator columns into a single survival object; `survfit()` with a right-hand side of `sex` produces separate KM curves for each sex. `ggsurvplot()` draws the curves with pointwise confidence bands, a log-rank p-value, and a risk table below.

11.17.6 Output & Results

The `lung` dataset has 138 male and 90 female patients with advanced lung cancer. The KM median survival is 270 days for males (95% CI 212 to 310) and 426 days for females (95% CI 348 to 550). The log-rank test gives $\chi^2_1 = 10.3$, $p = 0.0013$, indicating a significantly higher survival for females.

11.17.7 Interpretation

Reporting follows the CONSORT conventions: "Median overall survival was 270 days (95% CI 212-310) for males and 426 days (95% CI 348-550) for females; log-rank $p = 0.001$ (Figure X)." The figure should always be shown with a risk table beneath it – the curve is nearly meaningless once the number at risk drops below a handful.

11.17.8 Practical Tips

- Always include a risk table. Survival curves at late follow-up with only 2 or 3 subjects remaining can swing wildly and mislead readers.
- Do not rely on the median alone as a comparison statistic; if the curves cross, the median can flip direction as the study matures.
- The log-rank test is most powerful when the proportional-hazards assumption holds. For curves that cross or diverge late, consider the Renyi (weighted) log-rank or a restricted mean survival time comparison.
- Do not attempt a KM estimate without an event in the at-risk set; the curve will simply be flat at 1.
- When follow-up is short relative to the event rate, many subjects will be censored and the KM estimate at late times will have wide confidence bands. Report the cumulative event count at a clinically relevant landmark (1-year, 5-year) rather than the tail of the curve.

11.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.17.10 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.18 Landmark Analysis

11.18.1 Introduction

Landmark analysis is the standard fix for the immortal-time bias that arises when subjects are classified by a post-baseline exposure (treatment response,

biomarker change, secondary event). Naively grouping subjects by a covariate measured after follow-up begins assigns "responder" status only to subjects who survived long enough to be assessed, automatically biasing comparisons in their favour. Landmark analysis sidesteps this by selecting a fixed landmark time, restricting to subjects alive and observable at that time, classifying them by covariate value *at the landmark*, and then analysing only the post-landmark survival.

11.18.2 Prerequisites

A working understanding of Cox regression, time-varying covariates, and the immortal-time bias that makes naive responder-vs-nonresponder analyses invalid.

11.18.3 Theory

At a chosen landmark time L:

1. Exclude subjects with an event before L — they cannot be classified by status at L.
2. Classify each remaining subject by their covariate value at L.
3. Treat L as the new time origin and analyse the residual survival $T_i - L$ for $T_i > L$.
4. Fit a standard Cox model on the post-landmark data.

Multiple landmarks can be combined into a "super model" with `id`-clustered variance and a smooth function of L as a covariate, gaining efficiency at the cost of additional model complexity (Van Houwelingen 2007).

11.18.4 Assumptions

The covariate value at landmark L is the relevant exposure for prognosis; sufficient follow-up beyond L remains; and the censoring mechanism after L is non-informative.

11.18.5 R Implementation

```
library(survival)

set.seed(2026)
n <- 200
d <- data.frame(
  id = 1:n,
  time = rexp(n, 0.1),
  status = rbinom(n, 1, 0.6),
  response = rbinom(n, 1, 0.5)      # binary time-varying covariate
)
```

```
L <- 6   # landmark at 6 months

# Subjects at risk at L
at_risk <- d[d$time >= L, ]
at_risk$time_from_L <- at_risk$time - L
at_risk$status_from_L <- at_risk$status

fit <- coxph(Surv(time_from_L, status_from_L) ~ response, data = at_risk)
summary(fit)
```

11.18.6 Output & Results

A standard Cox regression on the post-landmark cohort: hazard ratio for the covariate (here `response`) measured at the landmark, with 95 % confidence interval and Wald test. The number of events and at-risk subjects shrinks after the landmark, so confidence intervals are wider than in a full-cohort analysis.

11.18.7 Interpretation

A reporting sentence: "Among the 152 patients alive at the six-month landmark, treatment responders (n = 78) had a 50 % lower hazard of subsequent death than non-responders (HR 0.50, 95 % CI 0.30 to 0.83); the analysis avoids the immortal-time bias inherent in unrestricted responder-vs-non-responder comparisons." Always state the landmark time and the post-landmark sample size.

11.18.8 Practical Tips

- Choose the landmark time at a clinically meaningful assessment point (six-week imaging response, three-month biomarker re-measurement); arbitrary times can produce unstable estimates.
- Multiple landmarks (super-model approach) combine information across time and provide a smooth treatment-effect estimate; cluster the variance on subject id when fitting.
- Sample size drops at the landmark; power calculations should account for the post-landmark cohort, not the original.
- Compare landmark results with a full time-varying Cox model that uses the covariate as a `tt()` term; both should agree if the landmark choice is reasonable.
- Be transparent about exclusions: report how many subjects had events before L and were excluded, because this is the headline difference from a naive analysis.
- For complex time-varying patterns, consider joint longitudinal-survival models (`JM` or `joineRML`) instead of repeated landmark analyses.

11.18.9 R Packages Used

`survival` for the standard Cox fit on landmark cohorts; `dynpred` for super-model landmark analyses with smooth time effects; `JM` and `joineRML` for joint longitudinal-survival modelling that handles continuous time-varying covariates without landmarks; `mstate` for multi-state extensions.

11.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.18.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.19 Left Truncation

11.19.1 Introduction

Left truncation, also called delayed entry, occurs when subjects become observable only after surviving to a certain age or calendar time. Recruiting middle-aged adults into a cohort study means anyone who died as a child or young adult is structurally absent — they could not contribute. Treating delayed-entry data as if every subject had been followed from time zero overestimates survival, because the early period is observed only for the survivors. The counting-process formulation in `survival::Surv(tstart,tstop,event)` handles this naturally by restricting risk sets at each event time to subjects actually under observation then.

11.19.2 Prerequisites

A working understanding of survival analysis, the difference between censoring (subject leaves before event) and truncation (subject only enters under specific conditions), and the counting-process formulation of survival data.

11.19.3 Theory

For a left-truncated subject who enters at time τ_i and exits at time T_i (with event indicator δ_i), the contribution to the partial likelihood at event time t should appear only when $\tau_i < t \leq T_i$. The standard Surv(time, event) object does not encode the entry time τ_i and silently sets it to zero, biasing estimates upward. The two-argument form Surv(τ, T, event) (counting-process style) carries both endpoints; `survfit()` and `coxph()` use them to construct correct risk sets.

In age-as-timescale analyses (the natural framing in chronic-disease epidemiology), every subject is left-truncated at their age at enrolment.

11.19.4 Assumptions

The truncation mechanism is independent of the event of interest given covariates: subjects do not become observable for reasons related to their underlying risk profile.

11.19.5 R Implementation

```r
library(survival)

# Age-as-timescale cohort: enrolled between ages 40 and 70
set.seed(2026)
n <- 200
age_enroll <- runif(n, 40, 70)
event_age  <- age_enroll + rexp(n, 0.05)
end_age    <- pmin(event_age, age_enroll + runif(n, 5, 20))
status     <- as.integer(event_age <= end_age)

# Left-truncated Cox: tstart = entry age, tstop = exit age
fit <- coxph(Surv(age_enroll, end_age, status) ~ 1)
summary(fit)

# Kaplan-Meier with delayed entry
km <- survfit(Surv(age_enroll, end_age, status) ~ 1)
plot(km)
```

11.19.6 Output & Results

`survfit()` returns a Kaplan-Meier curve on the chosen time scale (age, calendar year, or time-since-event) that conditions on each subject's entry time. The Cox model with the counting-process `Surv` object produces unbiased hazard-ratio estimates even when entry times are heavily truncated.

11.19.7 Interpretation

A reporting sentence: "Using age as the time scale with left truncation at enrolment, the conditional survival from age 40 was 0.75 at age 60 (95 % CI 0.68 to 0.82); ignoring delayed entry inflates this estimate to 0.84 by treating mid-life enrolees as if they had been followed since age 0." Always state the time scale and the truncation mechanism in the methods section.

11.19.8 Practical Tips

- Always specify both `tstart` and `tstop` arguments to `Surv()` when entry times vary across subjects; the single-argument form silently assumes all subjects entered at zero.
- Age-as-timescale cohort studies are always left-truncated; this is the most common scenario in chronic-disease epidemiology and ageing research.
- Distinguish left truncation (subject only observable after a certain time) from left censoring (event occurred before observation began but the subject is observable now); the two are handled differently.
- Without left-truncation correction, you overestimate survival of cohorts enrolled at older ages because their early-life mortality is invisible to your data.
- `survfit()` with delayed entry produces correct conditional survival curves; check that the reported time scale starts at the minimum entry time, not zero.
- For competing risks with delayed entry, `cmprsk::cuminc` and `tidycmprsk::cuminc` accept counting-process input via `entry` arguments.

11.19.9 R Packages Used

`survival` for `Surv(tstart,tstop,event)` and `survfit()` / `coxph()` with counting-process input; `Epi` for left-truncation-aware tools targeted at epidemiology; `etm` for non-parametric multi-state estimation with delayed entry; `tidycmprsk` for left-truncation-aware competing-risks analysis.

11.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.19.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias

- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.20 Log-Rank Test: Details

11.20.1 Introduction

The log-rank test is the standard non-parametric test for comparing survival across two or more groups in randomised trials and observational studies. It accumulates differences between observed and expected event counts at each event time across the follow-up window and reports a chi-squared statistic. The test is most powerful under proportional hazards — the alternative for which it was designed — but variants (weighted log-rank, stratified log-rank) extend its applicability to non-proportional or covariate-adjusted settings.

11.20.2 Prerequisites

A working understanding of Kaplan-Meier estimation, censoring, and the proportional-hazards alternative as the natural target for the log-rank test.

11.20.3 Theory

At each unique event time t_j across all groups, the test compares the observed number of events in group k to the expected number under the null of equal hazards: $E_{kj} = d_j \cdot n_{kj}/n_j$, with d_j the total events at t_j and n_{kj}, n_j the numbers at risk. The standardised cumulative deviation across all event times forms a chi-squared statistic with $k-1$ degrees of freedom, where k is the number of groups.

Weighted variants (Wilcoxon, Tarone-Ware, Fleming-Harrington $G^{\rho,\gamma}$) give more weight to early or late differences and recover power when hazards are non-proportional. The stratified log-rank applies the test within strata defined by a categorical covariate and pools the contributions, controlling for that covariate while testing the main effect.

11.20.4 Assumptions

Proportional hazards across groups (for the standard log-rank), independent and non-informative censoring, and accurate event-time recording.

11.20.5 R Implementation

```
library(survival)

data(lung)
```

```r
fit <- survdiff(Surv(time, status) ~ sex, data = lung)
fit

# Stratified log-rank
survdiff(Surv(time, status) ~ sex + strata(ph.ecog), data = lung)
```

11.20.6 Output & Results

`survdiff()` returns observed and expected events per group, the chi-squared test statistic, degrees of freedom, and the asymptotic p-value. The stratified version partitions the comparison within levels of the strata variable, producing the same output structure but with the stratification controlled for.

11.20.7 Interpretation

A reporting sentence: "The log-rank test rejected the null of equal survival between sexes ($\chi^2_1 = 10.3$, $p = 0.001$); the female arm had 53 observed events compared with 86 expected under the null, while the male arm had 112 observed compared with 79 expected." Always pair the test with a hazard-ratio estimate (from Cox) and Kaplan-Meier curves; the test alone gives no effect size.

11.20.8 Practical Tips

- Use stratification (`+strata(var)`) when controlling for a covariate without testing it; stratified log-rank tests within strata and pools.
- When hazards cross, the standard log-rank can fail to detect a real difference; use the Fleming-Harrington $G^{1,0}$ test for late differences or $G^{0,1}$ for early ones.
- For many groups, log-rank gives an omnibus test; for pairwise comparisons follow up with `pairwise_survdiff()` from `survminer` and apply Bonferroni or Hochberg correction.
- Report Kaplan-Meier curves alongside the test result; a p-value without a curve is uninterpretable for clinical readers.
- Non-informative censoring is the assumption that fails most often in observational studies; if censoring depends on prognosis, the log-rank test is biased.
- For severely non-proportional hazards (clearly crossing curves), consider RMST as the primary summary instead of relying on the log-rank.

11.20.9 R Packages Used

`survival` for `survdiff()` with stratification; `survminer::pairwise_survdiff()` for pairwise log-rank tests with multiple-comparison correction; `nph` and `coin` for weighted-log-rank variants and exact tests; `FHtest` for Fleming-Harrington $G^{\rho,\gamma}$ tests.

11.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.20.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.21 Multi-State Models

11.21.1 Introduction

A multi-state model describes a subject's movement through a discrete set of states over time — healthy to diseased to dead, pre-transplant to post-transplant to rejected, or any clinical pathway with intermediate landmarks. It generalises ordinary survival (two states: alive, dead), competing risks (one origin, several absorbing states), and the illness-death model (one intermediate, one absorbing state). The framework gives a complete probabilistic picture of the disease course rather than collapsing it into a single time-to-first-event summary.

11.21.2 Prerequisites

A working understanding of Cox regression, competing-risks analysis, and the concept of transition intensities (state-specific hazards) as the multi-state generalisation of a single survival hazard.

11.21.3 Theory

A multi-state model represents states as nodes and possible transitions as directed edges. Each transition $j \to k$ has an associated transition intensity (hazard) $\alpha_{jk}(t)$. Transition-specific Cox models estimate covariate effects on each α_{jk} separately, with stratification by transition number ensuring each baseline can differ. Predicted transition probabilities $P_{jk}(s,t)$ — the probability of being in state k at time t given state j at time s — are obtained from the cumulative-hazard matrix via the product-integral, implemented in `mstate::probtrans()`.

The Markov property assumes the future depends only on the current state and time, not on the path taken to reach it; semi-Markov extensions condition on

time-since-entry-to-current-state instead.

11.21.4 Assumptions

Markov property (often, with semi-Markov extensions available), proportional hazards within each transition (or relaxed via spline-based or stratified models), and accurate state classification at each time point.

11.21.5 R Implementation

```r
library(mstate)

# Example: illness-death model (3 states)
# Load example data
data(ebmt3)

tmat <- trans.illdeath()     # transition matrix
d_ms <- msprep(data = ebmt3, trans = tmat,
               time = c(NA, "prtime", "rfstime"),
               status = c(NA, "prstat", "rfsstat"))

# Fit Cox models per transition
fit <- coxph(Surv(Tstart, Tstop, status) ~ dissub + age + strata(trans),
             data = d_ms)
summary(fit)
```

11.21.6 Output & Results

`msprep()` reshapes wide-format clinical data into the transition-stratified long format that multi-state Cox models expect. The Cox fit produces transition-specific coefficients (when interactions with `strata(trans)` are included). `mstate::msfit()` and `mstate::probtrans()` deliver predicted cumulative hazards per transition and transition probabilities $P_{jk}(s,t)$ at any horizon.

11.21.7 Interpretation

A reporting sentence: "Increasing age was associated with elevated transition hazards from healthy to relapsed (HR 1.02 per year, 95 % CI 1.01 to 1.04) and from healthy to death (HR 1.04, 95 % CI 1.02 to 1.06); the predicted probability of being in the relapsed state at five years was 28 % for a 30-year-old patient and 41 % for a 60-year-old." Predictions in absolute probabilities communicate multi-state results far better than the underlying transition-specific hazard ratios alone.

11.21.8 Practical Tips

- Draw the state diagram before any analysis; a clear picture of allowed transitions prevents misspecified transition matrices.
- `msprep()` converts wide patient-level data into the long format with one row per allowed transition per subject; this is the most common entry barrier for new users.
- Use `probtrans()` predictions to communicate results in absolute terms — readers absorb "20 % probability of death by year 5" much better than three transition hazard ratios.
- The Markov vs semi-Markov choice matters for sojourn-time-dependent hazards; `mstate` supports both via the `Tstart` argument and time-origin handling.
- For high-dimensional state spaces, sparse-state extensions in `msm` (continuous-time Markov chains for panel data) are often more tractable than full-likelihood multi-state models.
- Validate predictions on held-out data; multi-state Cox models can overfit transition-specific hazards when individual transitions have few events.

11.21.9 R Packages Used

`mstate` for the `msprep`/`msfit`/`probtrans` workflow with multi-state Cox models; `msm` for continuous-time Markov chain models on panel-observed states; `flexsurv::msfit.flexsurvreg` for parametric multi-state extensions; `etm` for non-parametric Aalen-Johansen-style transition probability estimation.

11.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.21.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.22 The Nelson-Aalen Estimator

11.22.1 Introduction

The Nelson-Aalen estimator is the standard non-parametric estimator of the cumulative hazard $H(t)$ for right-censored data. It is the natural counterpart of the Kaplan-Meier estimator: where KM directly targets the survival function, Nelson-Aalen targets the cumulative hazard, and the two are linked by $S(t) = \exp(-H(t))$. Nelson-Aalen has slightly better small-sample properties than Kaplan-Meier in the right tail and is the natural input for any analysis framed in terms of hazards rather than survival probabilities, including diagnostics for proportional hazards across groups.

11.22.2 Prerequisites

A working understanding of cumulative hazards, the Kaplan-Meier estimator, and the relationship $S(t) = \exp(-H(t))$.

11.22.3 Theory

The Nelson-Aalen estimator is

$$\hat{H}(t) = \sum_{t_i \le t} \frac{d_i}{n_i},$$

with d_i the number of events at t_i and n_i the number at risk just before t_i. Each event time contributes its empirical hazard increment d_i/n_i, and the cumulative hazard is the running sum. The variance estimator is

$$\widehat{\mathrm{Var}}(\hat{H}(t)) = \sum_{t_i \le t} d_i/n_i^2,$$

which gives pointwise confidence bands on the log-log scale. The Nelson-Aalen-based survival estimator $\hat{S}(t) = \exp(-\hat{H}(t))$ differs slightly from the Kaplan-Meier product-limit estimator and is preferred when the cumulative-hazard scale is the primary reporting target.

11.22.4 Assumptions

Independent, non-informative censoring. Adequate at-risk sets across the follow-up window — the right tail of the estimator has wide variance when only a handful of subjects remain.

11.22.5 R Implementation

```r
library(survival)

data(lung)
fit <- survfit(Surv(time, status) ~ 1, data = lung)

# Cumulative hazard via NA
plot(fit, fun = "cumhaz", main = "Nelson-Aalen cumulative hazard")

# Underlying values
summary(fit)$cumhaz |> head()
```

11.22.6 Output & Results

`survfit()` with `fun="cumhaz"` plots the Nelson-Aalen cumulative hazard with pointwise confidence bands. `summary(fit)$cumhaz` returns the underlying numeric values; the corresponding event times come from `summary(fit)$time`.

11.22.7 Interpretation

A reporting sentence: "The Nelson-Aalen cumulative hazard reached approximately 1.5 by day 800 (95 % CI 1.3 to 1.7), corresponding to a survival probability of $\exp(-1.5) = 0.22$ at that time; the slope of $\widehat{H}(t)$ in the first 200 days indicates an instantaneous hazard of approximately 0.0006 per day." Always pair the cumulative-hazard estimate with the implied survival probability for non-statistical readers.

11.22.8 Practical Tips

- Use Nelson-Aalen for cumulative-hazard reporting and diagnostic comparisons; use Kaplan-Meier for survival probabilities.
- Compare $\log \widehat{H}(t)$ across groups: parallel lines indicate proportional hazards (the Cox-model assumption); divergent or crossing lines indicate non-proportionality.
- Construct confidence bands on the log-log scale or via $H \pm z_{\alpha/2} \cdot \mathrm{SE}(H)$; the log-log version respects the non-negativity of cumulative hazard.
- For Cox regression baseline hazard estimation, Breslow's estimator is the slight variant used internally; both reduce to Nelson-Aalen when no covariates are present.
- For very small at-risk sets at late times, the estimator's variance balloons; interpret the right tail cautiously and consider truncating the plot at the last well-sampled time.
- For competing risks, the cumulative-cause-specific hazard is the obvious analogue; the Aalen-Johansen estimator generalises Nelson-Aalen to multi-state processes.

11.22.9 R Packages Used

`survival` for `survfit()` with `fun="cumhaz"`; `mstate` for multi-state extensions including the Aalen-Johansen estimator; `etm` for the empirical transition matrix; `ggsurvfit` for ggplot-flavoured Nelson-Aalen plots integrated with downstream tidyverse work.

11.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.22.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.23 Parametric Survival: Log-Normal

11.23.1 Introduction

The log-normal survival model assumes that $\log T$ follows a Normal distribution. Its distinctive feature among parametric survival models is a *non-monotone* hazard — the instantaneous risk rises early in follow-up, peaks, and then declines. This shape suits diseases or processes in which susceptible individuals fail relatively quickly while a hardy subset survives indefinitely: for example, post-surgical mortality (early peri-operative risk that wanes), some cancer recurrences, and time-to-engagement processes in behavioural data. Where Weibull and exponential models lock in monotone hazards, log-normal accommodates the rise-and-fall pattern naturally.

11.23.2 Prerequisites

A working understanding of the log-normal distribution, parametric survival regression, and the accelerated-failure-time (AFT) parameterisation as an alternative to the proportional-hazards form.

11.23.3 Theory

The log-normal survival function is

$$S(t) = 1 - \Phi\left(\frac{\log t - \mu}{\sigma}\right),$$

with hazard

$$h(t) = \frac{(1/(t\sigma))\phi((\log t - \mu)/\sigma)}{1 - \Phi((\log t - \mu)/\sigma)}.$$

The hazard has a unique maximum near $t \approx \exp(\mu - \sigma^2)$ for $\sigma > 1$ and decreases thereafter. The AFT parameterisation $\log T = X\beta + \sigma W$ (with $W \sim \mathcal{N}(0,1)$) makes regression coefficients interpretable as multiplicative effects on the *event time* — a one-unit increase in a covariate scales the time scale by $\exp(-\beta)$.

11.23.4 Assumptions

A non-monotone (rising-then-falling) hazard, log-Normal residual structure on the log time scale, independent observations, and non-informative censoring.

11.23.5 R Implementation

```
library(survival); library(flexsurv)

data(lung)
fit <- survreg(Surv(time, status) ~ age + sex, data = lung, dist =
 ↪  "lognormal")
summary(fit)

# flexsurv
flex <- flexsurvreg(Surv(time, status) ~ age + sex, data = lung, dist =
 ↪  "lnorm")
plot(flex, type = "hazard")
```

11.23.6 Output & Results

`survreg()` returns AFT coefficients on the log-time scale plus the scale parameter σ. `flexsurvreg()` produces equivalent estimates with cleaner survival, hazard, and quantile prediction methods. `plot(flex,type="hazard")` draws the model-implied hazard curve, making the rise-peak-decline shape visible against KM-based hazard estimates.

11.23.7 Interpretation

A reporting sentence: "A log-normal AFT model estimated that each additional year of age shortened expected event time by approximately 2 % ($\hat{\beta}_{\text{age}} = -0.02$, 95 % CI −0.04 to −0.00); the model-implied hazard peaked at approximately 9

months and declined thereafter, consistent with the observed concentration of events in the first year of follow-up." Always plot the implied hazard alongside Kaplan-Meier to verify the non-monotone shape is supported by the data.

11.23.8 Practical Tips

- Use log-normal when Kaplan-Meier or non-parametric hazard estimates show a clear "hump" of events early in follow-up; for monotone hazards, Weibull or exponential are simpler and more efficient.
- Log-logistic is a close alternative with similar non-monotone hazard but heavier tails; compare AICs across both before committing.
- AIC comparison across parametric families (exponential, Weibull, log-normal, log-logistic, generalised gamma) is the standard model-selection workflow; the generalised gamma nests several of these.
- For very flexible hazards beyond the log-normal shape, Royston-Parmar spline-based models offer more freedom while remaining parametric.
- Predictions on the natural time scale come via `predict(fit,type="quantile")` for `survreg` or `summary(flex,type="quantile")` for `flexsurv` objects.
- When some covariates produce monotone hazards and others produce non-monotone, consider a heterogeneous-effects extension or a flexible spline model rather than forcing one parametric family.

11.23.9 R Packages Used

`survival` for `survreg()` with `dist="lognormal"`; `flexsurv` for `flexsurvreg()` with cleaner prediction interfaces and rich plotting; `eha` for additional parametric hazard models; `rstpm2` for Royston-Parmar spline-based parametric extensions when the log-normal hazard shape is too restrictive.

11.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.23.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.24 Parametric Survival: Weibull

11.24.1 Introduction

The Weibull distribution is the most flexible parametric survival family that has a monotone hazard. Hazard is increasing when the shape parameter $k > 1$, constant (exponential) when $k = 1$, and decreasing when $k < 1$. Among the common parametric families it is unique in admitting both proportional-hazards and accelerated-failure-time interpretations, which makes it the natural starting point for parametric survival analysis whenever PH is plausible. Cleaner extrapolation, smaller standard errors, and explicit hazard shape are the practical pay-offs over the semi-parametric Cox model when its assumptions hold.

11.24.2 Prerequisites

A working understanding of the Weibull distribution, survival analysis, and the distinction between proportional-hazards and accelerated-failure-time parameterisations.

11.24.3 Theory

The Weibull survival function is

$$S(t) = \exp(-(t/\lambda)^k)\,,$$

with hazard $h(t) = (k/\lambda)(t/\lambda)^{k-1}$. The hazard is monotone: increasing for $k > 1$ (positive ageing), decreasing for $k < 1$ (early-failure regimes), and constant for $k = 1$ (exponential).

Two parameterisations coexist:

- **Accelerated-failure-time (AFT)** in `survreg(dist="weibull")`: $\log T = X\beta + \sigma W$ with W a standard extreme-value variate; the scale is $\sigma = 1/k$.
- **Shape-scale** in `flexsurv::flexsurvreg(dist="weibull")`: explicit shape k and scale λ, with PH-style covariate effects via interactions on λ.

The conversion between PH and AFT forms is $\beta_{\text{PH}} = -k \cdot \beta_{\text{AFT}}$.

11.24.4 Assumptions

A monotone hazard. The standard diagnostic is the log–log survival plot: $\log[-\log \widehat{S}(t)]$ against $\log t$ should be approximately linear with slope k if the Weibull holds.

11.24.5 R Implementation

```r
library(survival); library(flexsurv)

data(lung)

fit_survreg <- survreg(Surv(time, status) ~ age + sex, data = lung,
                       dist = "weibull")
summary(fit_survreg)

# flexsurvreg gives shape and scale directly
fit_flex <- flexsurvreg(Surv(time, status) ~ age + sex, data = lung,
                        dist = "weibull")
fit_flex
```

11.24.6 Output & Results

`survreg()` returns AFT coefficients (interpret as multiplicative effects on event time, $\exp(-\beta)$) plus the scale parameter. `flexsurvreg()` returns shape, scale, and PH-style hazard ratios directly — easier to communicate to a Cox-fluent audience. Both yield the same likelihood and the same fitted survival.

11.24.7 Interpretation

A reporting sentence: "A Weibull AFT model estimated shape parameter $\hat{k} = 1.31$ (95 % CI 1.10 to 1.55), indicating an increasing hazard over follow-up; each additional year of age accelerated the event by approximately 2 %, equivalent to a hazard ratio of 1.025 per year on the PH scale." When reporting, choose AFT or PH consistently with your audience: clinical readers expect HRs, while time-to-event-cost analyses prefer time-acceleration ratios.

11.24.8 Practical Tips

- Use the log–log survival plot (`plot(survfit(...),fun="cloglog")`) as the main diagnostic for Weibull suitability; clear non-linearity rules it out.
- For more flexible monotone hazards, compare AIC against generalised gamma and Gompertz; for non-monotone, switch to log-normal or Royston-Parmar splines.
- Convert between PH and AFT carefully: $\beta_{\text{PH}} = -k \cdot \beta_{\text{AFT}}$, and remember that AFT confidence intervals are on the log-time scale.
- For small samples, Weibull yields smoother survival estimates than Kaplan-Meier and lower-variance covariate effects than Cox; this is its main practical advantage over Cox.
- `flexsurv` simplifies prediction (`summary(fit,type="survival",t=...)`) and supports time-dependent shape parameters via spline extensions.
- Check Weibull plus Cox HR estimates; they should be close under PH.

Material disagreement points to an assumption failure that should be investigated.

11.24.9 R Packages Used

`survival` for `survreg()` with `dist="weibull"`; `flexsurv` for `flexsurvreg()` with shape/scale output and richer prediction; `eha` for additional parametric extensions; `rstpm2` for Royston-Parmar spline alternatives when monotone hazards are too restrictive.

11.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.24.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.25 Recurrent Events

11.25.1 Introduction

Recurrent-event analysis applies whenever a single subject can experience the event of interest multiple times during follow-up — hospital admissions, seizures, infections, asthma exacerbations, recurrent tumours. Treating only the first event collapses information and wastes power; treating all events as independent inflates effective sample size and produces invalid standard errors. Several modelling frameworks balance these extremes, each suited to a different scientific question.

11.25.2 Prerequisites

A working understanding of Cox regression, counting-process data formats, and within-subject correlation as a source of dependent observations.

11.25.3 Theory

The main approaches:

- **Andersen-Gill (AG)**: a counting-process Cox model that treats all events equally and uses cluster-robust standard errors to account for within-subject dependence. Estimates a single baseline hazard for the recurrent process.
- **Prentice-Williams-Peterson (PWP)**: stratifies by event number, allowing separate baseline hazards for the first, second, third, etc. event. Two flavours: total time (gap from study start) and gap time (gap from previous event). Use when the hazard structure changes with event order.
- **Wei-Lin-Weissfeld (WLW)**: fits a separate marginal Cox model for each event order, using only the first event for $K = 1$, the first two for $K = 2$, etc. Conceptually cleaner but increasingly out of favour because it conditions on having survived enough events to be at risk.
- **Shared-frailty**: a Cox model with a subject-level random effect u_i that captures unmeasured individual heterogeneity. Combine with AG when frailty is the natural interpretation of within-subject correlation.

11.25.4 Assumptions

The chosen correlation structure matches the data, the hazard specification (PH within strata or marginally) is appropriate, and counting-process intervals are correctly constructed (no overlaps, no gaps).

11.25.5 R Implementation

```r
library(survival)

# Counting-process data: id, tstart, tstop, status (1 = event)
data(cgd, package = "survival")

# Andersen-Gill
ag <- coxph(Surv(tstart, tstop, status) ~ treat + inherit,
            data = cgd, cluster = id)
summary(ag)

# PWP stratified by event number
cgd$event.num <- ave(cgd$status, cgd$id, FUN = cumsum) + 1
pwp <- coxph(Surv(tstart, tstop, status) ~ treat + inherit +
 ↪  strata(event.num),
             data = cgd, cluster = id)
summary(pwp)
```

11.25.6 Output & Results

Both fits return hazard-ratio estimates with cluster-robust standard errors. AG produces a single HR per covariate covering the full recurrent-event process;

PWP produces strata-specific baselines while sharing covariate effects across strata, allowing different baseline hazards per event order.

11.25.7 Interpretation

A reporting sentence: "An Andersen-Gill counting-process Cox model estimated that prophylactic treatment reduced the recurrent-infection hazard by 60 % (HR 0.40, 95 % CI 0.25 to 0.63), with cluster-robust SEs accounting for within-subject correlation; the PWP model stratified by event order produced a consistent estimate (HR 0.43, 95 % CI 0.26 to 0.70), suggesting the treatment effect does not vary materially across event order." Compare AG and PWP whenever the data structure permits.

11.25.8 Practical Tips

- AG is the default starting point for most applications and the easiest to interpret.
- PWP (gap-time) is appropriate when the hazard depends on time since the previous event; PWP (total time) when it depends on time since study start.
- WLW has fallen out of favour because it conditions on event order in a way that produces awkward causal interpretations.
- For unmeasured individual heterogeneity not captured by covariates, add a shared frailty term to the AG model (`+frailty(id)`).
- Always use cluster-robust SEs (`cluster=id`); ignoring within-subject correlation produces SEs that are too small and Type I error inflated.
- Pair hazard ratios with absolute counts of events, expected rates, and cumulative mean functions; recurrent-event audiences typically reason in absolute frequencies rather than hazard ratios.

11.25.9 R Packages Used

`survival` for `coxph()` with `cluster()` and `strata()`; `frailtypack` for joint frailty models with terminal events and recurrent events fitted simultaneously; `reReg` for additional recurrent-event methods including marginal mean models; `mets` for clustered survival data with flexible dependence structures.

11.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.25.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.26 Restricted Mean Survival Time (RMST)

11.26.1 Introduction

The restricted mean survival time (RMST) is the expected survival time truncated at a clinically meaningful horizon τ. Where the hazard ratio from a Cox model summarises a relative effect that requires proportional hazards to be interpretable, RMST quantifies an absolute time difference (in days, months, or years) and remains valid even when the hazard ratio is not constant. It has become an increasingly standard endpoint in oncology trials and immunotherapy studies, where crossing or delayed survival curves are common and where regulators and clinicians want a number expressed on the natural time scale.

11.26.2 Prerequisites

A working understanding of survival functions, the Kaplan-Meier estimator, and the issues with proportional-hazards assumptions in modern clinical trials.

11.26.3 Theory

For a survival function $S(t)$ and a horizon τ, the RMST is

$$\mathrm{RMST}(\tau) = \mathbb{E}[\min(T, \tau)] = \int_0^\tau S(t)\,\mathrm{d}t.$$

Geometrically it is the area under the KM curve from time zero to τ. Group differences are summarised by the difference $\mathrm{RMST}_1(\tau) - \mathrm{RMST}_0(\tau)$ or by the ratio. Standard errors come from the asymptotic theory developed by Royston and Parmar; bootstrap confidence intervals are an alternative when the asymptotic approximation is suspect. Crucially, the construction makes no assumption about the relationship between hazards in the two groups — non-proportionality, treatment-effect waning, and crossing curves are all handled gracefully.

11.26.4 Assumptions

Sufficient follow-up to estimate $S(t)$ reliably up to τ; an explicit choice of τ that is clinically meaningful and within the support of the data; non-informative censoring up to τ.

11.26.5 R Implementation

```r
library(survRM2)

data(lung)

# Truncation at 600 days
rmst2(time = lung$time, status = lung$status,
      arm = lung$sex - 1, tau = 600)
```

11.26.6 Output & Results

`rmst2()` returns the RMST for each arm, the difference and ratio between arms, and 95 % confidence intervals derived from the asymptotic theory. The output is structured for direct inclusion in clinical-trial reports.

11.26.7 Interpretation

A reporting sentence: "Through 600 days of follow-up, female patients accumulated an average of 380 days alive vs 310 days for male patients; the RMST difference was 70 days (95 % CI 20 to 120, $p = 0.005$). Because the hazards were not proportional (Schoenfeld test $p = 0.04$), RMST is the preferred summary." Always state τ and justify it on clinical grounds; the same data can yield different RMST conclusions at different horizons.

11.26.8 Practical Tips

- Choose τ based on clinical relevance *and* on data adequacy: τ should be smaller than the longest follow-up time across both arms so the KM estimate is stable.
- RMST handles non-proportional hazards, treatment-effect waning, and crossing curves where Cox HR fails; for any trial with these features, RMST is preferable.
- Report RMST on the absolute time scale (days, months, years); clinicians and patients reason about absolute survival, not log hazard ratios.
- For multi-arm trials, run pairwise RMST comparisons with Bonferroni or Hochberg correction; `survRM2::rmst2()` handles two-arm comparisons natively.
- Pair RMST with KM curves and a hazard-ratio sensitivity analysis; reviewers expect to see all three when proportional-hazards assumptions are in question.
- For very small samples or sparse late-time data, switch to bootstrap confidence intervals via `bootRMST` or hand-coded `boot::boot()` resampling on the patient-level data.

11.26.9 R Packages Used

`survRM2` for `rmst2()` and the canonical RMST workflow; `survival` for the underlying Kaplan-Meier estimation; `pseudo` for pseudo-observation-based RMST regression with covariates; `flexsurv` for parametric models that integrate analytically over $S(t)$; `survminer::ggsurvplot()` for KM curves that complement the RMST report.

11.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.26.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.27 Schoenfeld Residuals

11.27.1 Introduction

Schoenfeld residuals are the standard diagnostic for the proportional-hazards (PH) assumption in Cox regression. They are computed at each event time as the difference between the failing subject's covariate value and the expected covariate value across the risk set. Under PH, Schoenfeld residuals are uncorrelated with time; a trend across time signals that the hazard ratio is itself time-varying, which violates the central assumption of the Cox model and invalidates a single-number hazard-ratio interpretation.

11.27.2 Prerequisites

A working understanding of Cox proportional-hazards regression, the partial likelihood, and the difference between the proportional-hazards and accelerated-failure-time framings of survival regression.

11.27.3 Theory

At event time t_i in a Cox model, the Schoenfeld residual for covariate x is

$$r_i = x_i - \hat{\mathbb{E}}[x \mid \text{risk set at } t_i],$$

with the expectation weighted by the partial-likelihood weights $\hat{w}_j = \exp(X_j\hat{\beta}) / \sum_k \exp(X_k\hat{\beta})$ over the risk set. Grambsch and Therneau (1994) developed scaled versions whose correlation with a function of time $g(t)$ — typically $\log t$ or rank-time — yields a χ^2_1 test of PH for each covariate, plus a global χ^2_p test for the joint null. The function `cox.zph()` implements this test and returns smoothed-residual plots that visualise the time-varying effect when one is present.

11.27.4 Assumptions

A Cox model has been fitted, with sufficient events to estimate the smoothed Schoenfeld residual function.

11.27.5 R Implementation

```r
library(survival)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)

zph <- cox.zph(fit)
print(zph)

# Visualise
par(mfrow = c(1, 3))
plot(zph)
par(mfrow = c(1, 1))
```

11.27.6 Output & Results

`print(zph)` returns per-covariate χ^2 statistics, degrees of freedom, and p-values, plus a global test against the joint null that all hazard ratios are constant. `plot(zph)` produces one diagnostic panel per covariate showing the smoothed Schoenfeld residual against transformed time, with a horizontal reference line at the constant-hazard-ratio assumption.

11.27.7 Interpretation

A reporting sentence: "Schoenfeld residual plots for age, sex, and ECOG performance status showed no systematic trend (per-covariate p all > 0.20, global $p = 0.55$); the proportional-hazards assumption is supported." When PH fails:

"The hazard ratio for treatment varied with time (Schoenfeld $p < 0.001$, decreasing trend); we report restricted-mean survival difference rather than a single Cox HR."

11.27.8 Practical Tips

- Always run `cox.zph()` after fitting a Cox model; reporting an HR without a PH check is no longer acceptable in clinical-trial papers.
- When PH fails for one covariate, options include: stratifying by it (`+strata(var)`), modelling its effect as time-varying with `tt()` or `survSplit()`, or switching to RMST as the summary of treatment effect.
- Categorical covariates with k levels produce $k - 1$ Schoenfeld tests; PH may hold for some contrasts and fail for others.
- Small samples have low power to detect PH violations; pair the formal test with a visual inspection of `plot(zph)` rather than relying on p-values alone.
- For strictly time-varying effects, switch to flexible parametric models (`flexsurv::flexsurvreg` with splines on log time) or include `tt(...)` time-transform functions in `coxph()`.
- A failed PH test in itself is not reason to abandon the Cox model; it is a reason to refine the model, change the summary, or stratify.

11.27.9 R Packages Used

`survival` for `coxph()` and `cox.zph()`; `survminer::ggcoxzph()` for ggplot-style Schoenfeld diagnostic panels; `timereg` for additive hazards and structured time-varying-coefficient models; `flexsurv` for parametric alternatives when PH fails.

11.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.27.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.28 Stratified Cox Models

11.28.1 Introduction

A stratified Cox model is the standard fix when the proportional-hazards assumption fails across the levels of a categorical covariate but holds within them. Stratification estimates a separate baseline hazard $h_{0s}(t)$ for each stratum while constraining the covariate effects β to be common across strata. The cost is that you cannot estimate the effect of the stratifier itself — its baselines are absorbed — but the gain is honest hazard-ratio estimation for the rest of the covariates without forcing PH across the problematic dimension.

11.28.2 Prerequisites

A working understanding of Cox proportional-hazards regression, the diagnostic role of Schoenfeld residuals, and the concept of stratification as an alternative to direct adjustment.

11.28.3 Theory

The stratified Cox model is

$$h(t \mid x, \text{stratum} = s) = h_{0s}(t) \exp(x^\top \beta).$$

The partial likelihood is summed across strata: each stratum contributes the standard Cox partial-likelihood term over its own risk sets, and the resulting score equations determine $\hat{\beta}$. Because the stratum baselines are unrestricted, no assumption is made about how $h_{0s}(t)$ relates across s. The trade-off is that the effect of being in stratum s versus another stratum cannot be estimated within the model — that contrast is precisely the term absorbed into the separate baselines.

11.28.4 Assumptions

Proportional hazards within each stratum (so the common β is meaningful), independent observations, and non-informative censoring.

11.28.5 R Implementation

```r
library(survival)

data(lung)
# If ph.ecog violates PH, stratify on it
fit <- coxph(Surv(time, status) ~ age + sex + strata(ph.ecog), data =
 ↪  lung)
summary(fit)
```

```
cox.zph(fit)
```

11.28.6 Output & Results

`summary(fit)` lists hazard ratios for non-stratified covariates with confidence intervals and Wald tests; the stratifier appears nowhere because its effect is absorbed into the per-stratum baselines. `cox.zph()` checks PH for the remaining covariates within strata.

11.28.7 Interpretation

A reporting sentence: "After Schoenfeld residuals indicated a PH violation for ECOG performance status (per-covariate $p < 0.01$), we fit a Cox model stratified on ECOG; the resulting hazard ratios for age (HR 1.02 per year, 95 % CI 1.01 to 1.03) and sex (HR 0.60 for female vs male, 95 % CI 0.42 to 0.85) are pooled across strata." When the stratifier is itself of clinical interest, supplement the model with stratum-specific Kaplan-Meier curves so readers can see the absorbed effect.

11.28.8 Practical Tips

- Stratify on a covariate when its PH fails and you do not need its hazard-ratio estimate; if you do need it, switch to a time-varying coefficient via `tt()` or `survSplit()` instead.
- Many small strata reduce the effective sample size for $\hat{\beta}$; pool levels (e.g. ECOG 0 vs 1+) when raw stratification leaves few events per stratum.
- Interactions between stratum and other covariates allow different β per stratum, generalising the model toward fully separate fits per stratum at the cost of degrees of freedom.
- Predictions and survival curves are stratum-specific; report which baseline is used for any predicted survival probability.
- Always verify that PH still holds for the unstratified covariates after stratification; failure there motivates additional stratification or a switch to time-varying effects.
- For continuous covariates that violate PH, stratification requires categorisation (typically into quartiles), which loses information; time-varying coefficients are usually preferable.

11.28.9 R Packages Used

`survival` for `coxph()` with `strata()`; `survminer::ggcoxzph()` for PH diagnostics within strata; `flexsurv` for parametric alternatives that can accommodate non-proportional hazards directly; `timereg` for additive hazards models when stratification is too coarse.

11.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.28.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.29 Simulating Survival Data

11.29.1 Introduction

Simulating censored time-to-event data is essential for three classes of work: power and sample-size calculations for trials with non-standard designs, validation of new statistical methods (does the estimator recover the truth?), and pedagogy. The inverse-hazard method generates event times from any specified hazard function and lets you build realistic datasets that match a target survival pattern, censoring distribution, and covariate-effect structure.

11.29.2 Prerequisites

A working understanding of hazard and cumulative hazard functions, the inverse-CDF method for random-number generation, and the relationship between proportional-hazards models and exponentiated covariate effects.

11.29.3 Theory

For a hazard $h(t \mid x)$ with cumulative hazard $H(t \mid x)$ and a uniform variate $U \sim \text{Uniform}(0, 1)$,

$$T = H^{-1}(-\log U \mid x)$$

has the target distribution. Under proportional hazards $H(t \mid x) = H_0(t) \exp(x^\top \beta)$, so

$$T_i = H_0^{-1}(-\log U_i / \exp(x_i^\top \beta)) \,.$$

Censoring is added by drawing an independent censoring time C_i from a chosen distribution (administrative, exponential, or empirical from a real cohort) and recording $\min(T_i, C_i)$ as the observation time and $\mathbf{1}\{T_i \leq C_i\}$ as the event indicator.

11.29.4 Assumptions

The specified baseline hazard and covariate-effect structure are the truth being simulated; the censoring distribution is independent of T_i unless the simulation explicitly handles informative censoring.

11.29.5 R Implementation

```r
library(simsurv)

set.seed(2026)
n <- 300
x <- data.frame(id = 1:n, arm = factor(rep(c("A", "B"), each = n/2)))
x$arm_num <- as.numeric(x$arm) - 1

# Weibull baseline: lambda = 0.1, gamma = 1.2
sim <- simsurv(lambdas = 0.1, gammas = 1.2,
               x = x, betas = c(arm_num = -0.5),
               maxt = 10)

# Merge with covariates
d <- merge(sim, x, by = "id")
head(d)

# Kaplan-Meier
library(survival)
fit_km <- survfit(Surv(eventtime, status) ~ arm, data = d)
plot(fit_km)
```

11.29.6 Output & Results

`simsurv()` returns a tibble with one row per subject containing the simulated event time, the censoring indicator (here taken at `maxt=10`), and the subject id. Merging back with the covariate frame gives a complete simulated dataset ready for analysis. The Kaplan-Meier plot shows the two arms separating with the designed effect size.

11.29.7 Interpretation

A reporting sentence: "Simulated Weibull data with shape 1.2, baseline scale 0.1, and a treatment hazard ratio of $\exp(-0.5) = 0.61$; a Cox model on the simulated

data recovered an HR of 0.59 (95 % CI 0.46 to 0.76), confirming the simulation is correctly calibrated." Always confirm that the analysis pipeline recovers the simulation parameters before using the simulator for power calculations.

11.29.8 Practical Tips

- `simsurv` supports Weibull and Gompertz baselines plus user-defined hazard functions; for arbitrary parametric shapes, supply `hazard` or `loghazard` directly.
- For competing risks, simulate each cause's event time independently and record the cause that occurred first; the resulting data have the correct cumulative incidence functions.
- Match the censoring distribution to the real-study target: administrative censoring at study end, plus loss-to-follow-up exponentials with rates calibrated to historical drop-out.
- For power calculations, run thousands of simulated datasets, fit the planned model on each, and record the proportion of times the test rejects at the chosen alpha; this is the standard Monte Carlo power estimator.
- For time-varying covariates, simulate via piecewise-constant hazards across pre-specified intervals; `simsurv` supports `tdefunction` for time-dependent covariate effects.
- For validation of new methods, re-simulate under several scenarios (PH violation, heavy censoring, sparse events) and check robustness; a method that works only on its design assumptions is brittle.

11.29.9 R Packages Used

`simsurv` for the canonical inverse-hazard simulator with Weibull, Gompertz, and user-defined baselines; `flexsurv` for `rflexsurv()` simulation from fitted flexible models; `coxed` for Cox-model-based simulations; `survSim` for additional simulation extensions including frailty and time-dependent effects.

11.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.29.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models

- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.30 Time-Dependent ROC

11.30.1 Introduction

Time-dependent ROC extends classical ROC analysis to right-censored survival data. At each chosen evaluation time t, the analysis treats subjects observed to fail before t as cases and subjects known to survive past t as controls; subjects censored before t are reweighted via inverse-probability-of-censoring weights. The resulting time-specific AUC summarises how well a continuous risk score (such as a Cox linear predictor) discriminates failures from survivors at t. It is the natural extension of the C-index to "what is the AUC at exactly one year" reporting.

11.30.2 Prerequisites

A working understanding of ROC curves, the C-index, Cox regression linear predictors, and inverse-probability-of-censoring weighting.

11.30.3 Theory

Two definitions of cases and controls compete:

- **Incident / dynamic** (I/D): cases are subjects who fail at time t exactly; controls are those still at risk past t. Useful for hazard-style discrimination.
- **Cumulative / dynamic** (C/D): cases are subjects who fail at any time $\leq t$; controls are those still at risk past t. The more common choice for clinical risk-score reporting.

Inverse-probability-of-censoring weights compensate for subjects censored before the comparison can be made; the censoring distribution is typically estimated by Kaplan-Meier on the censoring times. The result is a time-dependent AUC at each requested t with bootstrap or analytical confidence intervals.

11.30.4 Assumptions

Non-informative censoring (or correctly conditional on covariates if `weighting="cox"` or `"aalen"` is used) and a continuous risk marker that is meaningful as a ranking score.

11.30.5 R Implementation

```
library(timeROC); library(survival)
```

```r
data(lung)
fit <- coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
lp <- predict(fit, type = "lp")

tROC <- timeROC(T = lung$time, delta = lung$status, marker = lp,
                times = c(180, 365, 730), cause = 1,
                weighting = "marginal")
tROC
plot(tROC, time = 365, col = "steelblue")
```

11.30.6 Output & Results

`timeROC` returns time-specific AUCs with their standard errors at each requested horizon, along with empirical sensitivity-specificity pairs that can be plotted as a ROC curve. The plot shows the ROC curve at one selected time; running over multiple times reveals how discrimination evolves through follow-up.

11.30.7 Interpretation

A reporting sentence: "The Cox linear predictor's time-dependent AUC was 0.72 (95 % CI 0.65 to 0.79) at one year and 0.69 (95 % CI 0.62 to 0.76) at two years; discrimination remained stable across the two-year window relevant to clinical decision-making." Always report AUCs at clinically meaningful horizons, not just the global C-index, because discrimination can change substantially across follow-up.

11.30.8 Practical Tips

- Report AUCs at the horizons that drive clinical decisions; a five-year AUC is irrelevant for a six-month treatment decision and vice versa.
- The cumulative/dynamic definition is more common in clinical reporting; incident/dynamic is more useful when modelling hazards directly.
- `timeROC()` supports competing risks via the `cause` argument; the same workflow applies with the failure of interest specified.
- AUC alone does not capture calibration; pair time-dependent AUCs with calibration plots and Brier scores at the same times.
- For external validation, apply the development-cohort risk score to the validation cohort and compute time-dependent AUCs there; in-sample AUCs are optimistic.
- Use `weighting="cox"` or `weighting="aalen"` when censoring depends on covariates; the marginal weighting assumes censoring is uninformative given baseline covariates.

11.30.9 R Packages Used

`timeROC` for `timeROC()` with marginal, Cox, and Aalen-based weighting; `risksetROC` for an alternative kernel-smoothed implementation; `pec` for compatible prediction-error curves; `survAUC` for additional AUC variants including Uno's C-statistic.

11.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.30.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.31 Time-Varying Covariates

11.31.1 Introduction

Real survival studies routinely include covariates whose value changes during follow-up: treatment initiation, laboratory measurements repeated at each visit, secondary events that alter prognosis. Including these as time-fixed baseline values misrepresents the exposure history; pretending the baseline value applied throughout follow-up biases hazard-ratio estimates and is a leading cause of immortal-time errors. The standard remedy is the counting-process data layout, which splits each subject's follow-up into intervals during which covariate values are constant and lets the partial likelihood combine the contributions correctly.

11.31.2 Prerequisites

A working understanding of Cox regression, the partial likelihood, and the distinction between baseline (time-fixed) and time-varying covariates.

11.31.3 Theory

In counting-process format each row represents one interval $(\tau_{ij}^{\text{start}}, \tau_{ij}^{\text{stop}}]$ during which covariate values for subject i are constant. The columns are:

- `tstart`: interval start.
- `tstop`: interval end.
- `event`: 1 if the event occurred at `tstop`, otherwise 0.
- Covariates with their values during the interval.

The partial likelihood evaluates each subject's risk-set contribution using whichever interval includes the event time, so each event time naturally compares subjects with their then-current covariate values. Robust ("sandwich") variance estimators, requested via `cluster=id`, account for the within-subject correlation induced by multiple rows per subject.

11.31.4 Assumptions

Covariate values are constant within each interval and accurately measured. The covariate is "internal" if its trajectory could be affected by the event of interest (e.g., a biomarker that changes after treatment), or "external" if it is determined exogenously (e.g., scheduled treatment dose); inference from internal covariates requires careful causal interpretation.

11.31.5 R Implementation

```
library(survival)

data(cgd, package = "survival")
head(cgd)

# Multi-row-per-subject: counting-process format
fit <- coxph(Surv(tstart, tstop, status) ~ treat + inherit,
             data = cgd, cluster = id)
summary(fit)
```

11.31.6 Output & Results

`summary(fit)` produces the usual Cox-regression output with hazard ratios, confidence intervals, and Wald tests for each covariate, plus robust standard errors that respect the within-subject clustering. The number of subjects (one per cluster) and the number of events are reported alongside the number of intervals.

11.31.7 Interpretation

A reporting sentence: "After accounting for the time-varying treatment status across each subject's intervals, prophylactic treatment reduced the hazard of CGD infection by 60 % (HR 0.40, 95 % CI 0.25 to 0.63); robust standard errors were clustered on subject identifier to account for repeated intervals." When

the time-varying covariate is treatment status, always describe whether subjects could move between exposure groups and how often.

11.31.8 Practical Tips

- Use `survSplit()` (or `tmerge()` for multi-event data) to convert wide-format clinical data into the counting-process layout; doing this by hand is error-prone.
- Always specify `cluster=id` (or `cluster(id)` in the formula) so robust standard errors account for multiple rows per subject; ignoring clustering produces SEs that are too small.
- Check for overlapping intervals: the same subject should never have two rows whose intervals share any time.
- For external time-varying covariates (treatment delivered at fixed planned times), landmark analysis is a cleaner alternative that avoids the data-management overhead.
- For internal time-varying covariates (biomarkers that respond to the disease course), joint longitudinal-survival models (`JM`, `joineRML`) are preferable because they correctly handle the feedback loop.
- Be explicit in the methods section about which covariates were treated as time-varying and how often they were updated; readers cannot reproduce the analysis without this information.

11.31.9 R Packages Used

`survival` for `coxph()` with counting-process input, `survSplit()`, and `tmerge()`; `dynpred` for landmark and dynamic-prediction alternatives; `JM` and `joineRML` for joint longitudinal-survival models; `mstate` for multi-state extensions when intervals correspond to state transitions.

11.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.31.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.32 Weighted Log-Rank Tests

11.32.1 Introduction

The standard log-rank test gives every event time equal weight in the comparison between groups. That is the optimal weighting under proportional hazards but loses power when the difference between groups is concentrated at one end of follow-up — early differences that wane (typical of immunotherapy run-in effects), late differences that emerge after a delay (typical of immune-checkpoint inhibitors), or crossing-curve patterns. Weighted log-rank tests recover power in these settings by emphasising the time region where the difference is expected.

11.32.2 Prerequisites

A working understanding of the log-rank test, the proportional-hazards assumption, and the relationship between hazard ratios and survival curves at different follow-up times.

11.32.3 Theory

The general form of a weighted log-rank test is

$$\chi^2 = \frac{\left(\sum_j w(t_j)[O_j - E_j]\right)^2}{\sum_j w(t_j)^2 V_j},$$

where $O_j - E_j$ is the observed-minus-expected event count at t_j and V_j is its variance. Common weighting schemes:

- **Log-rank**: $w(t) = 1$. Optimal under proportional hazards.
- **Wilcoxon / Gehan-Breslow**: $w(t_j) = n_j$ (size of risk set). Emphasises early differences.
- **Peto-Peto**: $w(t) = \hat{S}(t)$. Similar to Wilcoxon but uses the pooled survival estimator.
- **Fleming-Harrington** $G^{\rho,\gamma}$: $w(t) = \hat{S}(t^-)^\rho\,[1-\hat{S}(t^-)]^\gamma$. $G^{1,0}$ emphasises early differences; $G^{0,1}$ emphasises late differences; $G^{1,1}$ emphasises mid-follow-up differences.

Combination tests (max-combo) compute several weights simultaneously and apply a multiple-testing correction; this preserves Type I control while gaining power across a range of alternative shapes.

11.32.4 Assumptions

Independent non-informative censoring. The chosen weight reflects the hazard pattern actually expected; post-hoc weight selection inflates Type I error and is statistical malpractice.

11.32.5 R Implementation

```r
library(survival)

data(lung)
# Standard log-rank (rho = 0)
survdiff(Surv(time, status) ~ sex, data = lung, rho = 0)

# Peto-Peto (rho = 1): weights early differences
survdiff(Surv(time, status) ~ sex, data = lung, rho = 1)

# Fleming-Harrington via nphRCT or similar package (conceptually)
```

11.32.6 Output & Results

`survdiff()` reports the chi-squared statistic, degrees of freedom, p-value, and observed-vs-expected counts under the chosen weight (`rho` argument). The `nph` and `nphRCT` packages add Fleming-Harrington $G^{\rho,\gamma}$ tests and combination tests with formal Type I control.

11.32.7 Interpretation

A reporting sentence: "The pre-specified Peto-Peto weighted log-rank test rejected the null of equal survival ($\chi^2_1 = 12.8$, $p < 0.001$); the standard log-rank produced a similar conclusion ($\chi^2_1 = 10.3$, $p = 0.001$). The Peto-Peto weighting was chosen a priori based on prior evidence that the treatment effect is concentrated in the first 18 months." Always state the weight in advance and justify it; weight-shopping is the most common abuse of these tests.

11.32.8 Practical Tips

- Pre-specify the weighting scheme in the protocol; switching weights after seeing the data inflates Type I error far above the nominal level.
- For trials with expected delayed treatment effects (immunotherapy), Fleming-Harrington $G^{0,1}$ or a max-combo test is increasingly the standard.
- Restricted-mean survival time (RMST) is a complementary approach for non-proportional hazards; report both when proportional hazards is in question.
- Combination tests (max-combo: standard log-rank, $G^{1,0}$, $G^{0,1}$, $G^{1,1}$) preserve Type I control via Hochberg-style adjustment; `nph::nphRCT` implements them.
- Inspect the Kaplan-Meier curve before choosing a weight; a clearly delayed effect rules out the standard log-rank as the primary test.
- Modern immuno-oncology trials routinely pre-specify a max-combo or RMST primary analysis; the era of one-size-fits-all log-rank reporting is

closing.

11.32.9 R Packages Used

`survival` for `survdiff()` with `rho` weighting; `nph` and `nphRCT` for Fleming-Harrington $G^{\rho,\gamma}$ tests and max-combo statistics; `survRM2` for RMST as an alternative summary; `coin` for permutation-based exact weighted log-rank tests.

11.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

11.32.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.33 Week 3, Session 2 — Competing risks and multistate models

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.33.1 Learning objectives

- Estimate cause-specific cumulative incidence with `tidycmprsk`.
- Contrast cause-specific hazards with the Fine-Gray subdistribution hazard.
- Set up a three-state illness-death multistate model with `mstate`.

11.33.2 Prerequisites

Survival analysis.

11.33.3 Background

Competing risks are events that preclude the event of interest: death from cancer competes with death from other causes. Kaplan- Meier treats the competing

event as censoring, which is wrong when the censoring is informative. Cumulative incidence functions (CIFs) give the probability that the event of interest has occurred by time t *in the presence of* competing risks.

Two hazard models are common. Cause-specific Cox models treat each cause in turn, censoring at other causes; they answer an aetiologic question ("does the covariate affect the rate of this cause?"). The Fine-Gray subdistribution hazard model fixes subjects with competing events in the risk set indefinitely; it answers a prognostic question ("does the covariate affect the cumulative incidence?").

Multistate models generalise survival to several possible transitions. The illness-death model has three states: healthy, ill, dead, with transitions healthy → ill, healthy → dead, and ill → dead. `mstate` builds these objects from tabular data.

Fine-Gray is descriptive; cause-specific Cox is causal (in the simple, aetiological sense). A covariate can increase the Fine-Gray subdistribution hazard of cause 1 either by increasing cause 1 itself or by decreasing cause 2 — it is a net effect on the cumulative incidence.

11.33.4 Setup

```
library(tidyverse)
library(survival)
library(tidycmprsk)
library(ggsurvfit)
library(mstate)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.33.5 1. Hypothesis

In the `survival::colon` data (recurrence and death as competing events), a treatment covariate affects both events. We estimate the cumulative incidence of each.

11.33.6 2. Visualise

```
data(colon, package = "survival")
# Keep death records (etype == 2) + recurrence merged; for teaching
# we build a two-event dataset:
dat <- colon |>
  filter(etype == 1) |>                      # recurrence row per subject
  transmute(id, time, status_rec = status,
            rx, age, sex) |>
  left_join(
    colon |> filter(etype == 2) |>
```

```
      transmute(id, time_d = time, status_d = status),
    by = "id"
  ) |>
  mutate(
    # event type: 0 censored, 1 recurrence, 2 death without recurrence
    etype = case_when(
      status_rec == 1 ~ 1L,
      status_d   == 1 ~ 2L,
      TRUE            ~ 0L
    ),
    etime = pmin(time, time_d, na.rm = TRUE),
    etype_f = factor(etype, levels = 0:2,
                     labels = c("censored", "recurrence", "death"))
  ) |>
  drop_na(etime, etype_f, rx)

ggplot(dat, aes(etime, fill = etype_f)) +
  geom_histogram(bins = 40, alpha = 0.7, position = "identity") +
  labs(x = "Time (days)", y = "Count", fill = NULL)
```

11.33.7 3. Assumptions

Non-informative censoring within cause; proportional hazards for Fine-Gray; for multistate, Markov transitions (future depends only on current state).

11.33.8 4. Conduct

```
cif <- cuminc(Surv(etime, etype_f) ~ rx, data = dat)
cif
```

```
ggsurvfit::ggcuminc(cif, outcome = c("recurrence", "death")) +
  labs(x = "Days", y = "Cumulative incidence")
```

```
# Fine-Gray subdistribution hazard for recurrence
fg <- crr(Surv(etime, etype_f) ~ rx + age + sex, data = dat,
          failcode = "recurrence")
fg
```

```
# Multistate: illness-death using mstate on a simple simulated dataset
n <- 300
sim <- tibble(
  id = seq_len(n),
  t_ill   = rexp(n, rate = 0.02),
  t_death = rexp(n, rate = 0.01)
) |>
```

```
  mutate(
    t1 = pmin(t_ill, t_death, 50),
    s1 = if_else(t_ill < t_death & t_ill <= 50, 1L, 0L),
    s2 = if_else(t_death <= 50 & t_death < t_ill, 1L, 0L)
  )

tmat <- transMat(x = list(c(2, 3), c(3), c()),
                 names = c("healthy", "ill", "dead"))
tmat
```

11.33.9 5. Concluding statement

Cause-specific cumulative incidence functions differed by treatment arm in the colon data, with the Fine-Gray subdistribution hazard estimated above. The three-state illness- death transition matrix (healthy → ill → dead) shows how `mstate` represents the problem; the full fit is straightforward once the long-format dataset is built via `msprep`.

11.33.10 Common pitfalls

- Reporting 1 − KM as the cumulative incidence in the presence of competing risks.
- Using Fine-Gray when the question is aetiological.
- Reporting both cause-specific and Fine-Gray hazard ratios without distinguishing them.
- Assuming Markov transitions in a multistate model when history matters.

11.33.11 Further reading

- Fine JP, Gray RJ (1999), *A proportional hazards model for the subdistribution of a competing risk.*
- Putter H, Fiocco M, Geskus RB (2007), *Tutorial in biostatistics: competing risks and multi-state models.*
- Austin PC, Fine JP (2017), *Practical recommendations for reporting Fine-Gray model analyses.*

11.33.12 Session info

```
sessionInfo()
```

11.33.13 See also — chapter index

- Methods in this chapter

- Find your method (Chapter 0)

11.34 Week 3, Session 4 — Survival ML (random survival forest, DeepSurv)

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

11.34.1 Learning objectives

- Fit a random survival forest and interpret its variable importance and risk stratification.
- Describe the DeepSurv architecture conceptually and when it is preferred over the forest.
- Compare against a Cox model on the same features.

11.34.2 Prerequisites

Cox regression and censoring.

11.34.3 Background

Survival analysis deals with time-to-event data in the presence of censoring. The Cox proportional-hazards model is the workhorse linear method; random survival forests (RSF) extend the idea of random forests to right-censored outcomes, using a log-rank split rule and Harrell's concordance as the default fit criterion. They handle nonlinearities and interactions automatically and are a good first nonparametric benchmark.

DeepSurv is a neural-network generalisation of the Cox partial likelihood: the linear predictor is replaced by a feed-forward network. In small biomedical datasets it rarely beats a well-tuned RSF or a Cox model with splines, but the approach scales to large cohorts and to inputs where a representation must be learned.

Evaluation of survival models uses time-integrated concordance (Harrell's C), the integrated Brier score, and — increasingly — decision curves at a clinically relevant horizon. Reporting only C on a single test set is weak evidence.

11.34.4 Setup

```
library(tidyverse)
library(survival)
library(randomForestSRC)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.34.5 1. Goal

Fit an RSF on `survival::pbc` and compare to a Cox model using Harrell's concordance.

11.34.6 2. Approach

```
d <- survival::pbc |>
  as_tibble() |>
  drop_na(bili, albumin, age, edema, protime, stage) |>
  mutate(status = ifelse(status == 2, 1, 0))

ggplot(d, aes(time / 365.25, fill = factor(status))) +
  geom_histogram(bins = 30, alpha = 0.7, position = "identity") +
  labs(x = "follow-up (years)", y = "count", fill = "event")
```

11.34.7 3. Execution

```
fit_cox <- coxph(Surv(time, status) ~ bili + albumin + age + edema +
                   protime + stage, data = d)
fit_rsf <- rfsrc(Surv(time, status) ~ bili + albumin + age + edema +
                   protime + stage,
                 data = as.data.frame(d), ntree = 500,
                 importance = TRUE)
fit_rsf
```

```
tibble(var = names(fit_rsf$importance),
       imp = as.numeric(fit_rsf$importance)) |>
  arrange(desc(imp)) |>
  ggplot(aes(reorder(var, imp), imp)) +
  geom_col() + coord_flip() + labs(x = NULL, y = "VIMP")
```

11.34.8 4. Check

Compare concordance on held-out data.

```
idx <- sample(nrow(d), 0.7 * nrow(d))
tr <- d[idx, ]; te <- d[-idx, ]
cox2 <- coxph(Surv(time, status) ~ bili + albumin + age + edema +
                protime + stage, data = tr)
rsf2 <- rfsrc(Surv(time, status) ~ bili + albumin + age + edema +
                protime + stage,
              data = as.data.frame(tr), ntree = 500)
```

```
c_cox <- survival::concordance(cox2, newdata = te)$concordance
p_rsf <- predict(rsf2, newdata = as.data.frame(te))$predicted
c_rsf <- survival::concordance(Surv(te$time, te$status) ~ p_rsf,
                               reverse = TRUE)$concordance
c(cox = c_cox, rsf = c_rsf)
```

DeepSurv conceptual sketch.

```
library(torch)
# A DeepSurv-style loss computes the partial log-likelihood on the
# sorted risk set. The network output h(x) replaces the Cox linear
# predictor; the loss is  -(sum_{i:event}  h(x_i) - log sum_{j in R_i}
 ↪  exp(h(x_j))).
deepsurv_loss <- function(risk, time, event) {
  ord <- order(time, decreasing = TRUE)
  risk <- risk[ord]; event <- event[ord]
  log_cumsum <- torch_logcumsumexp(risk, dim = 1)
  -torch_sum(event * (risk - log_cumsum)) / torch_sum(event)
}
```

11.34.9 5. Report

> On `survival::pbc`, a random survival forest achieved test concordance `round(c_rsf,3)` compared with `round(c_cox,3)` for a multivariable Cox model. Bilirubin, albumin, and age dominated VIMP. A DeepSurv network with the partial-likelihood loss would be the natural next step on a larger cohort.

On this sample size, the RSF and Cox model are close; on larger datasets with nonlinearities, the forest often pulls ahead.

11.34.10 Common pitfalls

- Treating the Cox proportional-hazards assumption as always met; it often is not, and the RSF is more robust to violation.
- Comparing models without the same censoring distribution in the test set.
- Reporting only C without a calibration or decision curve at the horizon of interest.

11.34.11 Further reading

- Ishwaran H et al. (2008), *Random survival forests.*
- Katzman JL et al. (2018), *DeepSurv: Personalized treatment recommender system using a Cox proportional hazards deep neural network.*

11.34.12 Session info

```
sessionInfo()
```

11.34.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.35 Week 4, Session 3 — Survival primer: KM, log-rank, Cox PH

Inference lab: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.35.1 Learning objectives

- Estimate and plot a Kaplan-Meier survival curve for a two-group comparison.
- Run and interpret a log-rank test.
- Fit a Cox proportional-hazards model, check the PH assumption, and read a hazard ratio honestly.

11.35.2 Prerequisites

Comfort with GLMs and with reading a regression table.

11.35.3 Background

Survival analysis handles outcomes that are times to an event, where some subjects have not yet had the event when observation ends — the familiar right-censoring problem. Ignoring censoring by throwing out the censored observations biases the estimator; pretending the censored observations are event-free biases it the other way. The survival machinery handles censoring correctly by reasoning at each event time about who is still at risk.

Three tools cover most clinical applications: the Kaplan-Meier estimator for a non-parametric survival curve, the log-rank test for a two-group comparison, and Cox proportional-hazards regression for adjustment and multivariable modelling. The PH assumption — that the hazard ratio is constant over time — deserves explicit checking; when it fails, a time-varying coefficient or a stratified model is usually the right fix.

11.35.4 Setup

```
library(tidyverse)
library(survival)
library(ggsurvfit)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.35.5 1. Hypothesis

Using `survival::lung`, compare survival between male and female patients. H_0: hazard ratio $= 1$. H_1: hazard ratio $\neq 1$. $\alpha = 0.05$.

11.35.6 2. Visualise

```
lung2 <- lung |>
  mutate(sex = factor(sex, levels = 1:2, labels = c("male", "female")))

km <- survfit2(Surv(time, status) ~ sex, data = lung2)

km |>
  ggsurvfit() +
  add_confidence_interval() +
  add_risktable() +
  labs(x = "Days", y = "Survival probability")
```

11.35.7 3. Assumptions

Two assumptions are worth checking: non-informative censoring (patients censored at a given time are representative of those still at risk) and proportional hazards.

```
cox_fit <- coxph(Surv(time, status) ~ sex + age, data = lung2)
cox.zph(cox_fit)
```

A large p-value on each row and a non-systematic Schoenfeld residual plot support the PH assumption; a small p calls for a time-varying coefficient or stratification.

11.35.8 4. Conduct

```
lr <- survdiff(Surv(time, status) ~ sex, data = lung2)
lr
```

```
tidy(cox_fit, exponentiate = TRUE, conf.int = TRUE)
```

11.35.9 5. Concluding statement

```
hr_row <- tidy(cox_fit, exponentiate = TRUE, conf.int = TRUE) |>
  filter(term == "sexfemale")
```

> Female sex was associated with a lower risk of death relative to male sex (HR `round(hr_row$estimate,2)`; 95% CI `round(hr_row$conf.low,2)` to `round(hr_row$conf.high,2)`; p = `signif(hr_row$p.value,2)`), adjusted for age. The proportional-hazards assumption was not clearly violated (Schoenfeld test, all $p > 0.05$).

11.35.10 Common pitfalls

- Reporting a median survival that is not reached and calling it "undefined" in the results. Say so explicitly.
- Ignoring the PH check because the output looks clean.
- Comparing Kaplan-Meier curves by eye alone — add the log-rank test and the risk table.
- Collapsing a competing event (e.g. death from another cause) into censoring without justification.

11.35.11 Further reading

- Harrell FE. *Regression Modeling Strategies*, ch. 20.
- Therneau TM & Grambsch PM. *Modeling Survival Data.*

11.35.12 Session info

```
sessionInfo()
```

11.35.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.36 Week 3, Session 5 — Time-dependent Brier, IPA, external validation

Testing labs use the main template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.36.1 Learning objectives

- Compute a time-dependent Brier score and the Index of Prediction Accuracy (IPA).
- Simulate an external-validation cohort with distribution shift and evaluate a fitted model on it.
- Describe why a model can have a high internal AUC and a bad external IPA.

11.36.2 Prerequisites

Survival ML; calibration.

11.36.3 Background

Validation is the point at which a prediction model either earns its keep or does not. Internal validation — CV or the bootstrap on the training data — controls optimism from the fitting step. External validation applies the fixed model to an independent cohort and measures discrimination, calibration, and clinical utility. Time- dependent Brier scores and IPA (1 − Brier/Brier_null) summarise the cost of a probabilistic prediction at a given horizon, combining discrimination and calibration into a single number per time.

Real biomedical models routinely lose performance on external cohorts. Distribution shift (different age, comorbidities, measurement protocols) is the usual reason; miscalibration is the usual failure mode. This lab simulates both and shows the diagnostic signature each leaves behind.

TRIPOD-AI asks you to report all three — discrimination, calibration, clinical utility — on both internal and external data, and to disclose the shift between them. A reader who only sees AUC on internal data has no basis for deploying the model.

11.36.4 Setup

```
library(tidyverse)
library(survival)
library(pROC)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.36.5 1. Hypothesis

A Cox model fit on a development cohort retains its calibration and discrimination on an external cohort with shifted covariate distribution.

11.36.6 2. Visualise

```
make_cohort <- function(n, mu_x = 0, beta = 0.6) {
  x <- rnorm(n, mu_x)
  lp <- beta * x
  t <- rexp(n, rate = exp(lp - 1))
  c <- rexp(n, rate = 0.1)
  time <- pmin(t, c)
  event <- as.integer(t <= c)
  tibble(x = x, time = time, event = event)
}
dev  <- make_cohort(500, mu_x = 0)
ext  <- make_cohort(500, mu_x = 1.0)   # distribution shift
bind_rows(dev |> mutate(cohort = "dev"),
          ext |> mutate(cohort = "ext")) |>
  ggplot(aes(x, fill = cohort)) +
  geom_histogram(bins = 30, alpha = 0.7, position = "identity")
```

11.36.7 3. Assumptions

Independent right censoring; Cox proportional hazards in the development cohort; no time-varying covariates.

11.36.8 4. Conduct

Fit on dev; predict survival at a fixed horizon on both cohorts.

```
fit <- coxph(Surv(time, event) ~ x, data = dev)
horizon <- 3

predict_surv <- function(fit, newdata, t) {
  bh <- basehaz(fit, centered = FALSE)
  H0 <- approx(bh$time, bh$hazard, xout = t, rule = 2)$y
  lp <- predict(fit, newdata = newdata, type = "lp")
  exp(-H0 * exp(lp))
}

s_dev <- predict_surv(fit, dev, horizon)
s_ext <- predict_surv(fit, ext, horizon)

# Observed indicator (for simple Brier w/out censoring at horizon)
obs_dev <- dev$event == 1 & dev$time <= horizon
obs_ext <- ext$event == 1 & ext$time <= horizon

brier <- function(p_event, obs) mean((p_event - obs)^2)
brier_null <- function(obs) mean((mean(obs) - obs)^2)
```

```
b_dev <- brier(1 - s_dev, obs_dev); b0_dev <- brier_null(obs_dev)
b_ext <- brier(1 - s_ext, obs_ext); b0_ext <- brier_null(obs_ext)
ipa_dev <- 1 - b_dev / b0_dev
ipa_ext <- 1 - b_ext / b0_ext

auc_dev <- as.numeric(auc(roc(obs_dev, 1 - s_dev, quiet = TRUE)))
auc_ext <- as.numeric(auc(roc(obs_ext, 1 - s_ext, quiet = TRUE)))

tibble(cohort = c("dev", "ext"),
       auc = c(auc_dev, auc_ext),
       brier = c(b_dev, b_ext),
       ipa  = c(ipa_dev, ipa_ext))
```

A calibration plot at the horizon.

```
mk_cal <- function(p, y, lab) {
  tibble(p = p, y = y, cohort = lab) |>
    mutate(bin = cut(p, quantile(p, seq(0, 1, by = 0.1)),
                     include.lowest = TRUE)) |>
    group_by(cohort, bin) |>
    summarise(mean_pred = mean(p), obs = mean(y), .groups = "drop")
}
bind_rows(mk_cal(1 - s_dev, obs_dev, "dev"),
          mk_cal(1 - s_ext, obs_ext, "ext")) |>
  ggplot(aes(mean_pred, obs, colour = cohort)) +
  geom_point() + geom_line() +
  geom_abline(slope = 1, intercept = 0, colour = "grey50") +
  labs(x = "predicted event prob at horizon",
       y = "observed event proportion")
```

11.36.9 5. Concluding statement

> On a simulated external cohort with shifted covariate distribution, a Cox model fit on the development cohort retained discrimination (AUC `round(auc_ext,2)`) but lost calibration, with IPA falling from `round(ipa_dev,2)` (development) to `round(ipa_ext,2)` (external). The calibration plot showed systematic over-prediction in the shifted cohort.

Distribution shift does not always show up in AUC; it often shows up first in calibration. Reporting Brier/IPA alongside AUC catches this failure mode.

11.36.10 Common pitfalls

- Reporting AUC as the only discrimination metric on an external cohort.

- Ignoring censoring when computing a Brier score at a horizon; `pec` or `riskRegression` handle this properly.
- Recalibrating on external data and then reporting the resulting performance as external validation.

11.36.11 Further reading

- Kattan MW, Gerds TA (2018), *The index of prediction accuracy: an intuitive measure useful for evaluating risk prediction models*.
- Steyerberg EW, *Clinical Prediction Models*, ch. 15–17.

11.36.12 Session info

```
sessionInfo()
```

11.36.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.37 Week 3, Session 1 — Time-varying covariates and immortal-time bias

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.37.1 Learning objectives

- Recognise immortal-time bias in a survival analysis.
- Correct it using either a landmark analysis or a time-dependent Cox model.
- Describe time-varying covariates in counting-process notation.

11.37.2 Prerequisites

Survival basics; `coxph` syntax.

11.37.3 Background

Immortal-time bias arises when a period during which the outcome cannot occur (by definition of the exposure) is misclassified as follow-up for one of the exposure groups. A classic example: "users of drug X" are defined as anyone who ever received a prescription; these people must by definition have survived until their first prescription, and if you start their follow-up at baseline rather than at the

prescription, you give them "immortal" time that biases the hazard ratio toward protection.

Two standard fixes exist. Landmark analysis picks a time after baseline and classifies exposure based on what happened before that time; follow-up starts at the landmark and the bias disappears. Time-dependent Cox models treat exposure as a function of time and let patients change exposure group when they actually start the drug, using counting-process (start, stop, event) data.

Time-dependent covariates can be external (calendar time, drug availability) or internal (a lab value that evolves with the disease). Internal time-dependent covariates can introduce their own bias when conditioning on them — they may be mediators of the very effect you are studying.

11.37.4 Setup

```
library(tidyverse)
library(survival)
library(ggsurvfit)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.37.5 1. Hypothesis

We simulate a cohort where every patient has the same true hazard. Some patients start a drug at a random later time. A naive analysis that counts drug users from baseline should produce a spurious protective effect.

11.37.6 2. Visualise

```
n <- 500
dat <- tibble(
  id = seq_len(n),
  t_event = rexp(n, rate = 0.02),        # true event time
  t_drug  = rexp(n, rate = 0.05),        # time of drug start
  ever_drug = t_drug < t_event,          # did they start drug before
   ↪  event?
  t_drug_obs = pmin(t_drug, t_event),
  t_obs   = pmin(t_event, 100),          # admin censoring at 100
  event   = as.integer(t_event <= 100)
)

dat |>
  ggplot(aes(t_obs, fill = ever_drug)) +
  geom_histogram(bins = 30, alpha = 0.6,
                 position = "identity") +
```

```
  labs(x = "Observed time", y = "Count", fill = "Ever drug?")
```

11.37.7 3. Assumptions

Proportional hazards (for Cox), non-informative censoring, and independence of event times across patients. The "ever drug" split is deliberately biased by construction.

11.37.8 4. Conduct

```
# Naive (biased) analysis
fit_naive <- coxph(Surv(t_obs, event) ~ ever_drug, data = dat)
summary(fit_naive)
```

```
# Landmark at time 20: only include patients still at risk at 20,
# classify by drug status as of 20.
landmark <- 20
dat_lm <- dat |>
  filter(t_obs > landmark) |>
  mutate(drug_lm = t_drug < landmark,
         t_obs_lm = t_obs - landmark)

fit_lm <- coxph(Surv(t_obs_lm, event) ~ drug_lm, data = dat_lm)
summary(fit_lm)
```

```
# Time-dependent Cox: counting-process format
td <- dat |>
  transmute(
    id,
    tstart = 0,
    tstop  = if_else(ever_drug, t_drug_obs, t_obs),
    event  = if_else(ever_drug, 0L, event),
    drug   = 0L
  ) |>
  bind_rows(
    dat |>
      filter(ever_drug) |>
      transmute(id,
                tstart = t_drug_obs,
                tstop  = t_obs,
                event  = event,
                drug   = 1L)
  ) |>
  filter(tstop > tstart) |>
  arrange(id, tstart)
```

```
fit_td <- coxph(Surv(tstart, tstop, event) ~ drug, data = td)
summary(fit_td)
```

11.37.9 5. Concluding statement

The naive Cox analysis produced a hazard ratio of `round(exp(coef(fit_naive)),2)` for ever-drug — spuriously protective, because immortal time was counted as exposed follow-up. A landmark analysis at time 20 gave HR `round(exp(coef(fit_lm)),2)`, and a time-dependent Cox model gave HR `round(exp(coef(fit_td)),2)`, both close to the null as expected from the simulated truth.

11.37.10 Common pitfalls

- Defining exposure over the whole follow-up based on events that occur during it.
- Picking a landmark time after visually inspecting event curves.
- Conditioning on a post-baseline mediator via a time-dependent covariate and still calling it a total effect.
- Forgetting to split the risk set at the exposure change time in counting-process data.

11.37.11 Further reading

- Suissa S (2008), *Immortal time bias in pharmaco-epidemiology.*
- Therneau TM, Grambsch PM. *Modeling Survival Data: Extending the Cox Model.*
- Dafni U (2011), *Landmark analysis at the 25-year landmark point.*

11.37.12 Session info

```
sessionInfo()
```

11.37.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 12

Bioinformatics

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

RNA-seq differential expression, pathway and GO enrichment, and single-cell RNA-seq. The chapter assumes basic familiarity with `Bioconductor` and points back to the regression and ML chapters for the underlying machinery.

This chapter contains **50 method pages** and **3 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

12.1 Method pages

Method	Source slug
Alignment with BWA and Bowtie	`bwa-bowtie-alignment`
ATAC-Seq Analysis	`atac-seq-analysis`
Batch Correction with ComBat	`batch-correction-combat`
Bulk RNA-seq Differential Expression with DESeq2	`bulk-rnaseq-differential-expression`
Cell-Type Annotation	`scrna-cell-type-annotation`
ChIP-Seq Analysis	`chip-seq-analysis`
Copy Number Variation Analysis	`cnv-analysis`
Counting Reads with featureCounts	`counting-reads-feature-counts`
Differential Expression with edgeR	`edger-differential-expression`
Differential Expression with limma-voom	`limma-voom`
DNA Methylation Analysis	`methylation-analysis`
Drug-Target Interaction Mining	`drug-target-interaction`
FASTQ Quality Control	`fastq-quality-control`
Finding Marker Genes	`scrna-marker-genes`

Method	Source slug
Gene Annotation with biomaRt	`biomart-annotation`
Gene Ontology Enrichment	`gene-ontology-enrichment`
GSEA Preranked Analysis	`gsea-preranked`
GSVA Single-Sample Enrichment	`gsva-single-sample`
Heatmaps for RNA-seq	`heatmaps-rnaseq`
Integration with Harmony	`scrna-integration-harmony`
KEGG Pathway Enrichment	`pathway-enrichment-kegg`
MA Plots	`ma-plots`
Metagenomic Profiling	`metagenomic-profiling`
Microbiome Analysis with DADA2	`microbiome-dada2`
Microbiome Diversity Metrics	`microbiome-diversity`
Multi-Omics Integration	`multiomics-integration`
Multiple Sequence Alignment	`msa-biostrings`
PCA of RNA-seq Samples	`pca-of-rnaseq`
Phylogenetic Trees with ape	`phylogenetic-trees-ape`
Population Genetics Basics	`population-genetics-basics`
Protein Structure Prediction	`protein-structure-prediction`
Proteomics with MSstats	`proteomics-msstats`
Pseudoalignment with Salmon and Kallisto	`salmon-kallisto-pseudoalignment`
Read Trimming and Adapter Removal	`read-trimming-adapters`
RNA-seq Normalisation	`rnaseq-normalization`
scRNA Clustering	`scrna-clustering`
scRNA Normalisation	`scrna-normalization`
scRNA QC and Filtering	`scrna-qc-filtering`
Sequence Alignment: Overview	`sequence-alignment-overview`
Single-Cell with Seurat	`single-cell-seurat-intro`
Spatial Transcriptomics	`spatial-transcriptomics`
STAR: Spliced Alignment	`star-spliced-alignment`
Structural Variant Detection	`structural-variant-detection`
Surrogate Variable Analysis (SVA)	`surrogate-variable-analysis-sva`
Trajectory Analysis	`scrna-trajectory-analysis`
Transcript-to-Gene Summarisation	`tx-to-gene-summarization`
Variant Annotation with VEP	`annotation-variant-vep`
Variant Calling with GATK	`variant-calling-gatk`
VCF Manipulation	`vcf-manipulation`
Volcano Plots	`volcano-plots`

12.2 Labs

Lab
RNA-seq with DESeq2 / edgeR
Enrichment analysis (GSEA / ORA)
scRNA-seq with Seurat

12.3 Variant Annotation with VEP

12.3.1 Introduction

Ensembl's Variant Effect Predictor (VEP) assigns each variant a predicted consequence (missense, synonymous, splice-donor, intronic, etc.), gene and transcript IDs, and pathogenicity scores from predictors (SIFT, PolyPhen, CADD, REVEL). It is the standard annotator for variant interpretation in clinical genetics and population genomics.

12.3.2 Prerequisites

VCF file; Ensembl or RefSeq transcript database.

12.3.3 Theory

VEP traverses each variant, intersects it with transcript structures, and emits one record per overlapping transcript with the predicted functional consequence. Gene-level summaries pick the most severe consequence across transcripts (MANE-select or canonical transcript is standard for clinical use).

12.3.4 Assumptions

Reference build matches the VCF (GRCh37 vs GRCh38); transcript database is current.

12.3.5 R Implementation

VEP runs on the command line (Perl). For programmatic annotation in R, use `VariantAnnotation::locateVariants()` against a `TxDb` plus `predictCoding()` for protein consequences.

```
library(VariantAnnotation)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(BSgenome.Hsapiens.UCSC.hg19)

vcf_file <- system.file("extdata", "chr22.vcf.gz", package =
 ↪  "VariantAnnotation")
vcf <- readVcf(vcf_file, "hg19")
seqlevelsStyle(vcf) <- "UCSC"
```

```
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
loc  <- locateVariants(rowRanges(vcf), txdb, CodingVariants())
table(loc$LOCATION)

pc <- predictCoding(vcf, txdb, Hsapiens)
head(mcols(pc)[, c("CONSEQUENCE", "REFAA", "VARAA")])
```

A representative VEP command (not run here):

```
vep -i input.vcf -o output.vcf --cache --sift p --polyphen p --canonical
```

12.3.6 Output & Results

Per-variant, per-transcript annotation with consequence (missense_variant, synonymous_variant, etc.) and amino-acid substitution information.

12.3.7 Interpretation

"VEP annotation identified 12 missense variants predicted deleterious by SIFT and PolyPhen; one nonsense variant in BRCA1 was classified as likely pathogenic."

12.3.8 Practical Tips

- Always check assembly/transcript version match between caller and annotator.
- For clinical work, use VEP's `--pick` or `--flag_pick_allele_gene` to obtain a single canonical annotation per variant.
- Add pathogenicity plugins (CADD, REVEL, SpliceAI) for deleteriousness prediction.
- `ANNOVAR` is a competing annotator with a different gene-model approach; results can differ at splice boundaries.
- For batch annotation of millions of variants, VEP's offline cache mode is essential; REST/web queries do not scale.

12.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.3.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.4 ATAC-Seq Analysis

12.4.1 Introduction

ATAC-seq (Assay for Transposase-Accessible Chromatin using sequencing) measures genome-wide chromatin accessibility by tagging accessible DNA regions with a hyperactive Tn5 transposase. Peaks in the resulting alignment density correspond to open chromatin — regulatory elements such as promoters and enhancers — and differential accessibility analysis between conditions reveals cell-state-specific regulatory changes. ATAC-seq has largely replaced DNase-seq for accessibility profiling because of its lower input requirement (10{,}000 cells vs millions) and simpler protocol.

12.4.2 Prerequisites

A working understanding of read alignment, BAM files, peak-calling concepts (typically MACS2), and regulatory genomics (promoters, enhancers, transcription factor binding).

12.4.3 Theory

The standard ATAC-seq pipeline:

1. **QC**: TSS enrichment score, fragment-length distribution with characteristic nucleosomal peaks (mono-, di-, tri-nucleosomal at approximately 150, 300, 450 bp).
2. **Duplicate and mitochondrial-read removal**: typical mitochondrial fraction is 20–50 % of reads and must be removed before peak calling.
3. **Peak calling**: MACS2 with Tn5-specific flags (`--nomodel--shift-75--extsize150`) treats each end of the fragment as a Tn5 cut site.
4. **Consensus peak set**: combine peaks across replicates and conditions for joint quantitative analysis.
5. **Differential accessibility**: DiffBind, edgeR, or DESeq2 on peak-level read counts.
6. **Peak annotation**: ChIPseeker assigns peaks to nearest gene and annotates promoter vs distal categories.

12.4.4 Assumptions

Tn5 tagmentation is approximately uniform across accessible regions; sequencing depth is sufficient ($>$ 50 million reads per sample is typical); mitochondrial reads

have been removed before peak calling.

12.4.5 R Implementation

```
library(ATACseqQC); library(ChIPseeker)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)

# Fragment-length diagnostic from an example ATAC bam shipped with
 ↪  ATACseqQC
bam <- system.file("extdata", "GL1.bam", package = "ATACseqQC")
# fragSize <- fragSizeDist(bam, bamFiles.labels = "example")

# Peak annotation example with a toy peak GRanges
set.seed(2026)
peaks <- GRanges(
  seqnames = rep("chr1", 50),
  ranges = IRanges(start = sort(sample(1e7:1.05e7, 50)),
                   width = 500)
)
anno <- annotatePeak(peaks,
                     TxDb = TxDb.Hsapiens.UCSC.hg19.knownGene,
                     level = "gene",
                     verbose = FALSE)
as.data.frame(anno)[1:5, c("seqnames", "start", "annotation", "geneId")]
```

12.4.6 Output & Results

`ATACseqQC::fragSizeDist()` produces the diagnostic fragment-size distribution; visible peaks at the nucleosomal lengths confirm successful tagmentation. `ChIPseeker::annotatePeak()` assigns each peak to the nearest gene and categorises it as promoter, intronic, intergenic, etc.

12.4.7 Interpretation

A reporting sentence: "ATAC-seq on CD8 and regulatory T cells (n = 4 per group) identified 25{,}412 reproducible peaks at IDR < 0.05; ChIPseeker annotation showed 35 % promoter-proximal and 65 % distal-regulatory locations. Differential accessibility analysis identified 1{,}240 peaks at FDR < 0.05, enriched for AP-1 family motifs (Jun, Fos)." Always report TSS enrichment, peak counts, and the consensus peak strategy.

12.4.8 Practical Tips

- Remove duplicates and mitochondrial reads with `samtoolsmarkdup` and chromosome filtering before peak calling; mitochondrial reads dominate ATAC libraries and confound peak calling if retained.

- Check TSS enrichment score (computed by `ATACseqQC` or by `chrom.sizes`-aware tools); scores above 7 are typical for fresh tissue, scores above 4 are acceptable for frozen samples.
- For differential accessibility, define a consensus peak set across replicates (DiffBind-style) and quantify all samples on the same regions; calling peaks separately per sample produces unstable comparisons.
- Motif enrichment via `chromVAR` provides per-sample transcription-factor activity scores; pair with differential accessibility to identify TFs driving the observed changes.
- For single-cell ATAC-seq (scATAC-seq), use `Signac` (Seurat-flavoured) or `ArchR`; the workflow differs substantially from bulk ATAC.
- Use BlackList regions (ENCODE) to exclude artefact-prone genomic regions before downstream analysis.

12.4.9 R Packages Used

`ATACseqQC` for fragment-size distribution and TSS enrichment QC; `ChIPseeker` for peak annotation; `DiffBind` for differential-binding analysis; `chromVAR` for motif accessibility scores; `Signac` and `ArchR` for single-cell ATAC-seq; `Rsubread` and `GenomicAlignments` for upstream alignment and quantification.

12.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.4.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.5 Batch Correction with ComBat

12.5.1 Introduction

Batch effects — systematic technical variation introduced by different sequencing dates, instruments, library-prep operators, or reagent lots — bias gene-expression measurements in ways that can swamp biological signal if not addressed. ComBat is the most widely used batch-correction method, applying an empirical-Bayes adjustment to centre and scale each gene's distribution per batch. It is appropriate when batches are known and condition is not confounded

with batch; the alternative — including batch as a covariate in differential-expression models — preserves more information when downstream analysis explicitly models batch.

12.5.2 Prerequisites

A working understanding of normalisation, the distinction between technical and biological variation, and basic empirical-Bayes shrinkage.

12.5.3 Theory

ComBat models each gene's expression as

$$Y_{ij} = \alpha_j + X\beta_j + \gamma_{bj} + \delta_{bj}\varepsilon_{ij},$$

with sample i, gene j, batch b, additive batch effect γ_{bj}, and multiplicative batch effect δ_{bj}. The empirical-Bayes step shrinks per-gene batch-effect estimates toward a common distribution across genes, stabilising estimation when batches contain few samples. The output is an expression matrix with batch effects removed but biological condition (specified via `mod`) preserved.

ComBat-Seq (in `sva`) is the count-data analogue, designed for raw RNA-seq counts before normalisation; standard ComBat assumes log-normalised input.

12.5.4 Assumptions

Batches are known; biological condition is not perfectly confounded with batch (otherwise correction is impossible); sufficient samples per batch for stable empirical-Bayes shrinkage.

12.5.5 R Implementation

```
library(sva)

set.seed(2026)
# Simulated: 10 genes x 20 samples, 2 batches, 2 conditions
expr <- matrix(rnorm(10 * 20), 10, 20)
batch <- c(rep(1, 10), rep(2, 10))
condition <- rep(c("A", "B"), 10)

# Add batch effect
expr[, batch == 2] <- expr[, batch == 2] + 2

mod <- model.matrix(~ condition)
expr_corrected <- ComBat(dat = expr, batch = batch, mod = mod)
```

```
# PCA before/after
# prcomp(t(expr))$x[, 1:2]; prcomp(t(expr_corrected))$x[, 1:2]
```

12.5.6 Output & Results

`ComBat()` returns the batch-corrected expression matrix; `ComBat_seq()` returns batch-corrected counts. PCA before and after correction is the standard QC check: condition-driven clustering should remain or strengthen, while batch-driven separation should diminish substantially.

12.5.7 Interpretation

A reporting sentence: "ComBat correction with condition included as a covariate (`mod=model.matrix(~condition)`) reduced batch variance from 35 % of PC1 to under 5 %, while preserving the condition-driven separation that explains 28 % of the remaining variance. The corrected matrix was used for downstream visualisation and clustering." Always report which covariates were preserved during correction.

12.5.8 Practical Tips

- Do *not* run ComBat when condition is fully confounded with batch — the correction will remove the biological signal along with the batch effect.
- Use `ComBat_seq()` for raw RNA-seq counts; standard ComBat is for log-normalised data.
- For differential-expression analysis, consider including batch as a covariate in the DE design (`design=~batch+condition` in DESeq2/edgeR/limma) rather than running ComBat first; the model-based approach is generally preferred.
- Always report batch-correction methods explicitly in publications; reviewers expect clarity about how batch was handled.
- Alternative methods include RUVSeq (latent factor extraction), svaseq (surrogate variable analysis), and `limma::removeBatchEffect()` (simple linear adjustment); the choice depends on whether batches are known.
- Pair correction with PCA before and after as a routine QC step; visible improvement is the validation for whether the correction worked.

12.5.9 R Packages Used

`sva::ComBat()` and `ComBat_seq()` for the canonical empirical-Bayes batch correction; `RUVSeq` for unsupervised factor-based correction; `limma::removeBatchEffect()` for simple linear batch removal; `harmony` for batch correction in single-cell RNA-seq workflows.

12.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.5.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.6 Gene Annotation with biomaRt

12.6.1 Introduction

`biomaRt` provides a programmatic interface to Ensembl BioMart for converting between gene identifier systems (Ensembl gene IDs, Entrez IDs, HGNC symbols, RefSeq, UniProt), retrieving genomic coordinates, transcript structures, GO term annotations, and orthologues across species. It is indispensable whenever a gene list — from differential expression, GWAS, or any other genomic analysis — needs to be enriched with curated annotation. The `biomaRt` workflow builds queries that filter, attribute, and merge in a single call, replacing dozens of manual lookups across web interfaces.

12.6.2 Prerequisites

A working understanding of gene identifier types (Ensembl vs Entrez vs HGNC symbol vs RefSeq), the difference between gene-level and transcript-level annotation, and basic familiarity with R data frames.

12.6.3 Theory

A `biomaRt` query targets a Mart (database) and dataset (species), specifying:

- **Filters**: which genes to query (input IDs and their type).
- **Attributes**: which annotations to return.
- **Values**: the actual identifier values.

The core call `getBM(attributes,filters,values,mart)` issues the query and returns a data frame. For reproducibility, pin the Ensembl release with `host="<month><year>.archive.ensembl.org"`; the default `host="www.ensembl.org"` points to the current release, which changes over time.

`biomaRt` also supports cross-species queries via `getLDS()` (linked datasets) for orthologue lookup, transcript-level queries for splicing analyses, and SNP queries against Ensembl variation.

12.6.4 Assumptions

Internet connectivity to the Ensembl BioMart server (rare downtime can interrupt workflows); input identifiers match the chosen Mart's species and identifier type.

12.6.5 R Implementation

```
library(biomaRt)

# Illustrative (requires internet; swap to offline cache in production):
# mart <- useMart("ensembl", dataset = "hsapiens_gene_ensembl",
#                 host = "www.ensembl.org")
#
# genes_symbol <- c("TP53", "BRCA1", "EGFR")
# annot <- getBM(
#   attributes = c("hgnc_symbol", "ensembl_gene_id",
#                  "entrezgene_id", "chromosome_name",
#                  "start_position", "end_position"),
#   filters    = "hgnc_symbol",
#   values     = genes_symbol,
#   mart       = mart
# )
# annot

# Simulated equivalent for offline illustration
set.seed(2026)
annot <- data.frame(
  hgnc_symbol      = c("TP53", "BRCA1", "EGFR"),
  ensembl_gene_id  = c("ENSG00000141510", "ENSG00000012048",
   ↪  "ENSG00000146648"),
  entrezgene_id    = c(7157, 672, 1956),
  chromosome_name  = c("17", "17", "7"),
  start_position   = c(7668402, 43044295, 55019032),
  end_position     = c(7687550, 43125483, 55211628)
)
annot
```

12.6.6 Output & Results

A data frame mapping the input identifiers to all requested attributes. In practice, the output is merged with differential-expression results or other gene-level

tables to attach genomic coordinates, alternative identifiers, and functional annotation.

12.6.7 Interpretation

A reporting sentence: "Gene annotation was retrieved from Ensembl release 110 via biomaRt with `host='jul2023.archive.ensembl.org'` for reproducibility; TP53 was localised to chr17:7{,}668{,}402–7{,}687{,}550, with Entrez ID 7157 for downstream KEGG/Reactome pathway analyses." Always pin the Ensembl release version when attaching annotation to published results.

12.6.8 Practical Tips

- Pin the release host (`host="jul2023.archive.ensembl.org"`) for reproducibility; the default points to the current release, which silently drifts over time.
- `biomaRt` queries can time out for large requests; split into chunks of a few thousand IDs and combine the results.
- For high-throughput annotation (re-running annotations on large gene sets), cache the results locally; re-querying 50{,}000 genes is slow.
- For offline workflows, use `org.Hs.eg.db` (and species-specific equivalents) or `AnnotationHub` for pre-packaged annotation.
- For cross-species orthologue lookup, `getLDS()` queries the homology Mart; this is the standard approach for translating findings between human and mouse studies.
- When the BioMart server is down, fall back to `AnnotationHub` or local packages; never let a transient network failure block downstream analysis.

12.6.9 R Packages Used

`biomaRt` for Ensembl BioMart queries; `org.Hs.eg.db` and species-specific equivalents for offline annotation; `AnnotationHub` for pre-curated annotation packages; `ensembldb` for offline Ensembl annotation in SQLite form; `tximeta` for metadata-aware annotation in RNA-seq workflows.

12.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.6.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR

- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.7 Bulk RNA-seq Differential Expression with DESeq2

12.7.1 Introduction

Bulk RNA-seq asks which genes change in expression between conditions. DESeq2 is the most widely used framework for the analysis: it models counts with a negative binomial distribution, shares dispersion information across genes to stabilise estimates in small samples, and tests each gene's log-fold change for significance with multiple-testing correction. A standard DESeq2 workflow goes from a raw count matrix to a list of differentially expressed genes in a dozen lines of R.

12.7.2 Prerequisites

The reader should understand what a count matrix is (rows = genes, columns = samples), the basic biology of RNA-seq (reads mapped to transcripts, summed per gene), and the idea of a generalised linear model. Familiarity with Bioconductor data structures (`SummarizedExperiment`) is helpful.

12.7.3 Theory

DESeq2 assumes that the read count K_{ij} for gene i in sample j follows a negative binomial distribution

$$K_{ij} \sim \mathrm{NB}(\text{mean} = \mu_{ij},\ \text{dispersion} = \alpha_i),$$

where $\mu_{ij} = s_j q_{ij}$ is the product of a sample-specific size factor s_j (for sequencing depth normalisation) and a gene-specific expected count q_{ij} that depends on the experimental design through a log-linear model $\log_2 q_{ij} = \sum_k x_{jk}\beta_{ik}$.

Three ideas make DESeq2 work well on small-sample studies:

1. **Size factors** are computed by the median-of-ratios method, which is robust to highly expressed genes dominating the normalisation.
2. **Dispersion shrinkage** borrows strength across genes: each gene's dispersion estimate is shrunk toward a mean-dispersion trend fitted across all genes, so that noisy per-gene estimates in small samples do not produce wild inference.
3. **Log-fold-change shrinkage** (via `apeglm` or `ashr`) pulls LFCs of low-count genes toward zero, preventing them from dominating ranked gene lists.

The Wald test compares the estimated coefficient of interest to zero, with standard errors that reflect both the design and the dispersion. The Benjamini-Hochberg procedure controls the false discovery rate across the thousands of genes tested.

12.7.4 Assumptions

- Read counts follow a negative binomial distribution – usually reasonable for bulk mRNA-seq; may fail for very-low-coverage or heavily PCR-amplified libraries.
- The dispersion-mean trend is smooth across genes.
- Replicates within a condition are exchangeable after accounting for modelled covariates.
- No confounding between the biological condition and technical covariates (batch, library prep date). If there is, include the technical covariate in the design.

12.7.5 R Implementation

```
library(DESeq2)
library(apeglm)
library(ggplot2)

counts <- matrix(rnbinom(20000 * 6, mu = 100, size = 10),
                 nrow = 20000, ncol = 6)
rownames(counts) <- paste0("gene", seq_len(20000))
colnames(counts) <- paste0("sample", seq_len(6))
coldata <- data.frame(condition = factor(rep(c("control", "treated"),
 ↪  each = 3)),
                      row.names = colnames(counts))

counts[1:200, 4:6] <- counts[1:200, 4:6] * 3

dds <- DESeqDataSetFromMatrix(countData = counts,
                              colData = coldata,
                              design = ~ condition)

dds <- dds[rowSums(counts(dds) >= 10) >= 3, ]

dds <- DESeq(dds)

res <- lfcShrink(dds, coef = "condition_treated_vs_control",
                 type = "apeglm")

summary(res)
head(res[order(res$padj), ])
```

```r
plotMA(res, ylim = c(-3, 3))
```

The toy simulation above creates 20,000 genes and 6 samples, upregulates the first 200 genes in the treated condition, and runs the standard DESeq2 pipeline: construct the DESeqDataSet, filter very-low-count genes, estimate size factors and dispersion, fit the model, and shrink LFCs for reporting.

12.7.6 Output & Results

The `summary(res)` output reports the number of genes up- and down-regulated at the default alpha (0.1): typically a few hundred in the simulated upregulated set, with a small number of false positives at FDR 10%. The MA plot shows that the shrunken LFCs are compressed toward zero at low mean expression, as desired.

12.7.7 Interpretation

For a manuscript: "Differential expression between treated and control samples was assessed with DESeq2 (v1.44). Size factors were estimated by the median-of-ratios method; dispersions were shared across genes via the DESeq2 trend; log-fold changes were shrunk using `apeglm`. 212 genes were differentially expressed at an FDR of 5% (154 up, 58 down)."

12.7.8 Practical Tips

- Use `vst()` or `rlog()` transformations for downstream visualisation (PCA, heatmaps); use raw counts in DESeq2's own pipeline.
- Filter low-count genes *before* running DESeq (e.g., keep genes with $\geq$ 10 counts in $\geq$ the smallest group size). This improves multiple-testing adjusted p-values without inflating false discoveries.
- Always inspect the PCA of normalised samples before differential expression. If samples cluster by batch rather than by condition, revise the design formula before interpreting results.
- `apeglm` shrinkage is the recommended default for LFC reporting. `ashr` handles contrasts; `normal` is the legacy method.
- Report shrunken LFCs for ranking and visualisation, but use the Wald test p-values for significance calling.

12.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

12.7.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.8 Alignment with BWA and Bowtie

12.8.1 Introduction

BWA-MEM and Bowtie2 are the dominant short-read DNA aligners in modern sequencing pipelines. Both use a Burrows-Wheeler-transformed (BWT) index of the reference genome to achieve rapid exact and approximate matching. The BWT compresses the reference while preserving the ability to query for substrings efficiently — the algorithmic insight that brought genome-scale alignment from hours per sample to minutes. Production alignment runs at the command line; R wrappers (`Rbowtie2`, `BWA` via `Rsubread`) integrate with downstream Bioconductor analyses.

12.8.2 Prerequisites

A working understanding of FASTQ format, reference genomes, and the SAM/BAM alignment format.

12.8.3 Theory

BWA-MEM (Burrows-Wheeler Aligner, Maximal Exact Matches) seeds local alignments by finding maximal exact matches, then extends via banded Smith-Waterman. It is the default aligner for whole-genome DNA sequencing because of its accuracy on long-insert paired-end reads and its handling of indels.

Bowtie2 offers end-to-end alignment (the read must align contiguously) or local alignment (soft-clipping allowed at read ends). It is fast and accurate for short reads and is a common choice for ChIP-seq, ATAC-seq, and small-genome DNA sequencing.

Both write SAM (text) or BAM (binary) output, including alignment scores, mapping quality, mate-pair information, and tags for soft-clipping, mismatches, and other features. For long reads (PacBio, Nanopore), use minimap2 instead — it is designed for the higher error rates and longer alignment lengths.

12.8.4 Assumptions

Short reads (typically below ~300 bp; for longer, switch to minimap2); reference genome appropriate to the species and assembly version of the experiment.

12.8.5 R Implementation

```r
library(Rbowtie2); library(Rsamtools)

# Build index (one-off)
# bowtie2_build(references = "reference.fa", bt2Index = "idx")

# Align
# bowtie2(bt2Index = "idx", samOutput = "aln.sam",
#         seq1 = "reads.fastq", overwrite = TRUE)

# Convert SAM to sorted, indexed BAM
# asBam("aln.sam", destination = "aln_sorted")
# indexBam("aln_sorted.bam")

# Read a short segment
# scanBam("aln_sorted.bam", param = ScanBamParam(which = GRanges("chr1",
 ↪  IRanges(1, 1000))))
```

12.8.6 Output & Results

The primary output is a BAM file containing aligned reads with mapping quality, CIGAR string, and tags. `samtoolsflagstat` (or `Rsamtools::quickBamFlagSummary()`) reports alignment statistics: total reads, reads aligned, properly paired, duplicates, secondary alignments. A 95–98 % alignment rate is typical for high-quality WGS data.

12.8.7 Interpretation

A reporting sentence: "BWA-MEM (v0.7.17) alignment to GRCh38 achieved 97 % overall alignment rate on a 30× whole-genome library, with <1 % ambiguously mapped reads (multi-mappers); after Picard MarkDuplicates, the duplicate rate was 8 %, consistent with a well-optimised PCR-based library prep." Always state the aligner version, reference assembly, and alignment / duplicate rates.

12.8.8 Practical Tips

- Use minimap2 for long reads (PacBio HiFi, Oxford Nanopore); BWA-MEM and Bowtie2 are short-read tools and lose accuracy beyond about 300 bp.
- Report alignment statistics (`flagstat`) after alignment as a routine QC step; declines in alignment rate signal contamination, library-prep problems, or reference mismatch.
- Index BAM files (`samtoolsindex` or `Rsamtools::indexBam()`) for random-access by downstream tools (variant calling, peak calling, browser visualisation).

- Deduplicate after alignment with Picard MarkDuplicates or `samtoolsmark dup`; PCR duplicates inflate variant-calling false positives if not removed.
- For paired-end reads, specify both mate files; the aligner uses insert-size constraints to improve mapping accuracy near repeats.
- For ChIP-seq and ATAC-seq, Bowtie2 with `--very-sensitive` is the standard; for WGS variant calling, BWA-MEM is preferred.

12.8.9 R Packages Used

`Rbowtie2` for `bowtie2()` and `bowtie2_build()`; `Rsubread::align()` for BWA-style alignment; `Rsamtools` for BAM manipulation, indexing, and flagstat-style statistics; `GenomicAlignments` for downstream alignment-aware analysis.

12.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.8.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.9 ChIP-Seq Analysis

12.9.1 Introduction

ChIP-seq (chromatin immunoprecipitation followed by sequencing) localises protein-DNA binding events genome-wide — transcription factor binding sites, histone modifications, chromatin remodellers. The resulting alignment density forms peaks at the binding loci; sharp peaks correspond to point-source binders (TFs), and broad domains correspond to histone-mark territories (H3K27me3, H3K9me3 spanning kilobases). The standard pipeline is alignment, peak calling with MACS2, annotation with ChIPseeker, motif enrichment with HOMER or `motifmatchr`, and differential binding with DiffBind.

12.9.2 Prerequisites

A working understanding of read alignment (BWA, Bowtie2), peak calling, and regulatory genomics including transcription factor binding sites and histone modifications.

12.9.3 Theory

Two peak-calling modes:

- **Narrow peaks** (TF ChIP, transcription start sites): MACS2 defaults are tuned for sharp, point-source binding events.
- **Broad peaks** (H3K27me3, H3K9me3, H3K36me3): use `--broad` to call diffuse domains spanning kilobases.

A matched input control (IgG or total chromatin) is essential for background normalisation; ChIP enrichment is measured as ChIP-vs-input fold enrichment per peak region. Standard QC includes:

- **FRiP** (fraction of reads in peaks): > 1 % indicates successful IP; lower values suggest weak antibody or insufficient enrichment.
- **TSS enrichment**: read coverage at transcription start sites; high values confirm chromatin accessibility was preserved.
- **Cross-correlation**: relative strand cross-correlation should peak at the fragment length, indicating successful IP.

12.9.4 Assumptions

Cross-linking and immunoprecipitation are efficient and specific; the control input matches ChIP depth; duplicates have been removed (essential for ChIP-seq peak calling).

12.9.5 R Implementation

```
library(ChIPseeker)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)

# Example: annotate a toy peak set
set.seed(2026)
peaks <- GRanges(
  seqnames = rep("chr1", 100),
  ranges = IRanges(start = sort(sample(1e7:5e7, 100)), width = 400)
)

anno <- annotatePeak(peaks,
                     TxDb = TxDb.Hsapiens.UCSC.hg19.knownGene,
                     level = "gene",
                     verbose = FALSE)
as.data.frame(anno)[1:5, c("seqnames", "annotation", "distanceToTSS")]

# Feature-type distribution
plotAnnoBar(anno)
```

```
# (Motif-enrichment example would require sequence extraction with
#  BSgenome and motifmatchr::matchMotifs against a JASPAR PWM set.)
```

12.9.6 Output & Results

`annotatePeak()` assigns each peak to its nearest gene and categorises as promoter, intron, intergenic, etc. `plotAnnoBar()` summarises the feature-type distribution; typical TF ChIP-seq shows ~30 % promoter, ~40 % intron, ~25 % distal-intergenic.

12.9.7 Interpretation

A reporting sentence: "MACS2 (with input control, $q < 0.01$) called 18{,}230 high-confidence STAT1 ChIP peaks in IFN-γ-stimulated cells; 42 % were promoter-proximal, 35 % intronic, 18 % distal-intergenic. De novo motif analysis (HOMER) recovered the canonical STAT1 GAS motif (TTCN3GAA) at $p < 10^{-100}$, confirming antibody specificity. FRiP was 12 %, TSS enrichment 16, both well within QC standards." Always report FRiP, TSS enrichment, and motif validation.

12.9.8 Practical Tips

- Always use a matched input control (IgG, sonicated input, or no-antibody mock); without it, MACS2 falls back to generic background assumptions and inflates false positives.
- Check FRiP per sample; values below 1 % indicate weak IP, incorrect peak-calling mode, or sample failure.
- For histone marks (especially H3K27me3, H3K36me3, H3K9me3), use `--broad`; sharp peak callers miss diffuse domains entirely.
- Validate antibody specificity with de novo motif enrichment (HOMER's `findMotifsGenome.pl` or MEME-ChIP); recovering the expected motif is the gold standard for ChIP success.
- For differential binding across conditions, use DiffBind which builds consensus peak sets and runs DESeq2 or edgeR on peak-level read counts.
- For CUT&RUN or CUT&Tag (the modern higher-resolution alternatives), the same downstream tools apply; specialised peak callers (SEACR) handle the lower-background data more accurately.

12.9.9 R Packages Used

`ChIPseeker` for peak annotation and visualisation; `DiffBind` for differential-binding analysis; `motifmatchr` for motif enrichment with PWM databases; `chromVAR` for chromatin-accessibility-style motif activity scores; `ChIPQC` for quality control on ChIP-seq experiments.

12.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.9.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.10 Copy Number Variation Analysis

12.10.1 Introduction

Copy number variations (CNVs) are large-scale gains or losses of genomic regions, critical in oncology (tumour aneuploidy), developmental disorders (trisomies, deletions), and population genetics. WGS, exome, or array-based readouts feed segmentation algorithms that partition the genome into constant-copy-number segments.

12.10.2 Prerequisites

Coverage or intensity data (read depth, log R ratio); genome windowing.

12.10.3 Theory

Read-depth CNV calling: normalised read counts in windows scale linearly with copy number. The log-ratio series is smoothed and segmented via **Circular Binary Segmentation (CBS)** (DNAcopy), HMMs (`HMMcopy`), or Bayesian segmentation.

For tumour sequencing, methods like `ASCAT` and `FACETS` jointly estimate purity, ploidy, and allele-specific copy number from B-allele frequencies.

12.10.4 Assumptions

Coverage uniformity (controlled by GC / mappability correction); matched normal for tumour calling (ideal); segments have consistent copy number within.

12.10.5 R Implementation

```r
library(DNAcopy)

set.seed(2026)
n_probes <- 1000
chrom <- rep(1:5, each = n_probes / 5)
maploc <- rep(seq(1, 1e7, length.out = n_probes / 5), 5)

# Simulated log ratios with two CNV segments
logR <- rnorm(n_probes, 0, 0.2)
logR[201:300] <- logR[201:300] + 0.8    # duplication on chr2
logR[601:680] <- logR[601:680] - 1.0    # deletion   on chr4

cna <- CNA(genomdat = logR, chrom = chrom, maploc = maploc,
           data.type = "logratio", sampleid = "tumor1")
smoothed <- smooth.CNA(cna)
seg <- segment(smoothed, verbose = 0)
seg$output                     # segment means and boundaries

plot(seg)
```

12.10.6 Output & Results

A segment table with chromosome, start, end, number of probes, and mean log ratio; the simulated duplication and deletion are recovered as distinct segments.

12.10.7 Interpretation

"CNV segmentation of the tumour identified gain on 17q (log2 ratio 0.85) and loss on 13q (log2 ratio -1.1); the 13q deletion spans RB1, consistent with the retinoblastoma diagnosis."

12.10.8 Practical Tips

- GC bias correction (`QDNAseq`) is essential for low-coverage sequencing; uncorrected data produces spurious segments.
- For tumour studies, always use a matched normal to control germline CNVs and sequencing artefacts.
- Integer copy-number calls require ploidy and purity estimates (ASCAT, FACETS); log ratios alone give only relative copy number.
- Report CNV frequencies across cohorts with GISTIC2, which controls for background aneuploidy.
- For rare-CNV studies, focus on large events ($>$ 100 kb); small events are dominated by noise in array data.

12.10.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.10.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.11 Counting Reads with featureCounts

12.11.1 Introduction

After read alignment, the next step in most RNA-seq pipelines is summarising aligned reads into per-gene (or per-feature) count matrices for differential-expression analysis. `featureCounts` (in `Rsubread`) and HTSeq-count are the two standard tools; both consume BAM files plus a GTF annotation and produce a gene-by-sample count matrix. The choices that matter — strand-specificity, multi-mapper handling, paired-end mode, and meta-feature aggregation — directly affect downstream conclusions and must match the experimental library prep.

12.11.2 Prerequisites

A working understanding of BAM alignment files, GTF/GFF annotation formats, and the difference between transcript-level and gene-level expression quantification.

12.11.3 Theory

Each aligned read is assigned to a feature (gene, exon, or other annotated region) it overlaps. Key configuration choices:

- **Strand-specificity**: `strandSpecific=0` (unstranded), `=1` (forward-stranded, dUTP), `=2` (reverse-stranded, TruSeq SR). Wrong setting halves the assigned counts.
- **Multi-mapper handling**: usually counted once at the primary alignment; fractional counting (`fraction=TRUE`) splits credit across all alignments of a multi-mapper.
- **Paired-end**: count fragments rather than reads; both mates must align consistently.

- **Meta-feature**: aggregate exon-level counts into gene-level by `gene_id`; alternatively, count at the exon level for splicing analyses.

For transcript-level counts, pseudo-alignment tools (Salmon, Kallisto) skip the BAM-producing alignment step and provide isoform-level abundance estimates directly.

12.11.4 Assumptions

The GTF annotation matches the genome reference used for alignment; the strand-specificity setting matches the library prep; aligned BAMs are coordinate-sorted (or sortable by `featureCounts`).

12.11.5 R Implementation

```
library(Rsubread)

# Build counts from BAM + GTF
# counts <- featureCounts(files = c("s1.bam", "s2.bam"),
#                         annot.ext = "annotation.gtf",
#                         isGTFAnnotationFile = TRUE,
#                         GTF.featureType = "exon",
#                         GTF.attrType = "gene_id",
#                         strandSpecific = 2,
#                         isPairedEnd = TRUE)

# counts$counts: gene x sample matrix
```

12.11.6 Output & Results

`featureCounts()` returns a list with `$counts` (gene-by-sample matrix), `$annotation` (per-gene metadata including length used for FPKM/TPM normalisation), and `$stat` (per-sample summary of assigned vs ambiguous vs unassigned reads).

12.11.7 Interpretation

A reporting sentence: "featureCounts (Rsubread v2.16) assigned 82 % of mapped reads to GENCODE v44 gene features (`strandSpecific=2`, `isPairedEnd=TRUE`); 8 % were ambiguous due to multi-gene overlap and 10 % were unassigned to any feature. The high assignment rate confirms the strand setting matches the dUTP library prep." Always quote the assignment rate; sub-50 % rates signal serious problems (wrong strand, annotation mismatch).

12.11.8 Practical Tips

- Report `$stat` summaries alongside the count matrix; low assignment rates (below 60 % for typical RNA-seq) signal alignment, annotation, or strand-setting problems.
- Strand-specificity must match library protocol exactly; check with `infer_experiment.py` from RSeQC or visual inspection in IGV when uncertain.
- Use `primaryOnly=TRUE` for ChIP-seq and ATAC-seq, where multi-mappers are typically excluded.
- For transcript-level (isoform) counts, switch to Salmon, Kallisto, or RSEM; gene-level counting collapses isoform-resolved information.
- Bundle counts with metadata in a `SummarizedExperiment` object for downstream Bioconductor workflows (DESeq2, edgeR, limma all accept `SummarizedExperiment`).
- Include feature length in the output and use it for normalisation when reporting TPM/FPKM; raw counts alone do not control for gene length differences.

12.11.9 R Packages Used

`Rsubread::featureCounts()` for the canonical implementation; `GenomicAlignments::summarizeOverlaps()` for an alternative Bioconductor-style approach; `tximport` for importing transcript-level estimates from Salmon/Kallisto into `SummarizedExperiment`; `tximeta` for metadata-aware imports.

12.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.11.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.12 Drug-Target Interaction Mining

12.12.1 Introduction

Drug-target interaction data – which compounds bind which proteins at what affinity – underpins target discovery and drug repositioning. Public databases (ChEMBL, DrugBank, PubChem, BindingDB, PharmGKB) archive millions of

bioassay measurements; programmatic access lets users assemble custom interaction tables.

12.12.2 Prerequisites

Familiarity with compound identifiers (SMILES, InChI, ChEMBL ID) and protein identifiers (UniProt).

12.12.3 Theory

Key bioassay quantities: - **IC50** – inhibitor concentration halving a biological response. - **Ki** – equilibrium inhibition constant. - **Kd** – dissociation constant. - **EC50** – effect concentration (agonist activity).

Lower = more potent. Values are typically reported in nM or uM; low-nM and sub-nM bindings are considered high-affinity drug-like hits.

12.12.4 Assumptions

Assays are comparable across studies (unreliable in practice – format, conditions, species differ); target and compound annotations are correct.

12.12.5 R Implementation

```
library(httr); library(jsonlite); library(dplyr)

# Query ChEMBL's REST API for bioactivities of a target (ERBB2 / HER2 =
 ↪  CHEMBL1824)
# This illustrative snippet is internet-free pseudo-code:
chembl_url <-
 ↪  paste0("https://www.ebi.ac.uk/chembl/api/data/activity.json",
                    "?target_chembl_id=CHEMBL1824&limit=100")
# res <- fromJSON(content(GET(chembl_url), "text"))
# activities <- res$activities

# Instead, simulate a small bioactivity table for illustration
set.seed(2026)
df <- data.frame(
  compound = paste0("CMP", 1:200),
  target   = sample(c("HER2", "EGFR", "MET"), 200, replace = TRUE),
  type     = sample(c("IC50", "Ki", "Kd"), 200, replace = TRUE),
  value_nM = round(10^runif(200, 0, 4))
)

# Potent binders (< 100 nM) per target
df %>%
```

```
  filter(value_nM < 100) %>%
  count(target, type, sort = TRUE)

# Rank compounds by median potency across multiple assays
df %>%
  group_by(compound, target) %>%
  summarise(median_pot = median(value_nM), n = n(), .groups = "drop") %>%
  arrange(median_pot) %>%
  head(5)
```

12.12.6 Output & Results

A ranked table of potent drug-target interactions per target and per compound. In practice this replaces the simulated data with a ChEMBL pull of $> 10{,}000$ measurements.

12.12.7 Interpretation

"ChEMBL mining identified 26 HER2 inhibitors with IC50 < 10 nM across independent assays; the most potent (CMP_X, median IC50 $= 2$ nM) was selected for in-silico docking."

12.12.8 Practical Tips

- Aggregate across assays using median or geometric mean to smooth inter-lab variance.
- Convert IC50 to pIC50 ($-\log_{10}$) for symmetric, Gaussian-like distributions.
- Filter by assay confidence score when using ChEMBL (confidence ≥ 7).
- Use `BindingDB` for affinity-focused data; `DrugBank` for approved-drug information.
- Cross-reference compounds via InChIKey to merge records across databases (SMILES is not unique).

12.12.9 Reporting

A reproducible drug-target-interaction analysis names the database snapshot, the date of access, the activity types retained, and the confidence or quality filter applied to the raw assay records. State whether replicate measurements on the same compound-target pair were aggregated by median, mean of pIC50, or geometric mean, since these choices materially shift downstream rankings. When potency thresholds such as the conventional sub-100-nanomolar cutoff are used to define hits, justify the threshold in terms of the assay distribution rather than treating it as a universal constant, and report how compounds with mixed agonist or antagonist annotations were resolved before the ranking step.

12.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.12.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.13 Differential Expression with edgeR

12.13.1 Introduction

edgeR is one of the three dominant RNA-seq differential-expression frameworks (alongside DESeq2 and limma-voom). It fits a negative-binomial generalised linear model to each gene's counts, with empirical-Bayes shrinkage borrowing information across genes to stabilise per-gene dispersion estimates. The combination of NB likelihood and shrinkage gives edgeR its characteristic strength on small samples and its competitive accuracy on standard RNA-seq experiments.

12.13.2 Prerequisites

A working understanding of negative-binomial regression, RNA-seq normalisation (TMM), generalised linear models with offsets, and the empirical-Bayes shrinkage approach.

12.13.3 Theory

Per gene i and sample j:

$$Y_{ij} \sim \mathrm{NB}(\mu_{ij}, \phi_i), \qquad \log \mu_{ij} = \log S_j + x_j^\top \beta_i,$$

with S_j the TMM-normalised library size offset and ϕ_i the gene-wise dispersion. Empirical-Bayes shrinkage (`estimateDisp`) pulls per-gene dispersions toward a common trend across genes, stabilising estimation.

Three test choices:

- **exactTest**: NB-exact test for two-group comparisons; rarely used today.

- **glmQLFTest**: quasi-likelihood F-test; the recommended default for nearly all designs because it accounts for uncertainty in dispersion estimation.
- **glmLRT**: classical likelihood-ratio test; more aggressive but anti-conservative when sample sizes are very small.

12.13.4 Assumptions

Counts approximately negative-binomial conditional on covariates; the design matrix encodes the biological and technical factors correctly; sufficient sample size for stable dispersion estimation (typically ≥ 3 per group).

12.13.5 R Implementation

```r
library(edgeR)

set.seed(2026)
counts <- matrix(rpois(20000 * 12, lambda = 30), 20000, 12)
group <- factor(rep(c("A", "B"), each = 6))

y <- DGEList(counts = counts, group = group)
y <- calcNormFactors(y)

design <- model.matrix(~ group)
y <- estimateDisp(y, design)

fit <- glmQLFit(y, design)
qlf <- glmQLFTest(fit, coef = 2)

topTags(qlf)
summary(decideTests(qlf))
```

12.13.6 Output & Results

`topTags()` returns the top differentially-expressed genes ordered by FDR-adjusted p-value, with $\log_2$ fold change, log-CPM, F-statistic, and BH-adjusted p-value. `decideTests()` summarises significance calls per direction (up / down / not significant) for cross-method comparison.

12.13.7 Interpretation

A reporting sentence: "edgeR's quasi-likelihood F-test identified 342 differentially expressed genes at FDR < 0.05 between conditions A and B (n $= 6$ per group), with 180 upregulated and 162 downregulated in condition B; the median absolute $\log_2$ fold change among significant genes was 1.4. Filtering with `filte`

`rByExpr` retained 14{,}500 of 20{,}000 genes for testing." Always state the FDR threshold, sample sizes, and pre-filtering decisions.

12.13.8 Practical Tips

- Use `glmQLFTest()` as the default for nearly all designs; it has better Type I error control than `glmLRT()` at small sample sizes.
- Filter low-count genes with `filterByExpr()` before dispersion estimation; testing 20{,}000 mostly-zero genes wastes power and produces unstable shrinkage.
- For paired or blocked designs, include the blocking factor in the `design` matrix; the QLF test handles the corresponding contrast correctly.
- Use `glmTreat()` for fold-change-thresholded inference (testing $|\text{LFC}| > a$ chosen cutoff rather than $\text{LFC} \neq 0$); this is more biologically meaningful than testing strict zero.
- Visualise with MA plot, volcano plot, and a heatmap of top genes; `Glimma::glimmaMA()` provides interactive versions for exploration.
- Compare edgeR results with DESeq2 and limma-voom on the same data; concordance across methods is reassuring, while disagreement signals model-sensitive genes that warrant investigation.

12.13.9 R Packages Used

`edgeR` for `DGEList`, `calcNormFactors`, `estimateDisp`, `glmQLFit`, `glmQLFTest`, and the canonical NB workflow; `limma` for the voom alternative; `DESeq2` for the third major DE method; `Glimma` for interactive MA and volcano plots; `EnhancedVolcano` for publication-ready volcano plots.

12.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.13.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.14 FASTQ Quality Control

12.14.1 Introduction

Every high-throughput sequencing analysis starts with FASTQ quality control. Before alignment, before quantification, before any downstream interpretation, you should inspect per-base quality, adapter contamination, GC content, and duplicate rate to decide on trimming and filtering strategies. The cost of skipping this step is high: low-quality data quietly degrade alignment accuracy, inflate variant-calling false positives, and bias differential-expression conclusions in ways that are hard to detect retrospectively.

12.14.2 Prerequisites

A working understanding of the FASTQ format, Phred quality scores, and basic Illumina sequencing biology.

12.14.3 Theory

The Phred quality score is $Q = -10 \log_{10} P(\text{error})$, so Q30 means 1 in 1000 probability of base-call error and Q40 means 1 in 10{,}000. Modern Illumina runs aim for Q30 or higher across most of each read, with quality typically dropping toward the 3' end.

Standard FastQC-style metrics:

- **Per-base sequence quality**: boxplot of Phred scores at each position; declines toward 3' are normal but should not drop below Q20.
- **Per-sequence quality**: histogram of mean read quality; bi-modal histograms suggest mixed-quality libraries.
- **GC content**: should match expected genomic GC; deviations indicate contamination or biased library prep.
- **Adapter contamination**: presence of known adapter sequences at 3' ends.
- **Overrepresented sequences**: top recurring sequences often pinpoint adapters or contamination.

12.14.4 Assumptions

Modern Illumina FASTQ files with Sanger-encoded Phred (Phred+33). Older Illumina encodings (Phred+64) require explicit specification.

12.14.5 R Implementation

```
library(ShortRead); library(Rqc)

# Assume demo FASTQ file path
```

```r
fq_file <- system.file("extdata", "ex1.fq", package = "ShortRead")

# Basic inspection
reads <- readFastq(fq_file)
length(reads)
head(width(reads))
mean(as.numeric(encoding(quality(reads))))   # mean Phred base

# Rqc for systematic QC
qc <- rqcQA(fq_file, workers = 1)
rqcReport(qc)
```

12.14.6 Output & Results

`Rqc::rqcQA()` produces an interactive HTML report covering per-position quality, GC content, duplicate rate, length distribution, and adapter flags. `ShortRead::readFastq()` provides programmatic access for custom diagnostics.

12.14.7 Interpretation

A reporting sentence: "FastQC-style QC on the 12 RNA-seq samples showed mean per-base Phred ranging from 35 to 38 across the read length, adapter contamination below 1 %, and GC content centred at 49 % (within the expected range for human transcriptome). After fastp trimming at Q15 with adapter removal, approximately 95 % of bases were retained per sample." Always report pre- and post-trim QC.

12.14.8 Practical Tips

- Use `fastp` or `trim_galore` at the command line for production QC and trimming; R tools (`Rqc`, `ShortRead`) are useful for exploration but are slower than the dedicated command-line utilities on large datasets.
- Report pre- and post-trim QC; the QC improvement validates the trimming step.
- GC content far from the expected genomic value suggests contamination, library prep biases, or species mismatch; investigate before alignment.
- Overrepresented sequences often pinpoint adapter sequences, rRNA contamination, or PCR duplicates of single dominant transcripts.
- For multiple samples, MultiQC aggregates per-sample reports into a single dashboard; this is essential for reviewing dozens or hundreds of samples consistently.
- Phred Q20 is the conventional threshold for "good" bases; trimming at Q15 retains more data while removing the worst calls.

12.14.9 R Packages Used

`ShortRead` for general FASTQ manipulation in R; `Rqc` for systematic QC reports; `Rfastp` as a wrapper around fastp for fast trimming and QC; `ngsReports` for parsing and aggregating MultiQC and FastQC outputs into R objects.

12.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.14.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.15 Gene Ontology Enrichment

12.15.1 Introduction

Gene Ontology (GO) enrichment asks whether a list of differentially-expressed genes is disproportionately annotated to specific GO terms — biological processes, molecular functions, or cellular components. It is the most common first-pass functional interpretation of an RNA-seq, proteomics, or single-cell DE list and is the entry point to pathway-level analysis. The dominant method is over-representation analysis (ORA) using Fisher's exact or hypergeometric tests; gene-set enrichment analysis (GSEA) extends to ranked gene lists without requiring a significance threshold.

12.15.2 Prerequisites

A working understanding of differential-expression analysis, GO annotations and their three branches (biological process, molecular function, cellular component), and basic Fisher's exact test framework.

12.15.3 Theory

ORA constructs a 2×2 contingency table for each GO term:

	DE	not DE
In term	a	b

	DE	not DE
Not in term	c	d

Fisher's exact test (or the hypergeometric distribution) tests whether more DE genes fall in the term than expected under random sampling from the universe. The universe is the set of all genes that *could* have been tested — typically the genes that passed pre-filtering — not the entire genome.

Multiple testing across thousands of GO terms is controlled via Benjamini-Hochberg FDR; nested GO term hierarchies introduce correlated tests, but BH adjustment remains conservative under dependence.

12.15.4 Assumptions

Genes are sampled uniformly from the universe given non-DE status; GO annotations are correct (recently updated); the universe matches the tested set (testing all genes from the genome inflates apparent enrichment when most are not in the tested set).

12.15.5 R Implementation

```
library(clusterProfiler); library(org.Hs.eg.db)

# Example with a small simulated DE list
set.seed(2026)
all_genes <- keys(org.Hs.eg.db, keytype = "SYMBOL")
de_genes  <- sample(all_genes, 200)

ego <- enrichGO(gene = de_genes,
                universe = all_genes,
                OrgDb = org.Hs.eg.db,
                keyType = "SYMBOL",
                ont = "BP",
                pAdjustMethod = "BH",
                pvalueCutoff = 0.05,
                qvalueCutoff = 0.2)

head(as.data.frame(ego))[, c("ID", "Description", "p.adjust", "Count")]
dotplot(ego, showCategory = 15)
```

12.15.6 Output & Results

`enrichGO()` returns a ranked table of GO terms with BH-adjusted p-values, gene counts, and the contributing gene symbols. `dotplot()` visualises the top hits as a horizontal dot plot with gene-count size and adjusted-p-value colour.

12.15.7 Interpretation

A reporting sentence: "GO over-representation analysis (BH-FDR < 0.05) identified 28 biological-process terms among the 200 DE genes; the top terms — `r esponsetostimulus` (45 genes), `immuneresponse` (32), `cellcycleregulation` (28) — reflect the experimental manipulation. `simplify()` reduced the redundant hierarchy to 12 representative terms." Always state the universe, the multiple-testing correction, and any term-redundancy reduction.

12.15.8 Practical Tips

- Define the universe correctly — always the set of genes that passed pre-filtering and were tested for DE, never the whole genome; using the latter inflates apparent enrichment.
- Use `simplify()` on the `enrichResult` object to collapse redundant nested GO terms; the unsimplified output often contains many parent-child duplications.
- Direction-agnostic ORA loses signal; for biology-aware interpretation, run up- and down-regulated lists separately.
- For ranked gene lists with continuous statistics, prefer GSEA (`gseGO()`) over ORA; GSEA does not require an arbitrary significance threshold.
- `goseq` adjusts for gene-length bias in RNA-seq (longer genes have more power for DE detection); use it when length bias is a concern.
- Pair GO with KEGG (`enrichKEGG`) and Reactome (`enrichPathway`) for complementary functional coverage.

12.15.9 R Packages Used

`clusterProfiler::enrichGO()`, `gseGO()`, `simplify()` for canonical GO enrichment; `org.Hs.eg.db` and species-specific packages for annotation; `topGO` and `go seq` for alternative implementations; `enrichplot` for `dotplot()`, `cnetplot()`, and `emapplot()` visualisations of enrichment results.

12.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.15.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.16 GSEA Preranked Analysis

12.16.1 Introduction

Gene Set Enrichment Analysis (GSEA) tests whether a gene set is concentrated at the top or bottom of a ranked gene list (e.g., by log fold change or signed p-value). Unlike over-representation analysis, it uses the full ranking and is more sensitive to coordinated modest changes within a pathway.

12.16.2 Prerequisites

Ranked gene statistic (LFC or signed -log10 p); a curated gene-set database (MSigDB, Reactome, KEGG).

12.16.3 Theory

GSEA walks down the ranked list, incrementing a running-sum statistic when encountering a set member (weighted by the rank statistic) and decrementing otherwise. The **enrichment score (ES)** is the maximum deviation from zero; a permutation-based null distribution yields p-values. `fgsea` uses a fast adaptive multi-level permutation scheme.

Normalised ES (NES) adjusts for gene-set size and allows cross-set comparison.

12.16.4 Assumptions

The ranking reflects biological signal; gene sets are well-curated; permutation null captures the null distribution of ES.

12.16.5 R Implementation

```
library(fgsea); library(msigdbr)

set.seed(2026)
n_genes <- 5000
genes <- paste0("g", 1:n_genes)
stats <- rnorm(n_genes); names(stats) <- genes

# Inject a pathway signal
pathway_A <- sample(genes, 30); stats[pathway_A] <- stats[pathway_A] +
 ↪  1.5
pathway_B <- sample(genes, 20); stats[pathway_B] <- stats[pathway_B] -
 ↪  1.2

pathways <- list(pathway_A = pathway_A, pathway_B = pathway_B,
                 pathway_C = sample(genes, 40))
```

```
fg <- fgsea(pathways = pathways, stats = sort(stats, decreasing = TRUE),
            minSize = 10, maxSize = 500)
fg[, .(pathway, NES, pval, padj)]
plotEnrichment(pathways$pathway_A, sort(stats, decreasing = TRUE))
```

12.16.6 Output & Results

A table with NES, p-value, and adjusted p-value per pathway. The enrichment plot for pathway_A shows a strong positive leading edge.

12.16.7 Interpretation

"GSEA against MSigDB Hallmark revealed positive enrichment for inflammatory response (NES = 1.8, BH-FDR < 0.01) and negative enrichment for oxidative phosphorylation (NES = -1.5)."

12.16.8 Practical Tips

- Use the `stat` column from DESeq2 or `t` from limma as the ranking; signed -log10(p-value) is an alternative.
- Remove NAs from the ranking; fgsea is sensitive to ties and NAs.
- Report the **leading-edge subset** for biologically driven interpretation.
- MSigDB has Hallmark (50 curated), C2 (curated pathways), C5 (GO), C7 (immune); match collection to biological question.
- GSEA tests concentrated enrichment, not direction-specific effects; combine with ORA for complementary evidence.

12.16.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.16.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.17 GSVA Single-Sample Enrichment

12.17.1 Introduction

GSVA (Gene Set Variation Analysis) transforms a gene-by-sample expression matrix into a pathway-by-sample matrix by assigning each sample a per-pathway enrichment score. The transformation enables pathways to be treated as features in any downstream analysis — Cox regression for survival, clustering, classification, or differential analysis at the pathway level. Where over-representation analysis (GO, KEGG over-representation) tells you which pathways are enriched in a hit list, GSVA tells you how each individual sample stands relative to a panel of pathways.

12.17.2 Prerequisites

A working understanding of gene-set enrichment, log-normalised expression matrices, and the difference between sample-level and contrast-level enrichment.

12.17.3 Theory

For each sample, GSVA computes an empirical CDF of expression values across genes (rank-based) and derives a Kolmogorov-Smirnov-like statistic for each gene set. The result is a per-sample, per-pathway score, typically on a roughly $[-1, 1]$ scale. ssGSEA (single-sample GSEA) is a related method differing primarily in the weighting scheme; both are implemented in the `GSVA` package.

The transformed pathway-by-sample matrix can be analysed with any standard tool: limma for differential pathway expression, Cox regression for survival, hierarchical clustering, supervised classification.

12.17.4 Assumptions

Expression is consistently normalised across samples; gene sets are accurately curated and relevant to the biology; scores are interpretable as relative within a study but not on an absolute scale across studies.

12.17.5 R Implementation

```
library(GSVA)

set.seed(2026)
n_genes <- 500; n_samples <- 20
expr <- matrix(rnorm(n_genes * n_samples), n_genes, n_samples)
rownames(expr) <- paste0("g", 1:n_genes)

pathways <- list(
  path_A = paste0("g", sample(1:n_genes, 30)),
```

```
  path_B = paste0("g", sample(1:n_genes, 40)),
  path_C = paste0("g", sample(1:n_genes, 25))
)

gsva_scores <- gsva(expr, pathways, method = "gsva", kcdf = "Gaussian",
                    verbose = FALSE)
dim(gsva_scores)        # pathways x samples
round(gsva_scores[, 1:4], 2)
```

12.17.6 Output & Results

`gsva()` returns a pathway-by-sample matrix of enrichment scores. Each column is one sample's pathway profile; high scores indicate the pathway's genes are coordinately up-regulated in that sample relative to others, low scores indicate the opposite.

12.17.7 Interpretation

A reporting sentence: "GSVA pathway scores (Hallmark v7.5 gene sets, log-CPM input, Gaussian kernel) served as features in a Cox regression of overall survival across 200 patients; the inflammatory-response pathway emerged as the strongest independent predictor (HR 1.7 per SD increase, 95 % CI 1.2 to 2.4, $p = 0.002$) after adjustment for age, stage, and treatment arm." Always state the gene-set source, the kernel, and the input data.

12.17.8 Practical Tips

- Use `kcdf="Gaussian"` for log-CPM, `kcdf="Poisson"` for raw counts; the wrong setting biases scores systematically.
- Downstream differential-pathway analysis works with a standard limma fit on the GSVA score matrix with the same design used for gene-level analysis.
- For single-cell expression, per-cell GSVA is very noisy; aggregate cells into pseudobulk first, or use signed-rank alternatives (`AUCell`, `UCell`) designed for the single-cell context.
- Always compute GSVA on a single, consistently normalised expression matrix; mixing normalisations across the input matrix produces invalid scores.
- Scores are relative within a study; do not compare raw GSVA scores across independent datasets without re-running on the combined matrix.
- For pathway exploration, the MSigDB Hallmark, KEGG, and Reactome gene sets (via `msigdbr::msigdbr()`) are the standard starting libraries.

12.17.9 R Packages Used

`GSVA::gsva()` for the canonical GSVA and ssGSEA implementation; `msigdbr` for MSigDB gene-set retrieval; `AUCell` and `UCell` for single-cell-friendly enrichment alternatives; `singscore` for an alternative rank-based single-sample method; `escape` for a `Seurat`-integrated GSVA workflow.

12.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.17.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.18 Heatmaps for RNA-seq

12.18.1 Introduction

Heatmaps compactly display an expression matrix as a colour grid with rows (genes) and columns (samples) optionally clustered hierarchically. For RNA-seq they are the standard visualisation for a differential-expression signature, a curated marker panel, or a pathway-of-interest gene set: colour intensity encodes scaled expression, dendrograms reveal structure, and annotation bars link sample columns to experimental metadata. Done well, a heatmap communicates more about the biological structure of an experiment than dozens of pages of statistical tables.

12.18.2 Prerequisites

A working understanding of log-CPM or variance-stabilised counts, hierarchical clustering, and the role of row scaling (z-scoring) in making heatmaps interpretable.

12.18.3 Theory

Raw counts are unsuitable for heatmaps because one strongly expressed gene dominates the colour range and obscures the pattern in lower-expression genes. The standard pre-processing is:

1. **Log-transform** counts (or use VST/rlog from DESeq2).
2. **Row z-score** each gene to unit variance and zero mean across samples; this scales each gene's display range comparably.
3. **Cluster rows and columns** by hierarchical clustering, typically Euclidean distance with Ward or complete linkage.

Annotation bars (sample group, batch, time point) overlay metadata on the column dendrogram, making it immediately visible whether sample clustering corresponds to biology or to technical artefacts.

12.18.4 Assumptions

Row scaling assumes comparable behaviour of genes (one z-score unit means the same thing across genes); column clustering assumes samples are comparable after normalisation; the chosen distance metric and linkage match the cluster structure.

12.18.5 R Implementation

```
library(pheatmap)

set.seed(2026)
n_genes <- 50; n_samples <- 20
mat <- matrix(rnorm(n_genes * n_samples), n_genes, n_samples)
rownames(mat) <- paste0("gene_", 1:n_genes)
colnames(mat) <- paste0("S", 1:n_samples)

# Introduce a DE pattern
group <- rep(c("ctrl", "trt"), each = 10)
mat[1:10, group == "trt"] <- mat[1:10, group == "trt"] + 3

annot <- data.frame(group = factor(group))
rownames(annot) <- colnames(mat)

pheatmap(mat, scale = "row",
         clustering_distance_rows = "euclidean",
         clustering_method = "ward.D2",
         annotation_col = annot,
         color = colorRampPalette(c("#2A9D8F", "white", "#E76F51"))(50),
         main = "Top DE genes (row-scaled z-scores)")
```

12.18.6 Output & Results

`pheatmap()` produces a coloured grid with row and column dendrograms, an annotation strip for the `group` factor, and a colour legend. The two sample clusters separate the control and treatment groups based on the 10 simulated

DE genes; the row dendrogram groups the DE genes into a coherent block distinct from the noise rows.

12.18.7 Interpretation

A reporting sentence (figure caption): "Row-scaled heatmap of the top 50 differentially-expressed genes (z-scores per row, blue-white-red diverging colour scale, Ward linkage on Euclidean distance); samples cluster into two distinct groups corresponding to the experimental conditions, and the top 10 row-cluster shows coherent up-regulation in the treatment group." Always describe the scaling, distance metric, and linkage method.

12.18.8 Practical Tips

- Always row-scale (z-score) for DE heatmaps; otherwise high-expression genes swamp the colour range and the pattern in lower-expression genes is invisible.
- Use `ComplexHeatmap::Heatmap()` for richer annotations, row/column splits by cluster, and multi-panel composition; `pheatmap` is simpler for one-off plots.
- Truncate extreme values (e.g., cap z-scores at ± 3) to prevent one outlier dominating the colour scale.
- Annotate samples with batch, condition, and time-point information in column annotations; the visual separation should match the biology, not the technical structure.
- Use variance-stabilised counts (DESeq2 `vst()` or `rlog()`) rather than raw log-CPM for more robust heatmaps when the dataset has wide expression range.
- Save the figure at the final publication size (`ggsave()` if using `ggplot`, `pdf()`/`png()` for base plots); shrinking heatmaps in a word processor produces unreadable cell labels.

12.18.9 R Packages Used

`pheatmap` for quick standard heatmaps; `ComplexHeatmap` for advanced layouts including multiple annotations, row/column splits, and composition with other panels; `DESeq2::vst()` and `rlog()` for variance-stabilising input transformations; `RColorBrewer` and `viridis` for colour palettes.

12.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.18.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.19 Differential Expression with limma-voom

12.19.1 Introduction

limma-voom is one of the three dominant differential-expression methods for RNA-seq (alongside DESeq2 and edgeR). The voom step transforms raw counts onto the log-CPM scale and computes per-observation precision weights that account for the mean-variance trend characteristic of count data. The result is suitable input to limma's full linear-modelling framework, which adds the empirical-Bayes shrinkage of variance estimates that drives limma's power on small samples and complex designs.

12.19.2 Prerequisites

A working understanding of linear models, RNA-seq normalisation (TMM via edgeR), the mean-variance relationship of count data, and the empirical-Bayes shrinkage approach.

12.19.3 Theory

`voom()` transforms raw counts via $\log_2(\text{CPM} + 0.5)$ and fits a smooth mean-variance trend across genes. The fitted trend produces a precision weight for each observation; downstream `lmFit()` regression uses these weights so that low-expression genes (with high relative variance) contribute less to the joint estimation. The empirical-Bayes step (`eBayes()`) shrinks per-gene variance estimates toward the global trend, yielding moderated t-statistics that have better small-sample properties than ordinary t-tests.

The full limma-voom pipeline supports arbitrarily complex designs (interactions, blocking, batch adjustment) through standard `model.matrix` notation, and the contrast machinery handles multi-condition comparisons cleanly via `contrasts.fit()`.

12.19.4 Assumptions

Counts have been TMM-normalised (via `calcNormFactors` from edgeR); the chosen design matrix correctly encodes the biological and technical factors; the mean-variance trend is approximately monotone (true for typical bulk RNA-seq).

12.19.5 R Implementation

```r
library(limma); library(edgeR)

set.seed(2026)
counts <- matrix(rpois(20000 * 12, lambda = 30), 20000, 12)
group <- factor(rep(c("A", "B"), each = 6))

dge <- DGEList(counts = counts); dge <- calcNormFactors(dge)
design <- model.matrix(~ group)

v <- voom(dge, design, plot = FALSE)
fit <- lmFit(v, design)
fit <- eBayes(fit)

topTable(fit, coef = 2, n = 10)
summary(decideTests(fit))
```

12.19.6 Output & Results

`topTable()` returns the top differentially-expressed genes with $\log_2$ fold change, average expression, moderated t-statistic, raw p-value, and Benjamini-Hochberg adjusted p-value. `decideTests()` summarises significance calls per contrast, useful for cross-method comparison.

12.19.7 Interpretation

A reporting sentence: "limma-voom identified 380 genes at FDR < 0.05 between conditions A and B (n $= 6$ per group), with mean fold change among significant genes of 1.7 (range 0.4 to 3.8); the corresponding edgeR analysis on the same data identified 342 genes, with 320 genes (84 %) shared between methods, indicating substantial concordance." Always state the FDR threshold and the design matrix structure.

12.19.8 Practical Tips

- limma-voom is competitive with edgeR's `glmQLFit` and DESeq2's negative-binomial Wald test; choose based on workflow conventions and personal preference rather than expected accuracy gains.
- `voomWithQualityWeights()` additionally down-weights noisy samples, useful when one or two samples have visibly poor QC; combine the two weight types when sample-level variability is substantial.
- For complex designs (interactions, multiple blocking factors), the standard `model.matrix` and `contrasts.fit()` machinery applies; limma's design language is among the most flexible.

- Voom is well-suited to scRNA-seq pseudobulk: aggregate cells per cluster into pseudo-bulk counts, then run voom on the resulting matrix.
- For very few samples per group (n < 3), variance estimates are unstable; consider DESeq2 with its variance-stabilising shrinkage, or rely heavily on the empirical Bayes step in limma.
- Pair the analysis with `voomQCPlot` or `meanvariancetrend()` diagnostic plots to confirm the variance trend is well-fitted.

12.19.9 R Packages Used

`limma` for `voom()`, `lmFit()`, `eBayes()`, `topTable()`, and the broader linear-modelling framework; `edgeR` for `DGEList`, `calcNormFactors` (TMM normalisation); `Glimma` for interactive MA and volcano plots; `EnhancedVolcano` for publication-ready volcano plots from limma output.

12.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.19.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.20 MA Plots

12.20.1 Introduction

An MA plot — also known as a mean-difference plot or Bland-Altman-style plot for genomics — displays the log fold change (M) on the y-axis against the mean expression intensity (A) on the x-axis. It is the standard diagnostic for differential-expression results: if the bulk of points (the non-DE genes) clusters around zero independent of mean intensity, normalisation has succeeded; systematic curvature or vertical drift indicates intensity-dependent bias that contaminates the DE calls. The MA plot complements the volcano plot by emphasising the relationship between effect size and expression level rather than effect size and statistical significance.

12.20.2 Prerequisites

A working understanding of log fold change, average expression on the log scale, and differential-expression output (DESeq2, edgeR, limma).

12.20.3 Theory

For two samples (or two groups) X, Y:

$$M = \log_2(Y/X), \qquad A = \tfrac{1}{2}\log_2(XY).$$

A well-normalised comparison shows M scattered around zero independently of A; the lowess curve of M vs A should be approximately flat at zero. Curvature, drift, or systematic offset indicates scaling or compositional bias that normalisation has failed to remove. The "trumpet" shape — wider scatter at low expression, narrower at high — is expected because low-expression genes have noisier fold-change estimates; this is information about precision, not bias.

12.20.4 Assumptions

Counts have been normalised (TMM, RLE, or similar); the majority of genes are non-differentially expressed (so the lowess of M should pass through zero); the dataset is sufficiently large to estimate the lowess curve stably.

12.20.5 R Implementation

```r
library(limma); library(edgeR)

set.seed(2026)
counts <- matrix(rpois(10000 * 6, lambda = 30), 10000, 6)
group  <- factor(rep(c("A", "B"), each = 3))
dge    <- DGEList(counts); dge <- calcNormFactors(dge)

design <- model.matrix(~ group)
v <- voom(dge, design)
fit <- lmFit(v, design); fit <- eBayes(fit)

plotMD(fit, coef = 2, status = decideTests(fit)[, 2],
       values = c(-1, 1), hl.col = c("#2A9D8F", "#E76F51"),
       main = "MA plot (limma)")
abline(h = 0, lty = 2)
```

12.20.6 Output & Results

`limma::plotMD()` plots logFC vs mean log-CPM with significant genes highlighted in colour and a horizontal reference line at zero. `DESeq2::plotMA()` pro-

duces the same plot from a results object in a single call. Most non-DE genes should cluster tightly around zero with the trumpet shape opening toward low mean expression.

12.20.7 Interpretation

A reporting sentence: "The MA plot (Fig. X) confirms that after TMM normalisation the log fold changes are centred on zero across the full mean-expression range; the lowess curve does not deviate from zero, indicating no intensity-dependent bias and supporting the validity of the differential-expression calls. Significant genes (highlighted) are distributed across the expression range." Always pair MA plots with volcano plots when reporting DE results.

12.20.8 Practical Tips

- Always produce an MA plot alongside the volcano plot when reporting DE results; the two communicate complementary information.
- Overlay a lowess curve (`lines(lowess(A,M))`) to diagnose intensity-dependent bias formally; visible drift signals normalisation problems.
- For DESeq2, `plotMA(res)` produces the standard MA plot in a single call; the function automatically applies LFC shrinkage if the results object contains shrunken estimates.
- Use shrinkage (`DESeq2::lfcShrink` with `apeglm` or `ashr`) to stabilise noisy log-fold-change estimates at low expression; the shrunken MA plot is more interpretable than the raw version.
- Label only the top few genes (5–10); labelling all significant hits makes the plot unreadable.
- For very large MA plots, alpha-blending or hexbin (`smoothScatter`) prevents overplotting in the dense central column.

12.20.9 R Packages Used

`limma::plotMD()` for MA plots from limma fits; `DESeq2::plotMA()` for MA plots from DESeq2 results; `edgeR::plotMD()` for the edgeR analogue; `Glimma::glimmaMA()` for interactive MA plots embeddable in HTML; `ggplot2` for hand-built versions with custom styling.

12.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.20.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.21 Metagenomic Profiling

12.21.1 Introduction

Shotgun metagenomics sequences all DNA in a sample rather than a marker gene, enabling species-level taxonomic and functional profiling. Two dominant approaches: MetaPhlAn (marker-gene alignment to species-specific genes) and Kraken (k-mer matching against a reference database).

12.21.2 Prerequisites

Shotgun sequencing basics; read QC; familiarity with amplicon methods.

12.21.3 Theory

MetaPhlAn maps reads to ~1M clade-specific markers; species abundances are inferred as relative marker coverage. Output is a per-species relative abundance table.

Kraken2 classifies reads via k-mer matches to a database (e.g., RefSeq Bacteria). Fast and comprehensive but prone to false positives for low-abundance species; pair with **Bracken** for abundance re-estimation.

Functional profiling: HUMAnN3 for pathway-level abundances (MetaCyc); eggNOG-mapper for gene-function annotation.

12.21.4 Assumptions

Reads are adapter-trimmed and host-decontaminated; reference database covers the expected taxa; relative abundances are interpretable only within a sample.

12.21.5 R Implementation

MetaPhlAn and Kraken run on the command line; R ingests the results for downstream diversity and DA analysis:

```
library(phyloseq)

set.seed(2026)
# Simulated MetaPhlAn-style output
species <- paste0("s__Species_", 1:50)
samples <- paste0("S", 1:20)
```

```r
abund <- matrix(rgamma(length(species) * length(samples), 0.5, 2),
                length(species), length(samples))
abund <- sweep(abund, 2, colSums(abund), "/")   # relative abundances
rownames(abund) <- species; colnames(abund) <- samples

# Coerce to a phyloseq object for diversity analyses
tax_tab <- matrix(species, length(species), 1,
                  dimnames = list(species, "Species"))
ps <- phyloseq(otu_table(round(abund * 1e6), taxa_are_rows = TRUE),
               tax_table(tax_tab))
ps

# Top species by mean abundance
head(sort(rowMeans(abund), decreasing = TRUE), 5)
```

Example command-line steps (not run here):

```
metaphlan sample.fastq --input_type fastq --bowtie2db metaphlan_db \
          -o sample_profile.txt
kraken2 --db kraken_db --paired R1.fq R2.fq --report sample.kreport
```

12.21.6 Output & Results

A species-by-sample abundance table; in phyloseq, all standard diversity and DA methods apply.

12.21.7 Interpretation

"MetaPhlAn profiling revealed 420 species; top differentially abundant species between gut inflammation cases and controls included Faecalibacterium prausnitzii (depleted) and Escherichia coli (enriched)."

12.21.8 Practical Tips

- Host-read removal (Bowtie2 vs the host genome) is essential; residual host DNA biases classification.
- Kraken2 + Bracken gives per-species quantification with reduced false positives vs Kraken alone.
- For functional analysis, HUMAnN3 provides pathway abundances suitable for differential pathway analysis with MaAsLin2 or LEfSe.
- ONT long reads enable better species resolution in complex communities.
- Do not mix relative and absolute abundances across analyses; compositional data require log-ratio transforms (CLR) or specific methods (ANCOM-BC).

12.21.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.21.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.22 DNA Methylation Analysis

12.22.1 Introduction

DNA methylation at CpG dinucleotides is a stable epigenetic mark that regulates gene expression and varies with age, tissue type, environmental exposures, and disease state. The two dominant assays are Illumina arrays (450K, 850K EPIC) measuring methylation at hundreds of thousands of CpGs and whole-genome bisulfite sequencing (WGBS) covering nearly all CpGs at higher cost. Differential methylation analysis identifies CpG positions or regions whose methylation differs between groups, often serving as biomarkers (e.g., cancer subtype classification, age estimation).

12.22.2 Prerequisites

A working understanding of bisulfite conversion chemistry (unmethylated C $\rightarrow$ T, methylated C $\rightarrow$ C), the difference between beta-values and M-values as methylation summaries, and basic linear-modelling on continuous outcomes.

12.22.3 Theory

Two methylation summaries:

- **Beta-value**: $\beta = M/(M + U + \alpha)$, the proportion of methylated reads (with α a small offset to stabilise low-coverage estimates). Intuitive (0 = fully unmethylated, 1 = fully methylated) but heteroscedastic — variance is largest near 0.5.
- **M-value**: $\log_2(M/U)$, the log-ratio of methylated to unmethylated. Linear and approximately homoscedastic; preferred for regression.

The standard pipeline:

1. Read raw IDAT files (`minfi::read.metharray.exp`).

2. Normalise (SWAN, functional normalisation, BMIQ).
3. Detect and remove outlier samples and cross-reactive probes.
4. Differential methylation at probe level via limma on M-values.
5. Region-level analysis (DMRcate, bumphunter) for smoother small-effect detection.

12.22.4 Assumptions

Array probes are well-designed (no major cross-reactivity); batch effects are identified and modelled; cell-type composition is adjusted for in blood or other heterogeneous samples — failure to do so creates spurious associations.

12.22.5 R Implementation

```r
library(limma)

set.seed(2026)
n_cpgs <- 1000; n_samples <- 12
M_values <- matrix(rnorm(n_cpgs * n_samples, mean = 0, sd = 2),
                   n_cpgs, n_samples)
rownames(M_values) <- paste0("cg", 1:n_cpgs)
colnames(M_values) <- paste0("S", 1:n_samples)

group <- factor(rep(c("ctrl", "case"), each = 6))
# Inject differential methylation in 50 CpGs
M_values[1:50, group == "case"] <- M_values[1:50, group == "case"] + 1.5

design <- model.matrix(~ group)
fit <- lmFit(M_values, design)
fit <- eBayes(fit)

topTable(fit, coef = 2, n = 5)
summary(decideTests(fit))
```

12.22.6 Output & Results

`topTable()` returns a per-CpG table with log fold change in M-value, raw p-value, and BH-adjusted p-value. The 50 injected CpGs should appear at the top of the ranked list with the largest absolute fold changes and most significant p-values.

12.22.7 Interpretation

A reporting sentence: "Epigenome-wide differential methylation analysis (limma on M-values, design adjusting for age and estimated cell-type composition) identified 1{,}240 differentially methylated CpGs at Bonferroni

$p < 10^{-7}$; 62 % were hypermethylated in cases, concentrated in CpG islands of known tumour-suppressor loci. Region-level analysis (DMRcate) identified 87 differentially methylated regions overlapping 64 unique genes." Always state the multiple-testing correction, cell-type adjustment strategy, and use of M-values vs beta-values.

12.22.8 Practical Tips

- Fit regression models on M-values (homoscedastic), but report effect sizes in beta-values (more interpretable as "% methylation difference") for clinical audiences.
- Cell-type deconvolution (`EpiDISH`, `FlowSorted.Blood.450k`) is mandatory for blood-based EWAS; unadjusted results predominantly reflect composition differences.
- Region-level analysis (DMRcate, bumphunter) is more powerful than probe-level for small consistent effects across nearby CpGs; it also reduces multiple-testing burden.
- Batch effects (array plate, chip, scan date) are very strong in methylation data; include as covariates or run ComBat.
- For age-related studies, the Horvath, Hannum, or PhenoAge clocks give DNA methylation age estimates that are biologically interpretable; report alongside chronological age.
- Standardised EWAS pipelines (`meffil`, `PaCES`) offer one-call QC and analysis with sensible defaults.

12.22.9 R Packages Used

`minfi` for IDAT reading, normalisation, and basic differential analysis; `limma` for linear-modelling on M-values; `DMRcate` for region-level differential methylation; `EpiDISH` and `FlowSorted.Blood.450k` for cell-type deconvolution; `bumphunter` for an alternative region detector; `meffil` for an end-to-end EWAS pipeline; `methylKit` for WGBS analysis.

12.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.22.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.23 Microbiome Analysis with DADA2

12.23.1 Introduction

DADA2 (Divisive Amplicon Denoising Algorithm 2) infers exact amplicon sequence variants (ASVs) from 16S rRNA, 18S, ITS, or other amplicon sequencing data. It replaces the older OTU-clustering approach (which lumps similar sequences together at, typically, 97 % identity) with single-nucleotide resolution. ASVs are reproducible across studies — the same sequence is the same variant regardless of who runs the analysis — and allow finer taxonomic assignment than the threshold-based OTU representation.

12.23.2 Prerequisites

A working understanding of amplicon sequencing (paired-end or single-end Illumina), primer trimming, and basic microbial ecology including the OTU vs ASV distinction.

12.23.3 Theory

DADA2 fits a position- and quality-aware error model from the observed FASTQ data, then uses it to distinguish real biological variants from sequencing errors. The intuition: a sequence appearing more often than the error model predicts is a candidate ASV; sequences explained by errors from a more abundant variant are merged into it. The final output is a per-sample ASV count table (analogous to a gene count matrix in RNA-seq) that feeds the same downstream pipeline — alpha and beta diversity, differential-abundance testing, ordination.

The standard pipeline: filter and trim FASTQs, learn error rates, denoise per sample, merge paired-end reads, construct a sequence table, remove chimeras, assign taxonomy.

12.23.4 Assumptions

Primers have been removed before DADA2 input; paired-end reads overlap sufficiently for merging; error rates are sample-specific (DADA2 estimates them per run).

12.23.5 R Implementation

DADA2 processes FASTQs on disk; here we simulate a post-DADA2 ASV count table for downstream analysis:

```
library(phyloseq)

set.seed(2026)
n_asv <- 80; n_samples <- 20
```

```
otu_tab <- matrix(rpois(n_asv * n_samples, lambda = 10), n_asv,
 ↪  n_samples)
rownames(otu_tab) <- paste0("ASV", 1:n_asv)
colnames(otu_tab) <- paste0("S", 1:n_samples)

tax_tab <- matrix(sample(c("Firmicutes", "Bacteroidetes",
 ↪  "Proteobacteria"),
                         n_asv, replace = TRUE), n_asv, 1)
rownames(tax_tab) <- rownames(otu_tab); colnames(tax_tab) <- "Phylum"

meta <- data.frame(group = factor(rep(c("ctrl", "case"), each = 10)),
                   row.names = colnames(otu_tab))

ps <- phyloseq(otu_table(otu_tab, taxa_are_rows = TRUE),
               tax_table(tax_tab),
               sample_data(meta))

sort(sample_sums(ps), decreasing = TRUE)[1:5]   # per-sample depth
taxa_sums(ps)[1:5]                              # per-ASV total
```

The actual DADA2 calls (commented for the simulation example):

```
# filterAndTrim(fnFs, filtFs, fnRs, filtRs, truncLen = c(240, 160),
#               maxEE = c(2, 2), truncQ = 2, compress = TRUE)
# errF <- learnErrors(filtFs); errR <- learnErrors(filtRs)
# dadaFs <- dada(filtFs, err = errF); dadaRs <- dada(filtRs, err = errR)
# mergers <- mergePairs(dadaFs, filtFs, dadaRs, filtRs)
# seqtab <- makeSequenceTable(mergers)
# seqtab.nochim <- removeBimeraDenovo(seqtab, method = "consensus")
```

12.23.6 Output & Results

The DADA2 output is an ASV-by-sample count matrix together with taxonomy assignments from SILVA or GTDB. Wrapping the count table, taxonomy, and sample metadata in a `phyloseq` object provides the standard input for downstream community ecology analyses.

12.23.7 Interpretation

A reporting sentence: “DADA2 inferred 2{,}148 amplicon sequence variants across 50 samples after chimera removal; 980 ASVs were assigned to phylum level using the SILVA v138 reference, with median read depth of 42{,}300 reads per sample. The resulting `phyloseq` object served as input to alpha-diversity (Shannon, Simpson) and beta-diversity (Bray-Curtis NMDS) analyses.” Always state the reference database version and the read-retention rate.

12.23.8 Practical Tips

- Always run `removeBimeraDenovo()` after sequence-table construction to strip chimeric ASVs; they accumulate during PCR and can be a substantial fraction of the count.
- Assign taxonomy with SILVA (v138 or later for 16S) or GTDB; species-level assignment for short amplicons is usually unreliable due to limited resolution.
- Filter ASVs present in fewer than 2 samples to reduce noise; very rare ASVs are often sequencing artefacts.
- `DECIPHER::IdTaxa()` provides a faster alternative to DADA2's `assignTaxonomy()` and supports the same SILVA reference.
- Always check per-sample read-retention at each DADA2 step (filtering, denoising, merging, chimera removal); large drops at any step usually mean over-strict parameters.
- For 16S V4 region with overlapping 250 bp paired-end reads, `truncLen=c(240,160)` is a sensible starting point; tune based on quality profiles.

12.23.9 R Packages Used

`dada2` for ASV inference and the canonical workflow; `phyloseq` for downstream community-ecology analysis on the ASV count table; `DECIPHER` for fast taxonomy assignment; `vegan` for alpha and beta diversity; `ANCOMBC` and `ALDEx2` for compositional differential-abundance testing.

12.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.23.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.24 Microbiome Diversity Metrics

12.24.1 Introduction

Diversity metrics summarise microbial community composition. **Alpha diversity** measures richness and evenness within a sample; **beta diversity** measures

compositional dissimilarity between samples. Both are essential for comparing communities and detecting treatment, disease, or niche effects.

12.24.2 Prerequisites

ASV or OTU count table; phylogenetic tree (for UniFrac); sample metadata.

12.24.3 Theory

Alpha diversity: - **Richness** (# ASVs observed). - **Shannon index** $H = -\sum p_i \log p_i$ – evenness-weighted. - **Simpson** $1 - \sum p_i^2$ – inverse concentration. - **Faith's PD** – phylogenetic tree branch length sum.

Beta diversity: - **Bray-Curtis** – abundance-weighted Manhattan dissimilarity. - **Jaccard** – presence/absence. - **UniFrac (weighted / unweighted)** – phylogeny-aware.

Permutational MANOVA (`adonis2`) tests whether group membership explains beta diversity.

12.24.4 Assumptions

Counts are non-negative; rarefaction or equivalent normalisation is applied before alpha diversity if depths vary widely; phylogenetic tree matches ASV labels.

12.24.5 R Implementation

```r
library(vegan)

set.seed(2026)
n_samples <- 20; n_asv <- 50
counts <- matrix(rpois(n_asv * n_samples, lambda = 10),
                 n_samples, n_asv)   # samples x ASVs

# Inject group differences
group <- factor(rep(c("ctrl", "case"), each = 10))
counts[group == "case", 1:15] <- counts[group == "case", 1:15] + 20

shannon <- diversity(counts, index = "shannon")
simpson <- diversity(counts, index = "simpson")
richness <- rowSums(counts > 0)

by(shannon, group, median)

bray <- vegdist(counts, method = "bray")
adonis_res <- adonis2(bray ~ group, permutations = 999)
adonis_res
```

12.24.6 Output & Results

Shannon / Simpson / richness per sample (group medians printed), and a PERMANOVA on Bray-Curtis showing whether group explains dissimilarity.

12.24.7 Interpretation

"Case samples had lower Shannon diversity (median 2.1 vs 2.9, Wilcoxon p = 0.004) and PERMANOVA on Bray-Curtis revealed significant community-level differences ($R^2 = 0.18$, p = 0.001)."

12.24.8 Practical Tips

- Rarefy to equal depth before alpha diversity to avoid depth-driven confounding; alternatively use rarefaction curves to confirm coverage.
- `phyloseq::estimate_richness` computes many alpha metrics in one call.
- Weighted UniFrac captures abundance-weighted phylogenetic distance; unweighted is presence-only.
- PERMANOVA is sensitive to dispersion heterogeneity; run `betadisper` to confirm equal dispersion before interpreting.
- PCoA visualisation on the beta dissimilarity matrix complements PERMANOVA.

12.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.24.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.25 Multiple Sequence Alignment

12.25.1 Introduction

Multiple sequence alignment (MSA) is the precursor to phylogeny, motif discovery, and structural modelling: it arranges residues so homologous positions align in columns. The `msa` Bioconductor package offers ClustalW, ClustalOmega, and Muscle under a unified R interface.

12.25.2 Prerequisites

FASTA sequences; substitution scoring (PAM, BLOSUM for proteins; match/mismatch for DNA).

12.25.3 Theory

Progressive alignment (ClustalW): build a guide tree from pairwise distances, then align sequences following the tree from leaves to root. **Iterative refinement** (Muscle, MAFFT) revisits partial alignments to improve scores. **Consistency-based** methods (T-Coffee) combine information from multiple library alignments.

Accuracy depends on sequence divergence: for highly divergent sequences, structural alignment (DALI for proteins) outperforms sequence-only methods.

12.25.4 Assumptions

Sequences are homologous; a single substitution model approximates the evolutionary process; insertions/deletions are rare enough for gap parameters to be meaningful.

12.25.5 R Implementation

```
library(Biostrings); library(msa)

# Example amino-acid sequences (hemoglobin across species)
aas <- readAAStringSet(
  system.file("examples", "exampleAA.fasta", package = "msa"))
aas
aln <- msa(aas, method = "ClustalW", verbose = FALSE)
aln

# Summary of alignment
print(aln, show = "complete")

# Convert to phangorn for phylogeny
library(phangorn)
aln_phy <- as.phyDat(as(aln, "AAStringSet"), type = "AA")
dm <- dist.ml(aln_phy)
tree <- NJ(dm)
plot(tree, main = "NJ tree from MSA", cex = 0.8)
```

12.25.6 Output & Results

An MSA object with aligned sequences and gap characters; a consensus is derivable via `consensusString`.

12.25.7 Interpretation

"ClustalW alignment of the hemoglobin orthologs reveals conserved haem-binding residues and lineage-specific substitutions on the surface helices."

12.25.8 Practical Tips

- For > 1000 sequences, MAFFT (outside R) is substantially faster than Biostrings/msa.
- Alignment quality matters more than tree-building algorithm; a bad MSA produces a bad phylogeny.
- Trim alignment columns with many gaps (e.g., `trimAl`) before tree inference.
- Visualise alignments with `ggmsa` to identify problematic columns.
- Amino-acid alignments are often more reliable than codon alignments for distantly related sequences; use back-translation when codon-level analysis is needed.

12.25.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.25.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.26 Multi-Omics Integration

12.26.1 Introduction

Multi-omics experiments measure transcriptome, methylome, proteome, etc. on the same samples. Joint analysis exposes coordinated variation across layers that single-layer analyses miss – cis-regulatory effects of methylation, mRNA-protein correspondence, or integrated disease subtypes.

12.26.2 Prerequisites

Single-omic analysis (DE, methylation, proteomics); PCA / factor analysis basics.

12.26.3 Theory

MOFA (Multi-Omics Factor Analysis) learns latent factors explaining variation across omics layers. Each factor has per-layer weights: factors loading strongly on multiple layers capture shared signal; single-layer factors reflect layer-specific variation. MOFA2 adds sparsity for interpretability.

Similarity network fusion (SNF) and **iCluster+** offer alternative approaches; DIABLO (`mixOmics`) builds supervised multi-omic classifiers.

12.26.4 Assumptions

Samples are matched across layers; layer-specific noise and scaling are handled by the model; sufficient sample size for factor inference (typically > 50).

12.26.5 R Implementation

```
library(MOFA2)

set.seed(2026)
n <- 50
# Latent factor drives correlated variation across 2 layers
z <- rnorm(n)

# Layer 1: 200 features (e.g., RNA)
m_rna <- matrix(rnorm(200 * n, 0, 1), 200, n)
m_rna[1:20, ] <- m_rna[1:20, ] + outer(rep(1.5, 20), z)

# Layer 2: 100 features (e.g., methylation)
m_meth <- matrix(rnorm(100 * n, 0, 1), 100, n)
m_meth[1:10, ] <- m_meth[1:10, ] - outer(rep(1.2, 10), z)

colnames(m_rna) <- colnames(m_meth) <- paste0("S", 1:n)
rownames(m_rna)  <- paste0("rna_",  1:200)
rownames(m_meth) <- paste0("meth_", 1:100)

mofa <- create_mofa(list(RNA = m_rna, Methylation = m_meth))
# Illustrative: actual fitting requires Python backend (reticulate +
 ↪  mofapy2).
# model_opts <- get_default_model_options(mofa)
# model_opts$num_factors <- 3
# mofa_trained <- run_mofa(mofa, training_data_options = list(),
#                          model_options = model_opts, outfile = tempfile())

mofa
```

12.26.6 Output & Results

A trained MOFA model with latent-factor scores per sample, per-layer feature weights, and variance decomposition per factor x layer.

12.26.7 Interpretation

"MOFA identified 5 latent factors; factor 1 explained 32 % of RNA and 18 % of methylation variance, linked to cancer subtype; factor 2 was RNA-specific and reflected proliferation signature."

12.26.8 Practical Tips

- Standardise each layer before MOFA (mean = 0, SD = 1) so no one layer dominates.
- Layer-specific factors often reflect technical artefacts; use them to diagnose batch effects.
- `DIABLO` in `mixOmics` is the better choice when integration is supervised by an outcome.
- MOFA requires Python via `reticulate`; test the install with `run_mofa2_docs()` before large runs.
- Interpret factor weights with domain knowledge; algorithmic factors without biology are uninformative.

12.26.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.26.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.27 KEGG Pathway Enrichment

12.27.1 Introduction

KEGG (Kyoto Encyclopedia of Genes and Genomes) is a curated database of metabolic and signalling pathways represented as directed graphs of genes and gene products. KEGG pathway enrichment asks whether the genes in a

differential-expression hit list are over-represented in any of the curated pathways, identifying functional themes in the differential signal. The companion visualisation tool `pathview` overlays log-fold-change values directly onto the canonical KEGG pathway diagrams, making it straightforward to communicate which specific genes within a pathway drive the enrichment signal.

12.27.2 Prerequisites

A working understanding of differential-expression analysis, Entrez gene identifiers, and basic over-representation analysis using hypergeometric or Fisher's exact tests.

12.27.3 Theory

KEGG enrichment uses the same statistical framework as GO over-representation analysis: a hypergeometric (Fisher's exact) test for each pathway compares the proportion of DE genes assigned to the pathway against the proportion expected by chance from the universe of all annotated genes. Adjusted p-values (BH or Benjamini-Hochberg) control the false-discovery rate across the many pathways tested.

Pathway visualisation via `pathview` rendering colours each gene node in the canonical KEGG diagram by user-supplied log fold change (or any continuous statistic), producing an annotated PNG/PDF of the pathway map. The visual immediately exposes whether the enrichment is driven by a coordinated up- or down-regulation across the pathway versus a few isolated hits.

12.27.4 Assumptions

DE genes are correctly mapped to Entrez IDs; KEGG pathway annotations are current (KEGG is updated regularly; old annotations miss recent additions); tests are independent across pathways (BH adjustment is conservative for the typical correlated-pathway structure).

12.27.5 R Implementation

```
library(clusterProfiler); library(org.Hs.eg.db)

set.seed(2026)
all_entrez <- keys(org.Hs.eg.db, keytype = "ENTREZID")
de_entrez  <- sample(all_entrez, 200)

ek <- enrichKEGG(gene = de_entrez,
                 organism = "hsa",
                 pvalueCutoff = 0.1)
head(as.data.frame(ek))[, c("ID", "Description", "p.adjust", "Count")]
```

```
# Visualise a pathway with LFC overlay
# library(pathview)
# lfc_vec <- setNames(rnorm(length(de_entrez)), de_entrez)
# pathview(gene.data = lfc_vec, pathway.id = "hsa04110", species = "hsa")
```

12.27.6 Output & Results

`enrichKEGG()` returns an `enrichResult` object with one row per pathway, including the pathway ID and description, the count of DE genes, BH-adjusted p-value, and the comma-separated list of contributing gene symbols. `pathview()` writes a PNG of the pathway diagram with each gene node coloured by its supplied log-fold-change value.

12.27.7 Interpretation

A reporting sentence: "KEGG enrichment with `clusterProfiler::enrichKEGG` (organism = hsa, BH-adjusted $p < 0.05$) identified the cell cycle (hsa04110, $p_{\text{adj}} = 0.001$, 23 genes) and p53 signalling (hsa04115, $p_{\text{adj}} = 0.008$, 14 genes) as top enriched pathways; pathview overlay on hsa04110 showed consistent up-regulation of cyclins and CDKs alongside down-regulation of CDK inhibitors." Always state the species code, the universe of background genes, and the multiple-testing correction.

12.27.8 Practical Tips

- KEGG functions in `clusterProfiler` require Entrez IDs; convert from gene symbols via `bitr(symbols,fromType="SYMBOL",toType="ENTREZID",OrgDb=org.Hs.eg.db)`.
- `enrichMKEGG()` uses KEGG modules (smaller, more specific functional units); it is preferable for metabolic studies where the broader KEGG pathway granularity is too coarse.
- `pathview()` requires internet access and writes output files to the working directory; treat each rendering as a one-off rather than running it in an automated pipeline.
- KEGG organism codes: `hsa` for human, `mmu` for mouse, `rno` for rat, `dme` for fly, `cel` for worm; a complete list is at the KEGG website.
- Combine KEGG (broad pathways) with Reactome (`ReactomePA::enrichPathway`) for complementary coverage; the two databases overlap but emphasise different aspects of biology.
- For genes with associated fold-change estimates, GSEA on KEGG pathways (`gseKEGG()`) is more powerful than over-representation analysis because it uses the full ranked gene list rather than thresholding into hit and non-hit sets.

12.27.9 R Packages Used

`clusterProfiler` for `enrichKEGG()`, `enrichMKEGG()`, `gseKEGG()`, and the broader enrichment-analysis framework; `pathview` for KEGG pathway visualisation with LFC overlay; `org.Hs.eg.db` (and species-specific equivalents) for Entrez ID conversion; `ReactomePA` for Reactome alternatives; `enrichplot` for ggplot-based visualisations of enrichment results.

12.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.27.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.28 PCA of RNA-seq Samples

12.28.1 Introduction

Sample-level PCA on transformed RNA-seq counts is the standard quality-control step before differential-expression analysis. It reveals whether biological samples cluster by experimental condition (the desired outcome), by batch or technical covariates (a confounding signal that needs correction), or by neither (which can suggest excessive noise or sample mix-ups). Running PCA early — before any modelling — prevents the most consequential downstream errors and motivates the design choices for the differential-expression step.

12.28.2 Prerequisites

A working understanding of principal components analysis, RNA-seq count data, and the role of variance-stabilising transformations in equalising the variance-mean relationship of count data.

12.28.3 Theory

Raw RNA-seq counts have variance that grows with the mean, violating the assumption of equal variance that PCA implicitly relies on. Variance-stabilising transformations (VST and the older `rlog` in DESeq2) apply a non-linear transformation that approximately equalises variance across the expression range;

the result is suitable for PCA, clustering, and other downstream methods that assume homoscedastic input.

PCA on the top-variance genes (typically the top 500 or so) highlights the dominant sources of variation. The proportion of variance explained by each PC indicates how much of the total variability is captured, and the score plot — samples coloured by metadata variables — exposes which design factors drive that variability.

12.28.4 Assumptions

Sufficient samples (rule of thumb: at least 6, ideally more) for meaningful PCA; counts have been variance-stabilised; the chosen number of top-variance genes is large enough to capture the structure but small enough to suppress noise.

12.28.5 R Implementation

```
library(DESeq2); library(ggplot2)

set.seed(2026)
counts <- matrix(rpois(20000 * 12, lambda = 30), nrow = 20000)
colnames(counts) <- paste0("s", 1:12)
coldata <- data.frame(condition = rep(c("A", "B"), each = 6),
                      batch = rep(c("X", "Y"), 6))

dds <- DESeqDataSetFromMatrix(counts, colData = coldata, design = ~
 ↪  condition)
vsd <- vst(dds, blind = TRUE)

plotPCA(vsd, intgroup = c("condition", "batch"))
```

12.28.6 Output & Results

`plotPCA()` returns a ggplot of samples projected onto PC1 and PC2 with axis labels showing the proportion of variance explained. Colouring by `condition` and shape by `batch` (or vice versa) makes both factors visible simultaneously.

12.28.7 Interpretation

A reporting sentence: "Sample-level PCA on VST-transformed counts (top 500 variance genes) showed PC1 capturing 40 % of variance and clearly separating treatment conditions, while PC2 (12 %) showed a weaker batch effect. We included `batch` in the differential-expression design to adjust for it." Always report the proportion of variance explained and what each PC corresponds to substantively.

12.28.8 Practical Tips

- Use `blind=TRUE` for QC PCA so the variance-stabilisation does not shrink toward the design; for downstream DE, the design-aware (`blind=FALSE`) transformation is appropriate.
- If batch dominates PC1, include batch in the DESeq2 (or limma) design; if batch overlaps perfectly with condition, the experiment is confounded and cannot be salvaged.
- Plot multiple metadata colourings (sex, age, RIN, library prep date) on the same PCA scatter to reveal hidden structure; outlier samples often correspond to specific covariates.
- Complement PCA with a sample-sample distance heatmap (`pheatmap(as.matrix(dist(t(assay(vsd)))))`); the two views agree on the broad story but emphasise different details.
- Outliers on PCA often indicate QC failures (low library complexity, contamination); investigate and consider exclusion before differential analysis.
- Compare PCA on raw counts, log-counts, and VST; raw counts have variance dominated by highly expressed genes, log-counts overcorrect, VST is calibrated for downstream analysis.

12.28.9 R Packages Used

`DESeq2` for `vst()`, `rlog()`, `plotPCA()` and the broader RNA-seq workflow; `pheatmap` for sample-distance heatmaps; `ggplot2` for custom PCA plots; `PCAtools` for richer PCA visualisations including biplots and loading plots.

12.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.28.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.29 Phylogenetic Trees with ape

12.29.1 Introduction

Phylogenetic trees depict the evolutionary relationships among sequences (species, strains, cells). They are central to taxonomy, molecular epidemiology,

and cancer-clone tracing. `ape` is the standard R package for tree construction and manipulation, and `phangorn` adds maximum-likelihood and parsimony methods.

12.29.2 Prerequisites

Multiple sequence alignment; nucleotide or amino-acid substitution models.

12.29.3 Theory

Distance-based methods (neighbour-joining, UPGMA) compute pairwise distances from the alignment and build a tree that approximates those distances; fast but statistically crude.

Maximum-likelihood (ML) methods posit a substitution model (JC69, K80, GTR + Gamma + I) and find the tree topology + branch lengths maximising the likelihood of the observed alignment.

Bootstrapping resamples alignment columns to assess topology confidence; bootstrap values ≥ 70 % are commonly considered well-supported.

12.29.4 Assumptions

Alignment columns are homologous and independent; the substitution model captures key features (transition/transversion bias, rate heterogeneity).

12.29.5 R Implementation

```
library(ape)

# Load an example alignment shipped with ape
data(woodmouse)          # DNAbin alignment of 15 sequences, 965 sites

# Distance matrix under K80 model
d <- dist.dna(woodmouse, model = "K80")

# Neighbour-joining tree
nj_tree <- nj(d)
plot(nj_tree, main = "Neighbour-joining tree (woodmouse)",
     cex = 0.6, font = 1)
add.scale.bar()

# Bootstrap support
bp <- boot.phylo(nj_tree, woodmouse,
                 function(x) nj(dist.dna(x, model = "K80")),
                 B = 100, quiet = TRUE)
plot(nj_tree, main = "NJ tree with bootstrap support",
```

```
     cex = 0.6, font = 1)
nodelabels(bp, cex = 0.5, frame = "none", adj = c(1.3, -0.3))
```

12.29.6 Output & Results

A neighbour-joining tree with branch lengths in substitutions per site; bootstrap values annotated at internal nodes.

12.29.7 Interpretation

"The neighbour-joining tree of the cytochrome b sequences recovers the regional subclades with bootstrap support > 80 %, consistent with the geographic sampling structure."

12.29.8 Practical Tips

- For genome-scale data, use ML tools like `RAxML-NG` or `IQ-TREE` externally and import trees with `ape::read.tree()`.
- NJ is fast but sensitive to model misspecification; ML is the gold standard for small-moderate alignments.
- Use `phangorn::modelTest()` to pick a substitution model by AIC/BIC before running ML.
- Annotate trees with `ggtree` for publication-ready figures including metadata overlays.
- For large strain trees (SARS-CoV-2), use Nextstrain / `treetime` rather than base ape.

12.29.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.29.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.30 Population Genetics Basics

12.30.1 Introduction

Population genetics quantifies genetic variation within and between populations. Allele frequencies, Hardy-Weinberg equilibrium (HWE), and F-statistics form the foundational toolkit for diagnosing genotyping errors, detecting population structure, and assessing selection or drift.

12.30.2 Prerequisites

Mendelian inheritance; genotype coding (0/1/2 for minor-allele dosage).

12.30.3 Theory

Allele frequency. For a biallelic SNP with minor-allele counts n_{AA}, n_{Aa}, n_{aa}, $p = (2n_{AA} + n_{Aa})/(2n)$.

Hardy-Weinberg equilibrium (HWE). Under random mating and no selection, expected genotype frequencies are $p^2, 2pq, q^2$. Exact HWE tests (`HWExact`) are preferred over the chi-square for rare variants.

F-statistics. $F_{ST} = (H_T - H_S)/H_T$ partitions heterozygosity between and within populations, quantifying structure (Wright, 1951). Typical human F_{ST} between continents ≈ 0.1.

12.30.4 Assumptions

Random mating, no selection, no mutation, no migration, infinite population size; genotyping is unbiased.

12.30.5 R Implementation

```
library(HardyWeinberg)

set.seed(2026)
# Simulate HWE: p = 0.3, n = 200 diploid individuals
p <- 0.3
n_AA <- rbinom(1, 200, p^2)
n_aa <- rbinom(1, 200, (1 - p)^2)
n_Aa <- 200 - n_AA - n_aa

gt <- c(AA = n_AA, AB = n_Aa, BB = n_aa)
HWExact(gt, verbose = TRUE)

# Simulate HWE violation (excess heterozygotes)
gt_violate <- c(AA = 20, AB = 160, BB = 20)
```

```r
HWExact(gt_violate, verbose = TRUE)

# F_ST between two populations (manual calculation)
p1 <- 0.2; p2 <- 0.5
p_bar <- mean(c(p1, p2))
H_T <- 2 * p_bar * (1 - p_bar)
H_S <- mean(c(2 * p1 * (1 - p1), 2 * p2 * (1 - p2)))
Fst <- (H_T - H_S) / H_T
Fst
```

12.30.6 Output & Results

HWE exact test p-value (large for the simulated HWE case; ~0 for the violated case). $F_{ST} \approx 0.1$ for the two-population toy example.

12.30.7 Interpretation

"HWE exact tests excluded 1,240 variants (p < 1e-6) from association analysis as likely genotyping artefacts. Between-cohort $F_{ST} = 0.012$ indicates minimal population stratification after ancestry matching."

12.30.8 Practical Tips

- Always HWE-filter variants in GWAS pre-processing; strong HWE violations usually indicate genotyping errors.
- Test HWE in controls only (not cases) because true disease associations can cause HWE departures in cases.
- For low-MAF variants (<5%), asymptotic chi-square is unreliable; use the exact test.
- `adegenet` and `hierfstat` provide comprehensive F-statistics and population-structure utilities.
- F_{ST} between samples can be inflated by small sample size; apply Weir-Cockerham estimator and bootstrap CIs.

12.30.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.30.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR

- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.31 Protein Structure Prediction

12.31.1 Introduction

AlphaFold 2 (2021) achieved near-experimental accuracy in predicting protein 3-D structure from amino-acid sequence, transforming structural biology. RoseTTAFold and ESMFold offer competitive alternatives with different speed/accuracy trade-offs. Downstream analyses include structural alignment, docking, and mutation impact prediction.

12.31.2 Prerequisites

Amino-acid sequence; familiarity with protein structure (secondary/tertiary elements).

12.31.3 Theory

AlphaFold uses attention-based deep learning trained on PDB structures and multiple-sequence-alignment (MSA) features. The model outputs per-residue (x, y, z) coordinates plus a per-residue confidence score (**pLDDT**, 0-100) and an inter-residue confidence (**PAE**).

pLDDT > 70 = confident; 50-70 = low confidence; < 50 = disordered/unreliable.

12.31.4 Assumptions

The protein has evolutionary neighbours (sufficient MSA) or is not unusually divergent; post-translational modifications and co-factors are not modelled.

12.31.5 R Implementation

AlphaFold prediction itself runs outside R (ColabFold, AlphaFold Docker, OpenFold). R is used for structural analysis via `bio3d`:

```r
library(bio3d)

# Read a PDB (here a built-in example: human CDK2)
pdb <- read.pdb(system.file("examples/hivp.pdb", package = "bio3d"))
summary(pdb)

# Secondary structure, residue count, chains
head(pdb$atom[, c("resno", "resid", "chain", "elety")])

# Torsion angles
```

```r
tor <- torsion.pdb(pdb)
head(tor$phi)

# Structural alignment of two chains (illustrative)
chain_a <- atom.select(pdb, chain = "A", elety = "CA")
chain_b <- atom.select(pdb, chain = "B", elety = "CA")
xyz_a <- pdb$xyz[chain_a$xyz]
xyz_b <- pdb$xyz[chain_b$xyz]
c(n_a = length(xyz_a) / 3, n_b = length(xyz_b) / 3)
```

12.31.6 Output & Results

A parsed PDB structure with atoms, residues, and chains accessible programmatically; torsion angles (phi, psi) define the backbone conformation.

12.31.7 Interpretation

"AlphaFold prediction of the novel kinase scored pLDDT 88.3 overall; the activation loop (residues 185-210) had pLDDT < 60, consistent with conformational flexibility seen in homologous kinases."

12.31.8 Practical Tips

- Always check pLDDT per residue; low-confidence regions are unreliable for structural analysis.
- PAE plots reveal inter-domain uncertainty; two confident domains with high PAE between them means the relative orientation is unreliable.
- For mutation impact, combine AlphaFold + molecular-dynamics simulation with Gromacs or AMBER.
- AlphaFold-Multimer handles complexes; standard AlphaFold is monomer-focused.
- ESMFold is faster but less accurate for novel folds; use for screening large sequence sets.

12.31.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.31.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR

- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.32 Proteomics with MSstats

12.32.1 Introduction

MSstats is the standard Bioconductor package for statistical analysis of bottom-up mass-spectrometry proteomics. It ingests peptide-level intensity from any search engine (MaxQuant, Skyline, OpenMS, Spectronaut) and fits mixed-effects models with protein-level contrasts for label-free (LFQ), TMT, and SRM/PRM workflows.

12.32.2 Prerequisites

Peptide-level MS data; experimental-design matrix (condition, bioreplicate, run).

12.32.3 Theory

For each protein, MSstats fits a linear mixed-effects model with run/feature random effects and condition fixed effects; the contrast produces a log-fold change and a moderated t-test analogue. Missing-value handling uses accelerated failure-time imputation or censoring (AFT). LOG2 intensity is the response scale.

12.32.4 Assumptions

Peptide intensity is a valid surrogate for protein abundance; missingness is either completely at random (MCAR) or left-censored at a detection limit (MNAR).

12.32.5 R Implementation

```
library(MSstats)

# Simulated MSstats-format table: peptide intensity x run x condition
set.seed(2026)
proteins <- paste0("P", 1:20)
pep_per_prot <- 3
runs <- paste0("R", 1:6)
conditions <- rep(c("ctrl", "trt"), each = 3)

df <- expand.grid(Protein = proteins,
                  Feature = 1:pep_per_prot,
                  Run = runs,
```

```
                  stringsAsFactors = FALSE)
df$Condition    <- conditions[match(df$Run, runs)]
df$BioReplicate <- df$Run
df$Intensity    <- 2^(rnorm(nrow(df), mean = 20, sd = 1))

# Inject DE in first 5 proteins
trt_rows <- df$Condition == "trt" & df$Protein %in% proteins[1:5]
df$Intensity[trt_rows] <- df$Intensity[trt_rows] * 2

# Mock up the MSstats-required column names and minimum columns
msdf <- data.frame(
  ProteinName = df$Protein,
  PeptideSequence = paste0("PEP", df$Feature),
  PrecursorCharge = 2,
  FragmentIon = NA, ProductCharge = NA,
  IsotopeLabelType = "L",
  Condition = df$Condition,
  BioReplicate = df$BioReplicate,
  Run = df$Run,
  Intensity = df$Intensity
)

# MSstats workflow: processing + protein-level model + contrast
# processed <- dataProcess(msdf)
# comparison <- matrix(c(-1, 1), nrow = 1,
#                      dimnames = list("trt_vs_ctrl", c("ctrl", "trt")))
# results <- groupComparison(contrast.matrix = comparison, data =
 ↪  processed)
# head(results$ComparisonResult)
head(msdf, 3)
```

12.32.6 Output & Results

Per-protein log-fold change, standard error, p-value, and adjusted p-value; MSstats also outputs processed data diagnostic plots (profile plots, condition plots) for QC.

12.32.7 Interpretation

"MSstats identified 142 proteins differentially abundant at BH-FDR < 0.05 between treatment and control; the top hit (log2FC $= 3.2$) corresponds to the direct drug target."

12.32.8 Practical Tips

- Provide replicate structure via `BioReplicate`; technical replicates must be distinct from biological replicates.
- Normalise within MSstats (`dataProcess(normalization="equalizeMedians")`) rather than pre-normalising, to keep the pipeline single-entry.
- For TMT, use `MSstatsTMT`; the mixed-effects model differs due to isobaric design structure.
- Filter peptides to unique-to-protein before modelling; shared peptides inflate between-protein correlation.
- Report both LFC and adjusted p-value; very high LFC with wide CI (few peptides) is weak evidence.

12.32.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.32.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.33 Read Trimming and Adapter Removal

12.33.1 Introduction

After FASTQ-level QC has identified low-quality bases and adapter contamination, read trimming removes them before alignment. The motivation is straightforward: sequencing-quality scores typically degrade toward the 3' end of reads, and adapter sequences appear when the insert is shorter than the read length. Both produce mismatched bases that bias alignment and increase mapping rates incorrectly. The `Rfastp` package provides an R-friendly interface to `fastp`, the fast C++ tool that handles trimming, adapter removal, quality filtering, and polytail cleanup in a single pass.

12.33.2 Prerequisites

A working understanding of FASTQ format, Phred quality scores, and Illumina adapter sequences (TruSeq, Nextera).

12.33.3 Theory

Standard trimming operations:

- **Quality trimming**: scan from the 3' end (and optionally 5') and remove bases falling below a Phred threshold; sliding-window variants trim when the mean quality in a window drops below a cutoff.
- **Adapter removal**: detect known adapter sequences via overlap matching with the read 3' end and trim them out. Auto-detection (matching to a library of common adapters) is usually reliable.
- **Length filtering**: discard reads that fall below a minimum length after trimming; very short reads align ambiguously and waste downstream resources.
- **Polytail trimming**: remove poly-A or poly-G contamination, common in NextSeq/NovaSeq dark-cycle artefacts.

12.33.4 Assumptions

Adapter sequences are known or auto-detectable; quality scores are on the standard Phred+33 encoding (true for all modern Illumina output).

12.33.5 R Implementation

```
library(Rfastp)

in_file  <- system.file("extdata", "reads1.fastq.gz", package = "Rfastp")
out_file <- tempfile(fileext = ".fastq.gz")

# Trim with fastp default heuristics
out <- rfastp(read1 = in_file, outputFastq = out_file,
              thread = 2,
              trimAdapter = TRUE, overlapDiff = 5)

out$summary
```

12.33.6 Output & Results

`rfastp()` returns a summary list with before/after statistics: reads passing filter, reads with adapters trimmed, mean read length before and after, base content. The HTML report (auto-generated by `fastp`) provides a richer visualisation of the trimming impact.

12.33.7 Interpretation

A reporting sentence: "Read trimming with fastp removed Illumina TruSeq adapters from 3.2 % of reads and quality-trimmed bases below Phred 15; 98 %

of reads were retained after trimming, with mean read length decreasing from 150 to 145 bp." Always report the adapter set, quality threshold, and retention rate.

12.33.8 Practical Tips

- Keep trimming settings mild — aggressive quality filtering removes data without proportional gains in downstream accuracy.
- Adapter auto-detection in `fastp` is usually reliable; specify adapter sequences explicitly only when working with non-standard library prep kits.
- For paired-end reads, trim both mates simultaneously and retain only properly-paired survivors; `Rfastp::rfastp()` with `read1` and `read2` arguments handles this automatically.
- Validate downstream QC (FastQC, MultiQC) after trimming; a successful trim improves per-base quality, per-base content, and adapter-content panels.
- Record trimming parameters in a methods document or workflow file (Snakemake, Nextflow); reproducibility requires exact tool versions and parameters.
- For UMI-tagged libraries (single-cell, low-input), use UMI-aware tools (`umi_tools`, fastp's UMI mode) before quality trimming to preserve UMI integrity.

12.33.9 R Packages Used

`Rfastp` for the canonical fastp wrapper; `ShortRead` for general FASTQ manipulation in R; `Biostrings` for adapter-sequence handling; `QuasR` for an integrated alignment workflow that includes trimming as a step.

12.33.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.33.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.34 RNA-seq Normalisation

12.34.1 Introduction

Raw RNA-seq read counts depend on library size and library composition; normalisation removes these technical sources of variation so that samples can be compared on equal footing. The standard methods — TMM (edgeR), RLE (DESeq2), upper-quartile, and CPM — differ in how aggressively they handle compositional differences and which assumption they make about the bulk of genes. Choosing the right normalisation is foundational; downstream differential-expression tools incorporate the normalisation factors as offsets in their GLM, so a poor choice undermines every conclusion that follows.

12.34.2 Prerequisites

A working understanding of count data, library size, and the difference between technical (sequencing depth) and biological (true expression) variation between samples.

12.34.3 Theory

The four standard methods:

- **CPM (counts per million)**: simple division of each count by the total library size in millions. Adjusts only for sequencing depth; does not address compositional differences (where one sample has a few highly expressed genes pulling resources from others).
- **TMM (Trimmed Mean of M-values)**: edgeR's default. Computes log-fold-changes between each sample and a reference, trims extreme values, and uses the trimmed mean as the normalisation factor. Robust to highly expressed compositional outliers.
- **RLE (Relative Log Expression)**: DESeq2's `estimateSizeFactors`. Takes the median of the ratio of each gene's count to its geometric mean across samples. Equivalent to TMM in spirit; both adjust for composition.
- **Upper-quartile**: scaling by the 75th-percentile count; less robust than TMM/RLE but useful for very sparse data.

TMM and RLE are the de facto standards; CPM is appropriate only for visualisation, not for statistical comparison.

12.34.4 Assumptions

Most genes are not differentially expressed (the "majority is unchanged" assumption that motivates trimmed-mean and median-based normalisation); samples are technical replicates of comparable biological material.

12.34.5 R Implementation

```r
library(edgeR); library(DESeq2)

counts <- matrix(rpois(1000, lambda = 10), 100, 10)
rownames(counts) <- paste0("g", 1:100)

# edgeR TMM
dge <- DGEList(counts = counts)
dge <- calcNormFactors(dge, method = "TMM")
dge$samples$norm.factors

# DESeq2 size factors
dds <- DESeqDataSetFromMatrix(counts, colData = DataFrame(id = 1:10),
                              design = ~ 1)
dds <- estimateSizeFactors(dds)
sizeFactors(dds)
```

12.34.6 Output & Results

`calcNormFactors()` returns per-sample TMM normalisation factors (centred near 1, with deviations indicating compositional differences); `estimateSizeFactors()` returns DESeq2 size factors with similar interpretation. Both are used as offsets in downstream GLMs by edgeR, DESeq2, and limma-voom.

12.34.7 Interpretation

A reporting sentence: "TMM normalisation factors ranged from 0.85 to 1.18 across the 10 samples, with no factor exceeding the conventional ±20 % concern threshold; the corresponding DESeq2 RLE size factors agreed within 5 %, confirming the normalisation is robust to method choice." Quote the range of normalisation factors; extreme values signal compositional anomalies.

12.34.8 Practical Tips

- TMM and RLE adjust for compositional differences; CPM does not. For statistical comparison, always use TMM or RLE; CPM is only appropriate for visualisation.
- Extreme normalisation factors (above 2 or below 0.5) warrant investigation; they often indicate a few very highly expressed genes dominating a sample.
- For single-cell RNA-seq, different normalisation methods apply (scran's pooling-based size factors, sctransform's regularised regression); bulk methods underperform on the sparser, lower-coverage single-cell data.
- Normalisation factors enter the downstream GLM as an offset: `log(\hat\mu_{ij})=\log(\mathrm{NF}_j\cdot\ell_j)+X_i^\top\beta`. The DE

method handles this automatically.
- TMM with `refColumn` set to a specific reference sample is useful when one sample is known to be representative; the default chooses an automatic reference.
- For mixed-tissue or very heterogeneous experiments, the "majority unchanged" assumption can fail; consider quantile normalisation (`limma::normalizeQuantiles`) or condition-aware alternatives.

12.34.9 R Packages Used

`edgeR::calcNormFactors()` for TMM, upper-quartile, and RLE alternatives; `DESeq2::estimateSizeFactors()` for the canonical RLE; `limma::cpm()` and `voom()` for normalisation in the limma framework; `scran` and `sctransform` for single-cell normalisation.

12.34.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.34.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.35 Pseudoalignment with Salmon and Kallisto

12.35.1 Introduction

Pseudo-alignment tools — Salmon, Kallisto, and the related selective-alignment variant in newer Salmon versions — quantify transcript-level expression without producing full BAM-file alignments. They match read k-mers to transcript fingerprints in an indexed reference and resolve multi-mappers via expectation-maximisation. The result is a 10-to-100-fold speedup over STAR or HISAT2 with comparable accuracy at the gene level, making pseudo-alignment the standard fast-path for RNA-seq quantification when isoform-level alignment is not the goal.

12.35.2 Prerequisites

A working understanding of RNA-seq, the difference between transcript-level and gene-level quantification, and basic familiarity with FASTQ inputs and gene-annotation formats.

12.35.3 Theory

Pseudo-aligners build an index of all transcripts in the reference, splitting them into k-mers (typically $k = 31$). At quantification time, each read is broken into k-mers, the compatible transcripts are identified, and an expectation-maximisation algorithm distributes ambiguous reads across compatible transcripts in proportion to expected expression. The output is per-transcript counts and TPM; gene-level aggregation uses a transcript-to-gene map via `tximport` or `tximeta`.

Salmon adds bias-correction flags (`--gcBias`, `--seqBias`, `--posBias`) that calibrate counts for GC content, hexamer priming bias, and positional sequencing bias. Decoy-aware indexing (using the genome as a decoy alongside the transcriptome) reduces spurious alignment to paralogues and improves accuracy for difficult regions.

12.35.4 Assumptions

A transcript annotation matched to the genome; polyA-selected or otherwise enriched mRNA library; reads predominantly map to known transcripts (pseudo-alignment cannot detect novel transcripts or junctions).

12.35.5 R Implementation

```
library(tximport); library(tximeta)

# Salmon output: quant.sf files per sample
# files <- c("s1/quant.sf", "s2/quant.sf")
# tx_map <- read.table("tx2gene.tsv", header = TRUE)
#
# txi <- tximport(files, type = "salmon", tx2gene = tx_map)
# dim(txi$counts)
#
# library(DESeq2)
# dds <- DESeqDataSetFromTximport(txi, colData = coldata, design = ~
 ↪  condition)
```

12.35.6 Output & Results

Salmon and Kallisto produce per-sample `quant.sf` (Salmon) or `abundance.tsv` (Kallisto) files containing transcript-level counts, TPM, and effective length. `tximport()` aggregates these to gene level with optional length-scaled TPM correction; the resulting matrix flows directly into DESeq2, edgeR, or limma-voom.

12.35.7 Interpretation

A reporting sentence: "Salmon (v1.10) with selective alignment, decoy-aware indexing, and `--gcBias--seqBias` flags quantified 45{,}000 transcripts per sample in approximately 3 minutes per sample (vs. 30 minutes for STAR); tximport aggregation yielded 22{,}000 protein-coding genes for DESeq2 analysis." Always state the bias-correction flags and the decoy strategy.

12.35.8 Practical Tips

- Use decoy-aware indexing (Salmon) by including the genome as a decoy alongside the transcriptome; this reduces spurious mapping to paralogous regions.
- Enable bias-correction flags (`--gcBias`, `--seqBias`) routinely; they improve accuracy without meaningful runtime cost.
- For transcript-level differential expression with uncertainty, use bootstrap replicates (Salmon's `--numBootstraps` or Kallisto's `bootstrap-samples`) and analyse with sleuth or swish (`fishpond`).
- `tximeta` extends `tximport` with automatic transcript-to-gene mapping based on the Salmon index's metadata; use it to avoid annotation-mismatch errors.
- Pseudo-alignment is unsuitable for variant calling, novel-junction detection, or splicing analysis; use STAR with the genome reference for these tasks.
- For single-cell RNA-seq, use alevin (in Salmon) or kallisto-bustools; they extend pseudo-alignment to the cell-barcode and UMI structure of droplet-based single-cell data.

12.35.9 R Packages Used

`tximport` for transcript-to-gene aggregation; `tximeta` for metadata-aware aggregation; `fishpond::swish()` for transcript-level DTE with bootstrap inferential replicates; `DRIMSeq` and `DEXSeq` for differential transcript usage. Salmon and Kallisto are command-line tools; their output integrates with all standard Bioconductor RNA-seq workflows.

12.35.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.35.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.36 Cell-Type Annotation

12.36.1 Introduction

After clustering, each cluster must be labelled with a biological identity: T cell, B cell, macrophage, tumour epithelial cell, neuron subtype. Cell-type annotation is the bridge between unsupervised clustering and biological interpretation. Two main approaches compete: reference-based methods (SingleR, Azimuth, scType) that score each cell or cluster against curated reference profiles, and manual annotation that inspects marker genes against literature and reference atlases.

12.36.2 Prerequisites

A working understanding of clustered scRNA-seq output, marker-gene analysis, and the available cell-type reference atlases (Human Cell Atlas, Tabula Muris, Monaco Immune, ENCODE).

12.36.3 Theory

SingleR computes per-cell Spearman correlations against each reference profile over reference-defined marker genes; cells are labelled with the highest-scoring reference label. A pruning step removes calls below a confidence threshold. The method is fast, automated, and produces both per-cell and per-cluster summaries.

Azimuth uses a transfer-learning approach: it projects query cells into the reference's latent space (typically a PCA + harmony-corrected representation) and transfers labels via nearest-neighbour matching. It is the modern Seurat-team-recommended approach when a curated reference atlas exists for the tissue.

Manual annotation inspects each cluster's top marker genes against curated marker lists (canonical T-cell markers: CD3D, CD3E; B cells: CD19, MS4A1; macrophages: CD68, CD163). Cluster-by-cluster manual annotation is essential when references are missing or the tissue is non-standard.

12.36.4 Assumptions

The reference atlas matches the tissue and species of interest; cell types present in the data are represented in the reference (rare or unusual states may be absent and require manual annotation).

12.36.5 R Implementation

```
library(SingleR); library(celldex)

# Reference: built-in Monaco immune cells
ref <- MonacoImmuneData()

# Toy query: simulate a cell-by-gene expression matrix
set.seed(2026)
query <- matrix(rpois(200 * 100, lambda = 2), 200, 100)
rownames(query) <- sample(rownames(ref), 200)   # match gene names
colnames(query) <- paste0("c", 1:100)

pred <- SingleR(test = query, ref = ref,
                labels = ref$label.main,
                de.method = "wilcox")

table(pred$labels)
head(pred[, c("labels", "pruned.labels")])
```

12.36.6 Output & Results

`SingleR()` returns per-cell predicted labels, pruned labels (high-confidence subset), and the underlying per-reference-label scores. Summary tables show predicted cell-type counts; `pruneScores()` allows fine-tuning of the confidence threshold.

12.36.7 Interpretation

A reporting sentence: "SingleR with the Monaco Immune reference assigned 62 % of cells to canonical PBMC types (T cells, B cells, NK cells, monocytes, dendritic cells); pruning removed 8 % low-confidence calls. The remaining 30 % unannotated cells were tumour-associated stromal cells confirmed manually using ACTA2, COL1A1, and CD90 markers." Always report the reference, pruning rate, and manual validation of automated calls.

12.36.8 Practical Tips

- Use a reference matching the tissue source — never annotate skin cells against a blood reference; the cell types simply do not match.

- Report pruned labels rather than raw labels; pruned labels exclude low-confidence calls that often reflect intermediate or unusual states.
- For novel tissues or rare states, manual annotation outperforms reference-based methods; the references cannot label what they have not seen.
- Azimuth (`SeuratDisk::LoadH5Seurat` + `MapQuery`) is the modern Seurat alternative; it handles batch differences between query and reference more gracefully than SingleR.
- Cross-validate automated cluster labels against marker gene expression; a cluster labelled "T cell" by SingleR but without CD3D/CD3E expression is suspect and warrants investigation.
- For very granular reference atlases (Human Cell Atlas with 100+ cell types), use cluster-level SingleR rather than per-cell to reduce noise.

12.36.9 R Packages Used

`SingleR` for the canonical reference-based annotation; `celldex` for built-in reference datasets (Monaco, Encode, Hematopoietic Cell Atlas, etc.); `Azimuth` for Seurat-flavoured transfer learning; `scType` for marker-database-based annotation; `clustifyr` for an alternative reference-based approach.

12.36.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.36.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.37 scRNA Clustering

12.37.1 Introduction

Single-cell RNA-seq clustering groups cells into putative cell types or transcriptional states based on transcriptome similarity. The dominant approach is graph-based community detection: construct a shared-nearest-neighbour (SNN) graph of cells in PCA space, then apply the Louvain or Leiden algorithm to identify densely connected communities. The approach scales to millions of cells, produces interpretable cluster boundaries, and is the standard in both Seurat and Bioconductor scRNA-seq workflows.

12.37.2 Prerequisites

A working understanding of normalised scRNA-seq counts, dimensionality reduction (PCA), and the concept of community detection in graphs.

12.37.3 Theory

The standard pipeline:

1. **K-nearest-neighbour graph** in reduced PCA space: each cell is connected to its k nearest neighbours (typically $k = 20$).
2. **Shared-nearest-neighbour (SNN) graph**: edge weights between cell pairs are determined by the number of shared nearest neighbours, emphasising robust local structure over noisy individual neighbour relationships.
3. **Community detection**: Louvain optimises modularity; Leiden refines Louvain to guarantee well-connected communities and has become the modern default.

The `resolution` parameter controls granularity: higher resolution yields more, smaller clusters; lower resolution yields fewer, larger ones. There is no universally correct resolution — the choice depends on the biological question and is typically validated by marker-gene analysis.

12.37.4 Assumptions

Cell types or states form reasonably discrete communities in PCA space; the SNN graph captures local biological structure; the chosen number of PCs (typically 10–50) captures the major biological variation.

12.37.5 R Implementation

```
library(Seurat)

set.seed(2026)
counts <- matrix(rpois(300 * 200, lambda = 2), 200, 300)
rownames(counts) <- paste0("g", 1:200)
colnames(counts) <- paste0("c", 1:300)

so <- CreateSeuratObject(counts)
so <- NormalizeData(so, verbose = FALSE)
so <- FindVariableFeatures(so, nfeatures = 100, verbose = FALSE)
so <- ScaleData(so, verbose = FALSE)
so <- RunPCA(so, npcs = 10, verbose = FALSE)
so <- FindNeighbors(so, dims = 1:10, verbose = FALSE)

so <- FindClusters(so, resolution = 0.3, verbose = FALSE)
n_low <- nlevels(Idents(so))
```

```
so <- FindClusters(so, resolution = 1.2, verbose = FALSE)
n_high <- nlevels(Idents(so))

c(low_res = n_low, high_res = n_high)
```

12.37.6 Output & Results

`FindClusters()` returns the same `Seurat` object with cluster identities added to the active idents and to per-cell metadata. Different resolutions produce different cluster counts; the `clustree` package visualises how cells move between clusters as resolution changes.

12.37.7 Interpretation

A reporting sentence: "Leiden clustering on the SNN graph (`algorithm=4`) at resolution 0.5 produced 12 clusters spanning the major immune cell-type families; clusters were annotated by canonical marker genes (CD3 for T cells, CD19 for B cells, CD14 for monocytes) before downstream differential-expression analysis." Always describe the algorithm, resolution, and PCs used.

12.37.8 Practical Tips

- Explore multiple resolutions and visualise cluster relationships across resolutions with `clustree::clustree()`; cluster stability across nearby resolutions is reassuring.
- Use Leiden (`algorithm=4` in Seurat or `cluster_leiden()` in `igraph`) over Louvain — Leiden guarantees connected communities, which Louvain does not.
- Never cluster directly on UMAP coordinates; UMAP distorts global distances and clustering on it produces unstable, geometry-dependent results. Always cluster on PCA space.
- The "correct" number of clusters is a biological judgement, not a statistical one; use marker-gene analysis to confirm whether clusters correspond to known or plausible cell types.
- For very large datasets ($>$ 500,000 cells), use Seurat v5 sketch-based clustering or Bioconductor's `bluster` package with subsampling for tractable computation.
- Compare clustering to integration-aware methods (`harmony`, `Symphony`) when batch effects span multiple datasets; clustering on uncorrected data confounds batch with biology.

12.37.9 R Packages Used

`Seurat::FindClusters()` for graph-based clustering with Louvain/Leiden; `bluster` for Bioconductor-style clustering with multiple algorithms; `clustree` for cross-resolution stability visualisation; `igraph::cluster_leiden()` for the underlying Leiden implementation; `harmony` for batch correction before clustering.

12.37.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.37.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.38 Integration with Harmony

12.38.1 Introduction

Datasets from different batches, donors, or technologies cluster by batch rather than biology without correction. Integration methods align shared cell types across datasets while preserving dataset-specific populations. Harmony iteratively centres cell embeddings by batch; Seurat's CCA/RPCA anchors cells between batches.

12.38.2 Prerequisites

Multiple scRNA-seq datasets with shared cell populations; normalised expression; PCA embeddings.

12.38.3 Theory

Harmony alternates (i) soft cluster assignment in PCA space, (ii) per-cluster batch centroid computation, (iii) shifting cells within each cluster by its batch-specific correction vector. Converges in a few iterations and scales to millions of cells.

Seurat CCA/RPCA integration finds anchor pairs across datasets via mutual nearest neighbours in a shared low-dimensional space and uses anchors to transform embeddings.

12.38.4 Assumptions

Shared cell types span batches; batch effects are corrective in PCA space; biology and batch are not perfectly confounded.

12.38.5 R Implementation

```
library(harmony); library(Seurat)

set.seed(2026)
# Simulate two batches with shared biology + batch shift
n <- 150
batch <- rep(c("A", "B"), each = n)

counts <- matrix(rpois(300 * 200, lambda = 2), 200, 300)
rownames(counts) <- paste0("g", 1:200)
colnames(counts) <- paste0("c", 1:300)
counts[, batch == "B"] <- counts[, batch == "B"] + 3   # batch effect

so <- CreateSeuratObject(counts)
so$batch <- batch
so <- NormalizeData(so, verbose = FALSE)
so <- FindVariableFeatures(so, nfeatures = 100, verbose = FALSE)
so <- ScaleData(so, verbose = FALSE)
so <- RunPCA(so, npcs = 10, verbose = FALSE)

so <- RunHarmony(so, group.by.vars = "batch", dims.use = 1:10,
                 verbose = FALSE)
so <- RunUMAP(so, reduction = "harmony", dims = 1:10, verbose = FALSE)
so <- FindNeighbors(so, reduction = "harmony", dims = 1:10, verbose =
 ↪  FALSE)
so <- FindClusters(so, resolution = 0.3, verbose = FALSE)

table(Batch = so$batch, Cluster = Idents(so))
```

12.38.6 Output & Results

A Harmony-corrected embedding; after clustering, batches should be represented in each cluster (shared cell types) rather than segregating by batch.

12.38.7 Interpretation

"After Harmony integration, all seven major cell types contained cells from both donors (mixing LISI = 1.84 / 2.0), indicating successful batch correction without over-mixing."

12.38.8 Practical Tips

- Use Harmony for speed; CCA/RPCA for heavier anchor-based correction when batches have unique cell types.
- Quantify integration quality with kBET, LISI, or silhouette by batch vs cell type.
- Never integrate across tissues; integrate within tissue with donors/batches as covariates.
- Over-integration destroys biology; always inspect cell-type marker expression post-integration.
- Batch as a fixed effect in the DE model is an alternative to explicit embedding integration for focused comparisons.

12.38.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.38.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.39 Finding Marker Genes

12.39.1 Introduction

Marker genes are genes differentially expressed in one cluster relative to others; they drive cell-type annotation, functional interpretation, and the connection between unsupervised clustering and known biology. Seurat's `FindMarkers` and `FindAllMarkers` implement the standard differential-expression tests (Wilcoxon rank-sum, MAST hurdle, DESeq2) on log-normalised or SCT-transformed data, per cluster, against either all other cells or a specified comparison cluster.

12.39.2 Prerequisites

A working understanding of clustered scRNA-seq output, log-fold-change interpretation, and the difference between cluster-vs-all and pairwise cluster comparisons.

12.39.3 Theory

For each cluster, the marker test compares cells in that cluster to cells in all other clusters (or a specified subset) gene-by-gene. The output for each gene includes:

- **avg_log2FC**: average log-fold change between the target cluster and the reference.
- **pct.1**: fraction of cells in the target cluster expressing the gene.
- **pct.2**: fraction of cells in the reference expressing the gene.
- **p_val_adj**: Benjamini-Hochberg adjusted p-value.

Wilcoxon rank-sum is the default test in Seurat (fast, robust); MAST handles zero-inflation more rigorously via a hurdle model; DESeq2 applies the negative-binomial framework (slow but principled).

Conventional marker criteria: $\log_2 \mathrm{FC} > 0.5$, adjusted $p < 0.05$, and `pct.1>0.25` (the gene is expressed in at least a quarter of the cluster's cells).

12.39.4 Assumptions

Cells within a cluster are reasonably homogeneous; the reference set is a valid comparison; multiple testing is corrected across genes within each comparison (but not across the joint cluster set, which would be over-conservative).

12.39.5 R Implementation

```
library(Seurat)

set.seed(2026)
counts <- matrix(rpois(300 * 200, lambda = 2), 200, 300)
rownames(counts) <- paste0("g", 1:200)
colnames(counts) <- paste0("c", 1:300)
# Inject a marker pattern in cells 1-50
counts[1:10, 1:50] <- counts[1:10, 1:50] + 10

so <- CreateSeuratObject(counts)
so <- NormalizeData(so, verbose = FALSE)
so <- FindVariableFeatures(so, nfeatures = 100, verbose = FALSE)
so <- ScaleData(so, verbose = FALSE)
so <- RunPCA(so, npcs = 10, verbose = FALSE)
so <- FindNeighbors(so, dims = 1:10, verbose = FALSE)
so <- FindClusters(so, resolution = 0.5, verbose = FALSE)

markers_all <- FindAllMarkers(so, only.pos = TRUE, min.pct = 0.25,
                              logfc.threshold = 0.25, verbose = FALSE)
head(markers_all[, c("cluster", "gene", "avg_log2FC", "p_val_adj",
 ↪  "pct.1")])
```

12.39.6 Output & Results

`FindAllMarkers()` returns a data frame with one row per significant marker per cluster, including cluster identity, gene name, log-fold change, adjusted p-value, and percentage of cells expressing in target vs reference. Filtering to `only.pos=TRUE` retains only up-regulated markers, which are typically the most interpretable for cell-type identification.

12.39.7 Interpretation

A reporting sentence: "FindAllMarkers (Wilcoxon test, `min.pct=0.25`, `logfc.threshold=0.25`, `only.pos=TRUE`) identified 1{,}240 unique markers across 12 clusters; cluster 2's top markers (CD8A, GZMB, PRF1) identify cytotoxic T cells, and cluster 5's markers (MS4A1, CD79A) identify B cells. Annotation followed manual review against the Human Cell Atlas reference." Always state the test, thresholds, and annotation source.

12.39.8 Practical Tips

- Use `only.pos=TRUE` to retain up-regulated markers; down-regulated markers are less interpretable in scRNA-seq because most genes are dropouts in any given cluster.
- Set `min.pct=0.25` to filter out genes with very limited expression in the target cluster; this is a quality gate that prevents spurious markers from low-detection genes.
- For publication-quality marker calls, use MAST (`test.use="MAST"`); it handles zero-inflation more rigorously than Wilcoxon and gives more conservative p-values.
- Always annotate clusters using markers *after* clustering — never adjust clusters to fit pre-conceived markers; the latter is circular reasoning.
- Do not perform DE testing between clusters produced by the same clustering on the same data; this is a form of double-dipping that inflates significance. Use post-hoc validation or independent test data instead.
- For automated cell-type annotation, pair markers with reference-based tools (`SingleR`, `scType`, `Azimuth`); they reduce manual annotation effort substantially.

12.39.9 R Packages Used

`Seurat::FindMarkers()` and `FindAllMarkers()` for the canonical workflow; `MAST` for hurdle-based DE testing; `DESeq2` for negative-binomial DE on pseudobulk; `SingleR` and `Azimuth` for reference-based automated annotation; `presto` for fast Wilcoxon implementations on large datasets.

12.39.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.39.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.40 scRNA Normalisation

12.40.1 Introduction

Single-cell RNA-seq cells differ in total UMI count by orders of magnitude — a typical 10x Chromium experiment ranges from a few hundred to tens of thousands of UMIs per cell. This library-size variation swamps biological signal if not corrected. Normalisation removes the library-size effect, and variance stabilisation goes further by flattening the mean-variance relationship so that downstream PCA and clustering are driven by biology rather than count magnitude.

12.40.2 Prerequisites

A working understanding of count data, library-size effects, and the negative-binomial distribution as the standard model for count overdispersion.

12.40.3 Theory

Three approaches dominate:

LogNormalize (Seurat default): divide each cell's count by its total, multiply by 10{,}000, take $\log_{1p}$. Simple, fast, and well understood; does not stabilise variance.

SCTransform: regularised negative-binomial regression with sequencing depth as the explanatory variable, producing Pearson residuals as the normalised output. Variance is stabilised by construction, and the regularised parameter pooling allows rare cell types to be recovered better than LogNormalize.

Pool-based size factors (`scran::computeSumFactors`): pool cells for robust size-factor estimation in low-depth settings, addressing the dropout problem that simple library-size normalisation does not.

12.40.4 Assumptions

Library-size effects are multiplicative on count expectations; counts follow a (possibly cell-specific) negative-binomial distribution with gene-specific dispersion.

12.40.5 R Implementation

```r
library(Seurat)

set.seed(2026)
counts <- matrix(rpois(300 * 200, lambda = 2), 200, 300)
rownames(counts) <- paste0("g", 1:200)
colnames(counts) <- paste0("c", 1:300)

so <- CreateSeuratObject(counts)

# Standard LogNormalize
so <- NormalizeData(so, verbose = FALSE)
so <- FindVariableFeatures(so, nfeatures = 100, verbose = FALSE)
so <- ScaleData(so, verbose = FALSE)

# Alternative: SCTransform
so_sct <- SCTransform(so, verbose = FALSE)

head(GetAssayData(so, assay = "RNA", slot = "data")[1:5, 1:4])
head(GetAssayData(so_sct, assay = "SCT", slot = "data")[1:5, 1:4])
```

12.40.6 Output & Results

Two normalised matrices: LogNormalize (log-counts-per-10k stored in `data` slot of the `RNA` assay) and SCT (Pearson residuals stored in `data` slot of the `SCT` assay). Both feed the same downstream pipeline (PCA, clustering, UMAP) but produce different clustering granularity in practice.

12.40.7 Interpretation

A reporting sentence: "After SCTransform normalisation (`vst.flavor='v2'`, `vars.to.regress='percent.mt'`), PCA separated the major immune cell types in the first 5 components; LogNormalize required 10 components to achieve comparable separation. The SCT-based clustering identified two additional rare populations (mast cells, plasmacytoid dendritic cells) that LogNormalize merged into larger clusters." Always state the normalisation method and any regressed-out covariates.

12.40.8 Practical Tips

- Use `SCTransform(vst.flavor="v2")`, the current recommended flavour; it improves on the original SCTransform parameter regularisation.
- For rare cell types, SCTransform consistently outperforms LogNormalize; for coarse-grained analyses either method works.
- Store raw counts in the `RNA` assay and SCT residuals in the `SCT` assay; avoid re-running LogNormalize after SCTransform on the same object.
- Do not run `NormalizeData()` after `SCTransform()`; the latter already produces normalised values, and stacking is incorrect.
- For multi-sample integration, normalise each sample independently before merging (`SCTransform` per sample, then `IntegrateData` or `harmony`); pooling raw counts then normalising loses sample-specific structure.
- Regress out unwanted technical covariates (mitochondrial percentage, cell-cycle scores) via `vars.to.regress` in `SCTransform`; this often improves downstream clustering.

12.40.9 R Packages Used

`Seurat::NormalizeData()` and `SCTransform()` for the canonical scRNA-seq normalisation methods; `scran::computeSumFactors()` for pool-based size factors in the Bioconductor ecosystem; `sctransform` (the underlying engine) for low-level access to the regression model; `scater` for additional QC and normalisation diagnostics.

12.40.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.40.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.41 scRNA QC and Filtering

12.41.1 Introduction

Single-cell RNA-seq experiments routinely contain empty droplets, stressed or apoptotic cells, doublets (two cells captured in one droplet), and various

technical artefacts that should not enter downstream clustering or differential-expression analysis. Per-cell QC filtering by library size, detected-gene count, and mitochondrial-read percentage removes the worst offenders. The trade-off is delicate: over-filtering excludes real biology (e.g. plasma cells with naturally low gene counts), while under-filtering produces clusters of damaged cells that mimic real cell types.

12.41.2 Prerequisites

A working understanding of scRNA-seq counts, the structure of droplet-based experiments, and the role of mitochondrial reads as a cell-health indicator.

12.41.3 Theory

Three canonical per-cell metrics dominate QC:

- **nCount_RNA** (total UMI count per cell): very low values indicate empty droplets or debris; very high values often indicate doublets.
- **nFeature_RNA** (number of detected genes): the lowest quantile typically corresponds to broken or empty droplets.
- **percent.mt** (percentage of reads mapping to mitochondrial genes): elevated values indicate apoptotic or stressed cells whose cytoplasmic mRNA has been degraded preferentially.

Thresholds are dataset-specific; fixed cutoffs (`percent.mt<15` for 10x human samples) work for typical experiments, but adaptive thresholds based on the median absolute deviation (`scater::isOutlier`) generalise better across datasets.

For doublet detection, dedicated tools (`scDblFinder`, `DoubletFinder`) provide more reliable identification than naive UMI-based heuristics.

12.41.4 Assumptions

Mitochondrial percentage is a valid proxy for cell health (fails in some tissue types — cardiomyocytes naturally have very high mitochondrial content); doublets show abnormally high UMI counts on average; the chosen filtering thresholds preserve biological diversity.

12.41.5 R Implementation

```
library(Seurat)

set.seed(2026)
counts <- matrix(rpois(300 * 200, lambda = 2), 200, 300)
rownames(counts) <- c(paste0("MT-", 1:10),
                      paste0("gene_", 11:200))
colnames(counts) <- paste0("cell_", 1:300)
```

```
counts[1:10, 1:30] <- counts[1:10, 1:30] + 20   # 30 cells with high MT

so <- CreateSeuratObject(counts)
so[["percent.mt"]] <- PercentageFeatureSet(so, pattern = "^MT-")

VlnPlot(so, c("nFeature_RNA", "nCount_RNA", "percent.mt"),
        ncol = 3, pt.size = 0.1)

so_f <- subset(so,
               subset = nFeature_RNA > 50 &
                        nFeature_RNA < 180 &
                        percent.mt   < 15)
dim(so_f)       # cells x genes after QC
```

12.41.6 Output & Results

`PercentageFeatureSet()` adds the mitochondrial percentage to per-cell metadata; `VlnPlot()` shows the distribution of QC metrics across cells. `subset()` applies thresholds and returns a filtered Seurat object. Inspecting the violin plots before and after filtering confirms that the chosen thresholds remove the tails without truncating the bulk of the distribution.

12.41.7 Interpretation

A reporting sentence: "Per-cell QC removed 4{,}166 of 12{,}400 cells (33.6 %) that fell outside the chosen thresholds (nFeature $<$ 500, nFeature $>$ 6{,}000, percent.mt $>$ 15); doublet detection with `scDblFinder` flagged an additional 312 cells (3.8 %), yielding 7{,}922 high-quality cells for clustering. Thresholds were chosen by inspecting violin plots per sample." Always state thresholds and the rationale.

12.41.8 Practical Tips

- Inspect QC metrics per sample or batch separately before pooling; differing distributions reflect technical variation that should inform sample-specific thresholds.
- Use median-absolute-deviation-based thresholds (`scater::isOutlier(metric,nmads=3)`) for reproducibility across datasets; fixed cutoffs are sample-specific.
- Run doublet detection (`scDblFinder` is the modern recommendation, `DoubletFinder` is the older popular alternative) separately from count-based QC; doublets often pass naïve UMI thresholds.
- Remove cells failing on multiple metrics (low UMI *and* low gene count *and* high mitochondrial percentage) rather than aggressively filtering on any single metric; combinations are more reliable than thresholds on individual

metrics.
- Document QC parameters in the methods section; results are highly sensitive to filter choice and reproducibility requires explicit thresholds.
- For tissues where mitochondrial percentage is biologically meaningful (heart, skeletal muscle), adjust thresholds accordingly or use alternative QC metrics.

12.41.9 R Packages Used

`Seurat::PercentageFeatureSet()`, `VlnPlot()`, `subset()` for the standard QC workflow; `scater::isOutlier()` for adaptive MAD-based thresholds; `scDblFinder` and `DoubletFinder` for doublet detection; `emptyDrops` (in `DropletUtils`) for distinguishing real cells from empty droplets at the upstream step.

12.41.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.41.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.42 Trajectory Analysis

12.42.1 Introduction

Trajectory inference orders single cells along a continuous developmental or activation path, producing a pseudotime value per cell that represents its progress along the trajectory. Biological processes such as cell differentiation, cell-cycle progression, T-cell activation, and disease progression are continuous in nature; clustering imposes discrete categories that conceal the underlying continuous structure. Trajectory methods recover that continuity by fitting smooth curves or graphs through the data in dimensional-reduced space.

12.42.2 Prerequisites

A working understanding of scRNA-seq normalisation, dimensional reduction (PCA, UMAP), clustering, and the conceptual difference between discrete cell types and continuous developmental trajectories.

12.42.3 Theory

The two dominant trajectory frameworks:

Slingshot constructs a minimum spanning tree on cluster centroids in a reduced-dimensional space (typically PCA), specifies or infers a starting cluster, and fits smooth principal curves through each lineage from the start. Pseudotime is the arc length along each curve, computed per cell.

Monocle3 uses a reverse-graph-embedding approach on UMAP coordinates, fitting a principal graph and ordering cells along it. Monocle3 handles complex branching topologies more naturally than Slingshot.

Both methods require either a specified root cluster or an inferred one (with biological-marker-based validation strongly recommended). Pseudotime values are then used as a continuous covariate in downstream gene-by-pseudotime models (`tradeSeq`, generalised additive models) to identify genes whose expression varies along the trajectory.

12.42.4 Assumptions

Cells lie on a low-dimensional biological manifold; the cluster topology approximates the trajectory topology; the chosen root cluster represents the developmental starting state.

12.42.5 R Implementation

```
library(slingshot); library(SingleCellExperiment)

set.seed(2026)
n_cells <- 200
x <- seq(0, 1, length.out = n_cells)
pc1 <- x + rnorm(n_cells, 0, 0.05)
pc2 <- x^2 + rnorm(n_cells, 0, 0.05)

sce <- SingleCellExperiment(
  assays = list(counts = matrix(rpois(n_cells * 50, 3), 50, n_cells))
)
reducedDim(sce, "PCA") <- cbind(pc1, pc2)
sce$cluster <- factor(cut(x, breaks = 4, labels = 1:4))

sce <- slingshot(sce, clusterLabels = sce$cluster,
                 reducedDim = "PCA", start.clus = "1")
head(slingPseudotime(sce))

plot(reducedDim(sce, "PCA"), col = sce$cluster, pch = 16, asp = 1,
     main = "Slingshot lineage")
lines(SlingshotDataSet(sce), lwd = 2, col = "black")
```

12.42.6 Output & Results

`slingshot()` returns the `SingleCellExperiment` object augmented with pseudotime values (one column per lineage). The plot overlays smooth principal curves on the PCA scatter, with cells coloured by cluster, making the inferred lineage structure visible.

12.42.7 Interpretation

A reporting sentence: "Slingshot trajectory inference on the cluster-resolved PCA embedding recovered a single lineage from cluster 1 (root, validated as quiescent stem cells by SOX2 expression) through clusters 2 and 3 to cluster 4 (terminally activated cells); pseudotime correlated strongly with the known maturation marker BMI1 ($\rho = 0.84$, $p < 0.001$)." Always validate trajectory direction with known biology.

12.42.8 Practical Tips

- Specify `start.clus` explicitly when biology supports a clear initial state; automatic root inference is often unreliable for complex datasets.
- Compute trajectories on PCA space, not UMAP; UMAP distorts distances and produces unstable trajectory inference.
- Test genes for differential expression along pseudotime with `tradeSeq` (generalised additive models on count data) for principled identification of trajectory-driven genes.
- Monocle3 is preferred when the trajectory has complex branching structure (multiple lineages diverging); Slingshot is simpler and works well for linear or weakly branching trajectories.
- Always validate the inferred direction against known biological markers (start-state vs end-state); pseudotime is direction-less without a defined root.
- Pair trajectory analysis with cluster annotation; the lineage structure should connect biologically meaningful states, not arbitrary clusters.

12.42.9 R Packages Used

`slingshot` for principal-curve-based trajectory inference; `monocle3` for graph-based trajectory inference with branching support; `tradeSeq` for differential gene expression along pseudotime; `condiments` for trajectory testing under multiple conditions; `dyno` for unified access to dozens of trajectory inference methods.

12.42.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.42.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.43 Sequence Alignment: Overview

12.43.1 Introduction

Sequence alignment is the foundational operation of bioinformatics: comparing DNA, RNA, or protein sequences to find similarities, detect mutations, identify conserved regions, or annotate functional domains. The two canonical algorithms are Needleman-Wunsch (global, full-length alignment) and Smith-Waterman (local, best-matching subregion). All modern aligners — BWA, Bowtie, BLAST, minimap2 — are heuristic approximations to these dynamic-programming gold standards, optimised for speed at the cost of theoretical optimality.

12.43.2 Prerequisites

A working understanding of DNA and protein sequences, the FASTA format, and basic dynamic programming as the algorithmic technique behind alignment.

12.43.3 Theory

Global alignment (Needleman-Wunsch) finds the optimal alignment across the entire length of both sequences. Appropriate when the sequences are believed to be homologous over their full length, e.g. comparing two complete protein sequences from related species.

Local alignment (Smith-Waterman) finds the optimal alignment of a substring of each sequence. Appropriate when only a region is conserved, e.g. searching for a domain match within a longer protein.

Scoring schemes combine match rewards, mismatch penalties, and gap penalties. For proteins, substitution matrices encode biological similarity:

- **BLOSUM62**: derived from blocks of distantly related proteins; the BLAST default.
- **PAM250**: derived from closely related proteins; appropriate for close evolutionary distances.

Affine gap penalties (gap-opening + gap-extension) better model real biological gaps than constant penalties.

12.43.4 Assumptions

Sequences are biologically comparable (same molecule type, similar functional category); the chosen scoring scheme reflects the evolutionary distance involved.

12.43.5 R Implementation

```
library(Biostrings)

s1 <- DNAString("ATCGATCGTACGTACG")
s2 <- DNAString("ATCAATCATACGTGG")

# Global alignment
ga <- pairwiseAlignment(s1, s2, type = "global", substitutionMatrix =
 ↪  nucleotideSubstitutionMatrix(),
                        gapOpening = 10, gapExtension = 4)
ga

# Local alignment
la <- pairwiseAlignment(s1, s2, type = "local",
                        substitutionMatrix =
                         ↪  nucleotideSubstitutionMatrix(),
                        gapOpening = 10, gapExtension = 4)
la

# Protein alignment with BLOSUM62
p1 <- AAString("ACDEFGHI")
p2 <- AAString("ACDFGHI")
data(BLOSUM62)
pairwiseAlignment(p1, p2, substitutionMatrix = BLOSUM62, type = "global",
                  gapOpening = 10, gapExtension = 4)
```

12.43.6 Output & Results

`pairwiseAlignment()` returns an object with the aligned sequences (with gaps inserted), the alignment score, and percent identity. Subset accessors (`pattern()`, `subject()`, `score()`, `pid()`) extract specific components.

12.43.7 Interpretation

A reporting sentence: "Needleman-Wunsch global alignment of two 16-bp DNA sequences scored 6 (13/16 matches, two mismatches and a single-base gap); Smith-Waterman local alignment returned the best-matching 13-character substring with no gaps. The protein alignment with BLOSUM62 highlighted the conserved core (8/9 identity) despite the single deletion." Always state the scoring scheme and gap penalties.

12.43.8 Practical Tips

- Choose the scoring scheme to match the question: BLOSUM for distant protein relatives, PAM for close ones; for nucleotide alignment, simple match/mismatch is usually adequate.
- For short-read alignment (NGS), use BWA, Bowtie2, or minimap2 — they are heuristic but vastly faster than Needleman-Wunsch on millions of reads.
- For multiple sequence alignment (MSA), use the `msa` package which wraps Clustal Omega, ClustalW, and MUSCLE; or external tools (MAFFT, T-Coffee) for very large alignments.
- Affine gap penalties (gap-open and gap-extension) better model biological gaps than constant penalties; the BLAST default is gap-open 10, gap-extension 4 for nucleotides.
- For long reads (PacBio, Nanopore), use minimap2 — it is designed for the higher error rates and longer alignment lengths.
- For sequence search rather than pairwise comparison, BLAST and DIAMOND scale to large databases via heuristic seeding.

12.43.9 R Packages Used

`Biostrings` for `pairwiseAlignment()`, BLOSUM/PAM matrices, and sequence-handling classes; `msa` for multiple sequence alignment; `Rsamtools` and `GenomicAlignments` for BAM-aware alignment workflows; `Rsubread` for short-read alignment integrated with the broader RNA-seq Bioconductor workflow.

12.43.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.43.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR

- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.44 Single-Cell with Seurat

12.44.1 Introduction

Seurat is the most widely used R framework for single-cell RNA-seq (scRNA-seq) analysis. A standard Seurat workflow loads a count matrix from Cell Ranger or a similar source, performs quality control on per-cell metrics, normalises expression, selects variable features, scales, runs dimensionality reduction (PCA + UMAP/t-SNE), clusters cells, and identifies marker genes. The framework's strength is its end-to-end coverage and large user community; the alternative Bioconductor stack (`SingleCellExperiment` + `scran`) offers comparable functionality with tighter integration into the broader Bioconductor ecosystem.

12.44.2 Prerequisites

A working understanding of single-cell RNA-seq experimental design, count matrices from droplet-based protocols (10x Chromium), and basic clustering and dimensionality-reduction concepts.

12.44.3 Theory

Seurat stores cells × genes counts and derived matrices in a `Seurat` object with structured slots for raw counts, normalised data, variable features, scaled data, dimensional reductions, and per-cell metadata. The standard pipeline:

1. **QC**: filter cells by minimum gene count, maximum percent mitochondrial reads, library size thresholds.
2. **Normalisation**: log-normalisation (`NormalizeData`) or regularised regression (`SCTransform`).
3. **Variable-feature selection**: identify genes with biologically interesting variability.
4. **Scaling**: centre and unit-variance scale variable features.
5. **PCA**: linear dimensionality reduction on scaled variable features.
6. **Graph-based clustering**: shared-nearest-neighbour graph + Louvain or Leiden community detection.
7. **UMAP / t-SNE**: non-linear visualisation only — not used for clustering decisions.

12.44.4 Assumptions

Each droplet is mostly single-cell (low doublet rate, typically below 5 %); per-cell library size correlates with mRNA content; variable genes capture biological signal rather than technical noise.

12.44.5 R Implementation

```r
library(Seurat)

set.seed(2026)
# Minimal simulated count matrix (300 cells x 200 genes)
counts <- matrix(rpois(300 * 200, lambda = 2), 200, 300)
rownames(counts) <- paste0("gene_", 1:200)
colnames(counts) <- paste0("cell_", 1:300)

so <- CreateSeuratObject(counts = counts, min.cells = 3, min.features =
 ↪  10)
so <- NormalizeData(so, verbose = FALSE)
so <- FindVariableFeatures(so, nfeatures = 100, verbose = FALSE)
so <- ScaleData(so, verbose = FALSE)
so <- RunPCA(so, npcs = 10, verbose = FALSE)
so <- FindNeighbors(so, dims = 1:10, verbose = FALSE)
so <- FindClusters(so, resolution = 0.5, verbose = FALSE)
so <- RunUMAP(so, dims = 1:10, verbose = FALSE)

table(Idents(so))
DimPlot(so, reduction = "umap", label = TRUE)
```

12.44.6 Output & Results

A `Seurat` object containing per-cell cluster assignments, dimensional-reduction coordinates (PCA, UMAP), and gene-level statistics. `DimPlot()` produces the canonical UMAP visualisation with cluster boundaries; `FeaturePlot()` overlays gene expression on the UMAP.

12.44.7 Interpretation

A reporting sentence: "Standard Seurat processing of the 300-cell dataset (`NormalizeData`, `FindVariableFeatures`, `ScaleData`, `RunPCA`, Louvain clustering at resolution 0.5) produced 5 clusters; cluster 3 uniquely expressed CD8A and CD8B, consistent with cytotoxic T cells, and cluster 1 expressed CD79A and MS4A1, consistent with B cells." Always describe the QC thresholds, normalisation method, number of PCs, and clustering resolution.

12.44.8 Practical Tips

- Run QC before normalisation; low-quality cells dominate variable-feature selection and contaminate downstream clustering.
- Use `SCTransform` instead of `NormalizeData+FindVariableFeatures+ScaleData` for a more principled variance-stabilising approach; SCTransform is the modern default for new analyses.

- The clustering `resolution` controls cluster granularity; explore values from 0.2 to 1.2 and pick based on biological interpretability rather than a single statistical criterion.
- Document Seurat version with `sessionInfo()`; the API has changed substantially between major versions (v3, v4, v5), and old code may not run on current versions.
- For very large datasets ($>$ 100,000 cells), use Seurat v5's sketch-based subsampling or switch to the Bioconductor stack (`SingleCellExperiment`, `scran`) for memory efficiency.
- Pair clustering results with marker-gene analysis (`FindAllMarkers`) and biological annotation (SingleR, scType) before reporting cluster identities.

12.44.9 R Packages Used

`Seurat` for the canonical end-to-end scRNA-seq workflow; `SingleCellExperiment` for the Bioconductor data structure; `scran` and `scater` for Bioconductor-style normalisation and clustering; `harmony` for batch correction; `SingleR` for automatic cell-type annotation; `DoubletFinder` for doublet detection.

12.44.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.44.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.45 Spatial Transcriptomics

12.45.1 Introduction

Spatial transcriptomics assays (10x Visium, Slide-seq, Stereo-seq, MERFISH) measure gene expression with spatial coordinates. They bridge histology and scRNA-seq: the tissue architecture is preserved, so analyses like spatial domain detection, neighbourhood effects, and tumour-immune interface mapping become possible.

12.45.2 Prerequisites

scRNA-seq analysis basics; familiarity with tissue histology.

12.45.3 Theory

Visium spots contain ~10-100 cells each, giving spot-resolution (not single-cell) expression. Analysis proceeds like scRNA-seq but with two additions: (i) spatial variable-gene detection (Moran's I), and (ii) spatial domain inference (BayesSpace, Giotto, or spatial-aware clustering).

Deconvolution (cell2location, RCTD, SpatialExperiment) decomposes each spot into cell-type proportions using a matched scRNA-seq reference.

12.45.4 Assumptions

Spatial coordinates are correctly registered to H&E; spot-level expression is a linear combination of underlying cell types; tissue processing preserves mRNA integrity.

12.45.5 R Implementation

```
library(Seurat)

set.seed(2026)
# Minimal spatial-like object: spots on a grid
n_spots <- 100
coords <- expand.grid(x = 1:10, y = 1:10)
counts <- matrix(rpois(200 * n_spots, lambda = 3), 200, n_spots)
rownames(counts) <- paste0("g", 1:200)
colnames(counts) <- paste0("spot_", 1:n_spots)

# Inject spatial pattern: top-left vs bottom-right
region <- ifelse(coords$x + coords$y < 10, "A", "B")
counts[1:20, region == "A"] <- counts[1:20, region == "A"] + 8

so <- CreateSeuratObject(counts)
so$x <- coords$x; so$y <- coords$y; so$region <- region

so <- NormalizeData(so, verbose = FALSE)
so <- FindVariableFeatures(so, nfeatures = 100, verbose = FALSE)
so <- ScaleData(so, verbose = FALSE)
so <- RunPCA(so, npcs = 10, verbose = FALSE)
so <- FindNeighbors(so, dims = 1:10, verbose = FALSE)
so <- FindClusters(so, resolution = 0.5, verbose = FALSE)

plot(coords, col = factor(Idents(so)), pch = 15, cex = 2,
     main = "Spot clusters on tissue coordinates",
     xlab = "x", ylab = "y")
```

12.45.6 Output & Results

A clustered spatial object; the 2-D scatter shows clusters aligning with spatial regions, recapitulating tissue architecture.

12.45.7 Interpretation

"Spatial clustering of the Visium section identified six domains corresponding to tumour core, tumour edge, stroma, and three immune niches; interface enrichment analysis revealed T-cell accumulation at the tumour-stroma boundary."

12.45.8 Practical Tips

- Always overlay clusters on H&E to validate biological interpretation.
- Moran's I (`SpatialExperiment::spatialCorrelation` or `Seurat::FindSpatiallyVariableFeatures`) finds spatially variable genes.
- Deconvolution is essential when Visium spots cover multiple cell types; use a matched scRNA-seq reference.
- BayesSpace and SpaGCN integrate spatial structure into clustering, giving smoother domain boundaries.
- Stereo-seq and MERFISH offer sub-cellular resolution; analysis pipelines differ slightly from spot-based Visium.

12.45.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.45.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.46 STAR: Spliced Alignment

12.46.1 Introduction

STAR (Spliced Transcripts Alignment to a Reference) is the de facto standard short-read RNA-seq aligner. It is designed specifically for the splice-aware alignment problem: reads from mature mRNA cross exon-exon junctions, and a naive aligner that requires contiguous matches misses these split alignments and misclassifies them as multi-mappers or unaligned. STAR's seed-extend algorithm

splits reads at candidate junctions, aligns the segments to the reference genome, and assembles them into spliced alignments — including across novel (unannotated) junctions.

12.46.2 Prerequisites

A working understanding of RNA-seq experimental design, the difference between mRNA and DNA sequencing, and basic familiarity with sequence alignment concepts.

12.46.3 Theory

STAR's two-pass mapping is the standard mode: the first pass aligns each read independently and accumulates a list of candidate splice junctions; the second pass adds those junctions to the reference index and re-aligns, producing more accurate spliced alignments especially across novel junctions. The output is a BAM file plus a tab-separated splice-junction summary (`SJ.out.tab`).

STAR also supports chimeric alignment for fusion-gene detection (reads spanning two distinct genomic loci), expectation-maximisation-based multi-mapper resolution, and stranded alignment when library protocols preserve transcript orientation.

12.46.4 Assumptions

Short RNA-seq reads (typically 50–150 bp Illumina); a reference genome and (preferably) a gene annotation in GTF format; sufficient memory — STAR's index for human genome takes about 30 GB of RAM.

12.46.5 R Implementation

```
library(Rsubread)

# Rsubread::align is Subjunc (splice-aware) or align
# index <- "subread_index"
# buildindex(basename = index, reference = "genome.fa")

# Splice-aware alignment
# align(index = index, readfile1 = "reads.fastq.gz",
#       output_file = "aligned.bam",
#       type = "rna", useAnnotation = TRUE,
#       annot.inbuilt = "hg38")
```

12.46.6 Output & Results

The primary output is a BAM file containing spliced alignments and a `SJ.out.tab` file summarising splice junctions detected (including their support count and annotation status). For typical RNA-seq experiments, 90–95 % of reads align uniquely; a much lower rate signals contamination, library-prep problems, or annotation mismatch.

12.46.7 Interpretation

A reporting sentence: "STAR (v2.7.10a) two-pass alignment to GRCh38 with the GENCODE v44 annotation aligned 94 % of reads uniquely; 2.3 million splice junctions were detected, of which 98 % matched annotated junctions, suggesting limited novel splicing in the experimental conditions." Always report the genome assembly, annotation version, and unique-mapping rate.

12.46.8 Practical Tips

- STAR needs about 30 GB of RAM for the human genome index; for smaller genomes (yeast, bacteria), it runs comfortably on a laptop.
- Two-pass mode (`twopassMode="Basic"`) is standard for novel-junction detection; the first-pass-only mode is faster but less accurate.
- For transcript-level quantification, pair STAR alignments with RSEM, Salmon, or Kallisto; the BAM is the input to abundance estimation.
- HISAT2 is a faster, lower-memory alternative aligner with similar accuracy; use it when memory is constrained.
- Pseudo-aligners (Salmon, Kallisto) skip the BAM-producing alignment entirely; faster but less suitable when isoform-level alignments are needed (variant calling, fusion detection).
- Use a reference GTF annotation to improve junction accuracy; novel-junction detection works without it but is more error-prone.

12.46.9 R Packages Used

`Rsubread` provides splice-aware alignment via `subjunc` and feature counting via `featureCounts`; `Rsamtools` and `GenomicAlignments` for downstream BAM manipulation; `tximport` for importing transcript-level abundance estimates from Salmon, Kallisto, or RSEM into Bioconductor; STAR itself is a command-line tool but its output integrates with all standard R/Bioconductor RNA-seq workflows.

12.46.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

12.46.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.47 Structural Variant Detection

12.47.1 Introduction

Structural variants (SVs) – deletions, duplications, inversions, translocations, and complex rearrangements – are genomic events $\geq$ 50 bp. They often have larger phenotypic impact than single-nucleotide variants but are harder to call because they involve breakpoints across long distances. Short-read callers like Manta and Delly, plus long-read callers (Sniffles, pbsv), are standard.

12.47.2 Prerequisites

BAM files from high-coverage WGS; reference genome.

12.47.3 Theory

Short-read SV callers combine three signals: - **Read pairs** with abnormal insert size or orientation. - **Split reads** whose two halves map to distant loci. - **Read-depth** anomalies (CNV overlap).

Long-read callers directly observe breakpoint-spanning reads, dramatically improving resolution of repeat-rich SVs.

12.47.4 Assumptions

Sequencing depth is adequate ($\geq$ 30x WGS); insert-size distribution is approximately normal; reference matches the sample ancestry.

12.47.5 R Implementation

Manta/Delly run on the command line producing VCF; R ingests and annotates:

```
library(VariantAnnotation)
library(StructuralVariantAnnotation)

# Use a VariantAnnotation-shipped VCF as a proxy
vcf_file <- system.file("extdata", "chr22.vcf.gz", package =
 ↪  "VariantAnnotation")
vcf <- readVcf(vcf_file, "hg19")
```

```r
vcf_sv <- vcf[1:10, ]   # illustrative subset; treat as SV records

# breakpointRanges builds a GRanges of breakpoints from SV VCF INFO fields
gr <- breakpointRanges(vcf_sv)
length(gr)
head(as.data.frame(gr))
```

Example command-line calls:

```
manta configManta.py --bam sample.bam --reference ref.fa --runDir
 ↪  manta_out
delly call -g ref.fa -o sample.bcf sample.bam
```

12.47.6 Output & Results

A VCF of SV calls with type (DEL, DUP, INV, INS, BND), coordinates, and supporting-read counts; annotation against genes yields per-gene SV impact.

12.47.7 Interpretation

"WGS SV calling identified 4,120 events (2,800 deletions, 1,100 duplications, 180 inversions, 40 translocations) per sample. The germline 17q12 deletion spans RCAD and was concordant with the clinical suspicion."

12.47.8 Practical Tips

- Use **two** callers and intersect (high-precision) or union (high-sensitivity); any single caller misses events.
- Filter by breakpoint support; single-read evidence is almost always false.
- Long-read WGS resolves complex rearrangements that short reads cannot; use for structural-disease phenotyping.
- Tumour SV calling requires matched normal to remove germline events; somatic callers: Manta + DELLY (cancer mode), GRIDSS.
- Annotate breakpoints with `StructuralVariantAnnotation` to identify gene fusions and breakpoint-disrupted regions.

12.47.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.47.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.48 Surrogate Variable Analysis (SVA)

12.48.1 Introduction

Surrogate variable analysis (SVA) estimates latent factors that capture unmodelled heterogeneity in high-dimensional expression data – unknown batch effects, sample-processing artefacts, or unmeasured biology correlated with technical variables. Including surrogate variables as covariates in downstream regression restores power and controls false positives where ComBat cannot apply (because the batch variable is itself unknown).

12.48.2 Prerequisites

Linear models on expression data, batch effects, ComBat.

12.48.3 Theory

SVA decomposes the residual expression matrix (after regressing out the primary model) into latent factors using a bootstrap-based singular value decomposition. The number of surrogate variables k is chosen by a permutation test (`num.sv`). Each surrogate is then added as a covariate in the downstream limma/edgeR model.

For RNA-seq count data, use `svaseq`, which applies SVA on a log-transformed scale suitable for negative-binomial outcomes.

12.48.4 Assumptions

The primary variable of interest is modelled correctly; surrogate variables are orthogonal to it; residual variation reflects technical or unmodelled factors.

12.48.5 R Implementation

```
library(sva); library(limma)

set.seed(2026)
n_genes <- 3000; n_samples <- 20
group   <- factor(rep(c("ctrl", "trt"), each = 10))
batch   <- factor(rep(c("A", "B"), times = 10))           # hidden
```

```
y <- matrix(rnorm(n_genes * n_samples), n_genes, n_samples)
y <- y + matrix(rep(as.numeric(batch) * 0.6, each = n_genes),
                n_genes, n_samples)

mod  <- model.matrix(~ group)
mod0 <- model.matrix(~ 1, data = data.frame(group))

n_sv <- num.sv(y, mod, method = "be")
svobj <- sva(y, mod, mod0, n.sv = n_sv)

mod_sv <- cbind(mod, svobj$sv)
fit <- lmFit(y, mod_sv); fit <- eBayes(fit)
topTable(fit, coef = 2, n = 5)
```

12.48.6 Output & Results

`svobj$sv` contains $n_s v$ surrogate variables per sample. Including them typically recovers differential expression that batch confounding otherwise hides.

12.48.7 Interpretation

"We estimated k surrogate variables using SVA and included them in the limma model, adjusting for unmodelled heterogeneity without needing to specify a batch variable."

12.48.8 Practical Tips

- Use `svaseq` for RNA-seq count data.
- The `num.sv(...,method="be")` Buja-Eyuboglu estimator is conservative; `"leek"` is more aggressive.
- Surrogates should not correlate with the variable of interest; `sva` orthogonalises them by construction.
- Always inspect what surrogates correlate with (library size, processing date) for interpretation.
- If the batch is known, prefer ComBat-seq or direct covariate inclusion; SVA is for hidden factors.

12.48.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.48.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.49 Transcript-to-Gene Summarisation

12.49.1 Introduction

Pseudo-alignment tools (Salmon, Kallisto) and EM-based quantifiers (RSEM) produce transcript-level (isoform-level) counts. Most differential-expression workflows operate at the gene level, so transcript counts must be aggregated to genes via a transcript-to-gene mapping derived from the annotation. The `tximport` package handles this aggregation cleanly and produces output ready for DESeq2, edgeR, or limma-voom; the related `tximeta` extends this with automatic metadata and reference-genome tracking.

12.49.2 Prerequisites

A working understanding of transcript-level vs gene-level expression quantification, RNA-seq annotation files (GTF), and the difference between counts, TPM, and effective length.

12.49.3 Theory

The simplest transcript-to-gene aggregation sums transcript counts within each gene: gene count $= \sum_{\text{transcripts} \in \text{gene}}$ transcript count. `tximport` adds two important refinements:

- **Length-scaled TPM** (`countsFromAbundance="lengthScaledTPM"`): combines transcript abundance and length information to produce gene-level counts that respect both isoform-resolution and gene-level interpretation.
- **Effective-length offsets**: passes effective-length information to downstream DE tools (DESeq2 with `tximeta::tximeta()` integration) for accurate normalisation.

Direct transcript-level DE (without aggregation) is supported by tools like swish (`fishpond`) and DRIMSeq for differential transcript usage; aggregation is the right choice when the question is about gene-level expression.

12.49.4 Assumptions

The transcript-to-gene map matches the annotation used for transcript quantification; sample-file ordering is consistent with metadata.

12.49.5 R Implementation

```
library(tximport); library(tximeta)

# Example: salmon outputs
# files <- list.files("salmon_out", pattern = "quant.sf", recursive =
 ↪  TRUE, full.names = TRUE)
# tx2gene <- read.table("tx2gene.tsv", header = TRUE)
#
# txi <- tximport(files, type = "salmon", tx2gene = tx2gene,
#                 countsFromAbundance = "lengthScaledTPM")
#
# txi$counts: gene x sample matrix
```

12.49.6 Output & Results

`tximport()` returns a list with `counts` (gene-by-sample matrix), `abundance` (gene-level TPM), and `length` (effective lengths used for downstream normalisation). The matrix is ready as input to `DESeq2::DESeqDataSetFromTximport()` or as a starting point for edgeR / limma workflows.

12.49.7 Interpretation

A reporting sentence: "tximport aggregated 190{,}000 transcript-level Salmon estimates to 22{,}000 gene-level counts using a GENCODE v44 transcript-to-gene map; `countsFromAbundance='lengthScaledTPM'` produced gene counts that DESeq2 then analysed with effective-length offsets passed via the `txi$length` matrix." Always document the aggregation method and tx-to-gene source.

12.49.8 Practical Tips

- `countsFromAbundance="lengthScaledTPM"` is the standard for DE analysis; it scales raw counts by transcript length while preserving the count interpretation.
- Use `tximeta` instead of raw `tximport` when possible — it automatically retrieves the transcript-to-gene map from the Salmon index's metadata, eliminating annotation-mismatch errors.
- For differential transcript usage (DTU), use DRIMSeq or DEXSeq on the transcript-level matrix; aggregation collapses isoform-level information.
- For differential transcript expression (DTE), use swish (`fishpond::swish`) on the transcript-level matrix with bootstrap inferential replicates from Salmon.
- Verify that sample order in `files` matches the rows of your metadata table; off-by-one errors here corrupt the entire downstream analysis silently.
- For very large studies, `tximport` reads files sequentially; parallelise with `BiocParallel` for substantial speedups.

12.49.9 R Packages Used

`tximport` for the canonical aggregation; `tximeta` for metadata-aware aggregation that handles tx-to-gene mapping automatically; `fishpond` for swish-based DTE with bootstrap replicates; `DRIMSeq` and `DEXSeq` for differential transcript usage at the transcript level.

12.49.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.49.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.50 Variant Calling with GATK

12.50.1 Introduction

Variant calling turns aligned reads into a list of single-nucleotide variants (SNVs) and small insertions/deletions (indels). GATK's HaplotypeCaller is the de facto standard: it locally reassembles regions with evidence of variation and emits probabilistic genotype calls in VCF format.

12.50.2 Prerequisites

Sequence alignment, BAM format, reference genome indexing.

12.50.3 Theory

HaplotypeCaller proceeds in four stages: define **active regions** (loci with potential variation), perform **local de novo assembly** of haplotypes, **align reads to haplotypes** via pair-HMM, and **genotype** using likelihoods. For cohorts, each sample is first emitted as a per-sample GVCF, then joint-genotyped with `GenotypeGVCFs` – this preserves reference evidence and scales linearly with samples.

Downstream filtering is done with variant-quality score recalibration (VQSR) for whole-genome data or hard filters for smaller studies.

12.50.4 Assumptions

Reads are duplicate-marked, base-quality-recalibrated (BQSR), and aligned to a standard reference; ploidy is specified.

12.50.5 R Implementation

The GATK workflow runs on the command line (Java-based); R interacts with VCF output via `VariantAnnotation`.

```
library(VariantAnnotation)

# Sample VCF from the package
vcf_file <- system.file("extdata", "chr22.vcf.gz", package =
 ↪  "VariantAnnotation")
vcf <- readVcf(vcf_file, "hg19")

vcf                                      # summary (SNPs, indels, samples)
head(rowRanges(vcf))                      # variant coordinates
head(info(vcf)[, c("AF", "DP")])          # population AF, depth
geno(vcf)$GT[1:5, 1:3]                    # genotype calls (sample x
 ↪  variant)
```

A representative GATK command (not run here):

```
gatk HaplotypeCaller -R ref.fa -I sample.bam -O sample.g.vcf.gz -ERC GVCF
gatk GenotypeGVCFs   -R ref.fa -V cohort.g.vcf.gz -O cohort.vcf.gz
```

12.50.6 Output & Results

A VCF / GVCF with per-variant coordinates, alleles, annotations, and per-sample genotypes, depths, and likelihoods.

12.50.7 Interpretation

"Joint genotyping with GATK HaplotypeCaller called 4.2M SNVs and 0.8M indels across the cohort; after VQSR with 99.9% sensitivity we retained 3.9M high-confidence SNVs."

12.50.8 Practical Tips

- Always BQSR before calling; skipping it inflates false positives, especially for low-frequency variants.
- Use GVCF workflow for multi-sample studies; `HaplotypeCaller` in direct VCF mode is only for single-sample work.
- `DeepVariant` is a competitive deep-learning alternative, producing similar quality without assembly.
- For somatic calls, use `Mutect2`, not HaplotypeCaller.

- Filter by depth, strand bias, and mapping quality (see `VariantFiltration`); many library artefacts look like rare variants.

12.50.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.50.10 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.51 VCF Manipulation

12.51.1 Introduction

VCF (Variant Call Format) is the standard tab-delimited format for representing genotype calls at genomic positions. Whether the data come from whole-genome sequencing, exome sequencing, genotyping arrays, or imputation, downstream analysis nearly always begins by reading, filtering, and subsetting a VCF. The `VariantAnnotation` Bioconductor package provides programmatic access in R; `bcftools` is the command-line workhorse for cohort-scale operations.

12.51.2 Prerequisites

A working understanding of VCF format basics (CHROM, POS, REF, ALT, FILTER, INFO, FORMAT, per-sample GT), the difference between phased and unphased genotypes, and the typical variant-level QC metrics (QUAL, depth, missingness, allele frequency).

12.51.3 Theory

A VCF row represents a variant at one genomic position; columns include INFO (variant-level annotations: allele frequency, depth, functional consequence) and per-sample FORMAT fields (genotype GT, depth DP, genotype quality GQ). Standard filtering steps:

- **PASS filter**: keep only variants flagged `PASS` by the calling pipeline; non-PASS variants are typically calibration artefacts.
- **Allele frequency**: minimum minor-allele frequency (MAF) threshold to retain common variants for association testing.

- **Depth and missingness**: minimum per-sample depth and maximum site-level missingness.
- **Hardy-Weinberg**: large deviations in unrelated controls indicate genotyping errors.
- **Multi-allelic normalisation**: split multi-allelic sites into bi-allelic rows and left-align indels (`bcftoolsnorm`).

12.51.4 Assumptions

VCF is bgzipped (`.vcf.gz`) and tabix-indexed for random access; reference build matches the calling pipeline; sample identifiers are consistent across cohorts when merging.

12.51.5 R Implementation

```
library(VariantAnnotation)

vcf_file <- system.file("extdata", "chr22.vcf.gz", package =
 ↪  "VariantAnnotation")

# Targeted reading (region + fields only; faster than full read)
rng <- GRanges("22", IRanges(50301422, width = 50000))
param <- ScanVcfParam(which = rng,
                      info = c("AF", "DP"),
                      geno = "GT")
vcf_sub <- readVcf(vcf_file, "hg19", param = param)
vcf_sub

# Filter by MAF
af <- unlist(info(vcf_sub)$AF)
af <- pmin(af, 1 - af)
vcf_common <- vcf_sub[af > 0.05, ]
length(vcf_common)

# Extract and reshape genotypes
gt <- geno(vcf_common)$GT
head(gt)
```

12.51.6 Output & Results

`readVcf()` returns a `VCF` Bioconductor object with structured slots for variant ranges, INFO fields, FORMAT fields, and sample information. Accessor functions (`info()`, `geno()`, `rowRanges()`) extract specific components, and indexing (`vcf[af>0.05,]`) filters by row.

12.51.7 Interpretation

A reporting sentence: "After filtering to PASS variants with MAF > 5 %, missingness < 5 %, and Hardy-Weinberg $p > 10^{-6}$ in 1{,}532 unrelated controls, we retained 412{,}380 high-quality common autosomal variants for association testing." Always describe the filtering pipeline in detail; cohort-comparison studies require precisely matched filtering.

12.51.8 Practical Tips

- For cohort-scale operations (millions of variants, thousands of samples), use `bcftools` at the command line; it is orders of magnitude faster than R-based reading and writing.
- Use `ScanVcfParam()` to read only the regions and fields you need; full WGS VCFs can be tens of GB and reading them entirely is wasteful.
- Filter by `FILTER=="PASS"` early; non-PASS variants accumulate in downstream analyses if not removed.
- Normalise variants with `bcftoolsnorm-m-any-freference.fa` before merging cohorts; multi-allelic representations differ across callers and produce silent merge errors.
- For association studies, export to PLINK or BGEN formats via `snpStats` or `plink2` for efficient large-scale regression.
- For functional annotation (consequence prediction, conservation scores), use `VariantAnnotation::predictCoding()`, `EnsemblVEP`, or `SnpEff`.

12.51.9 R Packages Used

`VariantAnnotation` for reading, writing, and accessor-based VCF manipulation; `Rsamtools` for tabix index handling; `snpStats` for PLINK-format export and SNP-level statistics; `bcftoolsR` and standalone `bcftools` for command-line operations; `vcfR` for an alternative interface focused on population genetics.

12.51.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.51.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.52 Volcano Plots

12.52.1 Introduction

A volcano plot summarises a differential-expression result in a single scatter: log fold change on the x-axis, $-\log_{10}$ adjusted p-value on the y-axis. Genes in the upper-left and upper-right corners — large effect *and* high precision — are the targets of interest; genes near the bottom of the plot fail the significance criterion regardless of effect size, and genes near zero on the x-axis fail the biological-relevance criterion regardless of significance. The combined visual makes the standard two-criterion ranking immediately apparent and is the canonical figure of any differential-expression report.

12.52.2 Prerequisites

A working understanding of differential-expression analysis output (DESeq2, edgeR, limma-voom), log fold change, and FDR-adjusted p-values.

12.52.3 Theory

The volcano plot's signature shape arises from the relationship between effect-size estimates and their standard errors: extreme fold changes are typically reported by genes with low expression (and therefore noisier estimates with worse p-values), so the "wings" of high fold change at low significance are sparse. The dense central column near zero fold change, with p-values uniformly distributed from very high to moderately significant, represents the bulk of unchanged genes.

Standard cutoffs combine fold-change and significance: $|\log_2 \mathrm{FC}| > 1$ (two-fold change) and adjusted $p < 0.05$ jointly. Either criterion alone is misleading — large effects with poor precision (e.g. low-expression genes) and small effects at extreme significance (e.g. very high-expression genes with tiny but reproducible changes) both produce different scientific conclusions.

12.52.4 Assumptions

The DE analysis was correctly specified; p-values are calibrated (DE methods generally produce well-calibrated p-values under their assumptions); fold-change sign is interpretable as a directional effect.

12.52.5 R Implementation

```
library(ggplot2)

set.seed(2026)
n <- 5000
```

```
logFC <- c(rnorm(n - 100), rnorm(100, 2))
pval  <- c(runif(n - 100), 10^(-runif(100, 2, 6)))
padj  <- p.adjust(pval, "BH")

df <- data.frame(logFC = logFC, padj = padj)
df$cat <- with(df, ifelse(padj < 0.05 & abs(logFC) > 1,
                          ifelse(logFC > 0, "up", "down"), "ns"))

ggplot(df, aes(logFC, -log10(padj), colour = cat)) +
  geom_point(alpha = 0.5, size = 1) +
  geom_hline(yintercept = -log10(0.05), linetype = 2) +
  geom_vline(xintercept = c(-1, 1), linetype = 2) +
  scale_colour_manual(values = c(up = "#E76F51",
                                 down = "#2A9D8F",
                                 ns = "grey70")) +
  labs(x = "log2 fold change", y = "-log10 adjusted p",
       title = "Volcano plot (DE results)") +
  theme_minimal()
```

12.52.6 Output & Results

A scatter with up- and down-regulated genes highlighted in distinct colours and dashed lines at the chosen FC and FDR thresholds. The visual immediately conveys how many genes pass both criteria, the symmetry of up vs down regulation, and the distribution of effect sizes among significant hits.

12.52.7 Interpretation

A reporting sentence (figure caption): "Volcano plot of differential expression between conditions A and B (n = 6 per group); 100 genes passed FDR < 0.05 and $|\log_2 \mathrm{FC}| > 1$ (45 up, 55 down); the five most significant genes are labelled. Dashed lines mark the FDR and fold-change thresholds." Always state the cutoffs and sample sizes.

12.52.8 Practical Tips

- Use `EnhancedVolcano::EnhancedVolcano()` for publication-ready volcano plots with automatic gene labelling, customisable thresholds, and `ggrepel`-based label placement.
- Truncate extreme $-\log_{10} p$ values (e.g. cap at 50) for readability; very small p-values otherwise stretch the y-axis and compress the bulk of the plot.
- Report both raw and FDR-adjusted p-value thresholds transparently in the figure caption.

- Add gene labels only for the top-ranked hits or biologically-curated genes of interest; many labels obscure the plot.
- Volcano plots do not show absolute expression magnitude; pair with MA plots when expression-dependent effects are suspected.
- For comparing multiple contrasts, use facets or different colour schemes; overlaying multiple contrasts on a single volcano usually fails.

12.52.9 R Packages Used

`ggplot2` for hand-built volcano plots; `EnhancedVolcano` for publication-ready volcano plots with extensive customisation; `ggrepel` for non-overlapping gene labels; `Glimma::glimmaVolcano()` for interactive volcano plots embeddable in HTML reports.

12.52.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

12.52.11 See also — labs in this chapter

- RNA-seq with DESeq2 / edgeR
- Enrichment analysis (GSEA / ORA)
- scRNA-seq with Seurat

12.53 Week 4, Session 2 — Enrichment analysis (GSEA / ORA)

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

12.53.1 Learning objectives

- Perform an over-representation analysis (ORA) by the hypergeometric test on a simulated example.
- Describe the GSEA running-sum statistic and when it is preferred over ORA.
- Recognise the hazards of gene-set overlap and of using different background universes.

12.53.2 Prerequisites

Multiple testing; bulk RNA-seq DE results.

12.53.3 Background

Pathway or gene-set enrichment analyses take a list of candidate genes — from a differential-expression analysis, say — and ask whether any annotated set contains more of them than expected by chance. Over- representation analysis (ORA) is the simplest version: given a universe of N genes, k of which are in a pathway, and a shortlist of n genes of which m are in that pathway, the hypergeometric distribution returns a p-value for the overlap.

Gene-set enrichment analysis (GSEA) uses the full ranked list of genes and tests whether members of a pathway are concentrated at the top or bottom of the ranking via a running-sum statistic. Unlike ORA, it does not require a threshold, and it retains information from genes that fall just short of significance.

Both methods are sensitive to the background universe. If the universe is "all human genes" but the measurement technology only detects half of them, the p-values are wrong. The correct universe is the set of genes actually tested in the upstream analysis.

12.53.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

12.53.5 1. Goal

Demonstrate a hypergeometric ORA by hand, then sketch the `fgsea` call pattern for a running-sum GSEA.

12.53.6 2. Approach

Simulate a universe of 5 000 genes; a pathway of 80 members is enriched for differentially expressed (DE) genes.

```
N <- 5000
pathway <- sample(seq_len(N), 80)
true_de <- unique(c(sample(pathway, 40), sample(setdiff(seq_len(N),
 ↪  pathway), 60)))

tibble(in_pathway = seq_len(N) %in% pathway,
       de = seq_len(N) %in% true_de) |>
  count(in_pathway, de) |>
```

```
  ggplot(aes(in_pathway, n, fill = de)) +
  geom_col(position = "dodge") +
  labs(x = "gene in pathway?", y = "count")
```

12.53.7 3. Execution

Hypergeometric test by hand.

```
n_de <- length(true_de)
k_in <- sum(true_de %in% pathway)
p_hyper <- phyper(k_in - 1, 80, N - 80, n_de, lower.tail = FALSE)
c(overlap = k_in, expected = n_de * 80 / N, p = p_hyper)
```

Make a ranked gene list (as if from DE) and sketch `fgsea`.

```
stat <- rnorm(N)
stat[pathway] <- stat[pathway] + 1.2    # shift pathway genes up
names(stat) <- paste0("g", seq_len(N))
stat <- sort(stat, decreasing = TRUE)

# Running-sum demonstration by hand.
in_set <- names(stat) %in% paste0("g", pathway)
p_hit <- abs(stat) / sum(abs(stat[in_set]))
p_miss <- 1 / sum(!in_set)
run_sum <- cumsum(ifelse(in_set, p_hit, -p_miss))
ES <- run_sum[which.max(abs(run_sum))]
tibble(i = seq_along(run_sum), rs = run_sum) |>
  ggplot(aes(i, rs)) + geom_line() +
  geom_hline(yintercept = 0, colour = "grey50") +
  labs(x = "rank", y = "running enrichment score")
```

`fgsea` sketch.

```
library(fgsea)
gene_sets <- list(pathway = paste0("g", pathway))
fgsea_res <- fgsea(pathways = gene_sets, stats = stat,
                   minSize = 15, maxSize = 500)
```

12.53.8 4. Check

Compare the ORA result for different shortlist sizes.

```
sweep_ora <- sapply(c(50, 100, 200, 500), function(top) {
  short <- names(stat)[seq_len(top)]
  k <- sum(short %in% paste0("g", pathway))
  phyper(k - 1, 80, N - 80, top, lower.tail = FALSE)
})
```

```
tibble(top = c(50, 100, 200, 500), p = sweep_ora)
```

12.53.9 5. Report

> In a simulated universe of 5 000 genes with a pathway of 80 members enriched for DE, the hypergeometric ORA for the full DE shortlist gave $p =$ `signif(p_hyper,2)`. A running-sum GSEA on the full ranked list produced an enrichment score of `round(ES,2)` with the running sum peaking near the top.

ORA is a quick, interpretable first pass; GSEA is better when the effect is diffuse. The two answer different questions and sometimes disagree; both are worth showing when the result matters.

12.53.10 Common pitfalls

- Using the full genome as the ORA universe when only a subset was assayed.
- Double-counting overlapping pathways as independent discoveries.
- Reporting a GSEA p-value without the normalised enrichment score.

12.53.11 Further reading

- Subramanian A et al. (2005), *Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles.*
- Huang DW, Sherman BT, Lempicki RA (2009), *Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists.*

12.53.12 Session info

```
sessionInfo()
```

12.53.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

12.54 Week 4, Session 1 — RNA-seq with DESeq2 / edgeR

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

12.54.1 Learning objectives

- Describe the count-model assumptions behind DESeq2 and edgeR and explain the role of dispersion estimation.
- Set up a design matrix for a two-group comparison and for a more complex experiment.
- Recognise when a log-transformed `limma-voom` workflow is preferable.

12.54.2 Prerequisites

GLMs; multiple testing.

12.54.3 Background

Bulk RNA-sequencing yields integer counts per gene per sample. Raw counts are heteroscedastic — variance grows with the mean — and the distribution has a long tail. DESeq2 and edgeR model counts as negative-binomial, with a gene-specific dispersion shrunk toward a trend estimated across the dataset. This shrinkage is the statistical heart of modern differential-expression analysis in low-replicate studies.

`limma-voom` takes a different route: model the log-counts with precision weights that reflect the count-level mean–variance trend. The output is a familiar linear-model workflow with all of its tools (contrasts, interactions, F-tests). For large or complex designs, `limma-voom` is often the most flexible.

DESeq2 and edgeR are Bioconductor packages, not CRAN. We show their call pattern with `#|eval:false` and walk through a conceptually parallel simulation with base R GLMs so the ideas run end-to-end in any environment.

12.54.4 Setup

```
library(tidyverse)
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

12.54.5 1. Goal

Simulate a small RNA-seq count matrix with two conditions and demonstrate the DESeq2/edgeR pipeline pattern alongside a base-R negative-binomial GLM that actually runs.

12.54.6 2. Approach

Simulate 500 genes × 10 samples (5 per group). Ten genes are truly differentially expressed.

```
n_gene <- 500; n_samp <- 10
group <- rep(c("A", "B"), each = 5)
mu <- 2 * runif(n_gene, 1, 10)   # baseline mean per gene
lfc <- c(rep(log(3), 10), rep(0, n_gene - 10))
counts <- sapply(seq_len(n_samp), function(j) {
  rate <- mu * exp(ifelse(group[j] == "B", lfc, 0))
  rnbinom(n_gene, mu = rate, size = 5)
})
rownames(counts) <- paste0("g", seq_len(n_gene))
colnames(counts) <- paste0("s", seq_len(n_samp))

tibble(mean = rowMeans(counts), var = apply(counts, 1, var)) |>
  ggplot(aes(mean, var)) + geom_point(alpha = 0.3) +
  scale_x_log10() + scale_y_log10() +
  geom_abline(slope = 1, intercept = 0, colour = "grey50") +
  labs(title = "mean–variance (log–log)")
```

12.54.7 3. Execution

A DESeq2 call pattern (not run).

```
library(DESeq2)
dds <- DESeqDataSetFromMatrix(
  countData = counts,
  colData   = data.frame(group = group),
  design    = ~ group
)
dds <- DESeq(dds)
res <- results(dds, contrast = c("group", "B", "A"))
head(res[order(res$padj), ])
```

An edgeR call pattern (not run).

```
library(edgeR)
dge <- DGEList(counts = counts, group = group)
dge <- calcNormFactors(dge)
design <- model.matrix(~ group)
dge <- estimateDisp(dge, design)
fit <- glmQLFit(dge, design)
qlf <- glmQLFTest(fit, coef = 2)
topTags(qlf)
```

A per-gene negative-binomial GLM that runs.

```
pvals <- numeric(n_gene)
logfcs <- numeric(n_gene)
for (g in seq_len(n_gene)) {
```

```
  y <- counts[g, ]
  fit <- try(glm.nb(y ~ group), silent = TRUE)
  if (inherits(fit, "try-error")) { pvals[g] <- NA; next }
  s <- summary(fit)
  pvals[g]  <- s$coefficients[2, 4]
  logfcs[g] <- s$coefficients[2, 1]
}
padj <- p.adjust(pvals, method = "BH")
```

12.54.8 4. Check

Volcano plot with the truly DE genes flagged.

```
truth <- c(rep("DE", 10), rep("null", n_gene - 10))
tibble(lfc = logfcs, padj = padj, truth = truth) |>
  drop_na() |>
  ggplot(aes(lfc, -log10(padj), colour = truth)) +
  geom_point(alpha = 0.6) +
  labs(x = "log fold change (B vs A)", y = "-log10(padj)")
```

```
mean(padj[1:10] < 0.05, na.rm = TRUE)
mean(padj[11:n_gene] < 0.05, na.rm = TRUE)
```

12.54.9 5. Report

> On a simulated 500-gene, 10-sample RNA-seq dataset with 10 truly differentially expressed genes (log fold change $\approx$ 1.1), per-gene negative-binomial GLMs followed by Benjamini–Hochberg adjustment recovered `round(mean(padj[1:10]<0.05,na.rm=TRUE)*100,0)`% of the true DE genes at FDR < 0.05 and yielded a false-discovery rate among null genes of `round(mean(padj[11:n_gene]<0.05,na.rm=TRUE)*100,1)`%.

DESeq2 and edgeR would improve power by shrinking gene-wise dispersion across the experiment; the per-gene fit shown here is a transparent but low-power version of the same idea.

12.54.10 Common pitfalls

- Running a t-test on raw counts — the variance structure is wrong.
- Filtering genes after seeing the results (data-driven filters are fine only if applied before testing).
- Forgetting to include batch or library-size terms in the design.

12.54.11 Further reading

- Love MI, Huber W, Anders S (2014), *Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2*.
- Law CW et al. (2014), *voom: precision weights unlock linear model analysis tools for RNA-seq read counts*.

12.54.12 Session info

```
sessionInfo()
```

12.54.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

12.55 Week 4, Session 3 — scRNA-seq with Seurat

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

12.55.1 Learning objectives

- Walk through the Seurat pipeline: QC, normalisation, feature selection, PCA, neighbour graph, clustering, UMAP.
- Describe why scRNA-seq counts need different handling from bulk counts.
- Produce a small analogous pipeline in base R with simulated cells and the `uwot` UMAP.

12.55.2 Prerequisites

UMAP; bulk RNA-seq differential expression.

12.55.3 Background

Single-cell RNA-seq measures gene expression in thousands of cells simultaneously, producing a sparse, noisy, zero-inflated matrix. Clustering and cell-type annotation are exploratory steps that require a chain of careful preprocessing: drop empty droplets, normalise library size, select variable genes, reduce to principal components, build a nearest-neighbour graph, cluster on the graph, embed with UMAP.

Seurat is the dominant R toolkit for this pipeline. Every step has alternatives — `SCTransform` vs `NormalizeData`, Louvain vs Leiden clustering, Harmony for

batch correction — and each choice changes the resulting partition. The good reflex is to run the same pipeline twice with a perturbed parameter and check that the biology survives.

Seurat is not on CRAN and its installation is non-trivial in a rendering environment. We sketch the pipeline with `#|eval:false` and run an analogous, tiny base-R + `uwot` pipeline on a simulated cell × gene matrix so the reasoning is visible in every render.

12.55.4 Setup

```
library(tidyverse)
library(uwot)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

12.55.5 1. Goal

Walk through an scRNA-seq analysis conceptually with a small simulated cell × gene matrix, ending at a 2-D embedding with cluster labels.

12.55.6 2. Approach

300 cells, 500 genes, three cell types with partly overlapping expression programmes.

```
n_cell <- 300; n_gene <- 500
cell_type <- rep(c("T", "B", "myeloid"), each = n_cell / 3)
mu_base <- exp(rnorm(n_gene, 1, 1))
X <- matrix(0, n_cell, n_gene)
for (i in seq_len(n_cell)) {
  program <- rep(1, n_gene)
  if (cell_type[i] == "T")       program[1:20]  <- 5
  if (cell_type[i] == "B")       program[21:40] <- 5
  if (cell_type[i] == "myeloid") program[41:60] <- 5
  X[i, ] <- rnbinom(n_gene, mu = mu_base * program, size = 2)
}
tibble(lib = rowSums(X), ct = cell_type) |>
  ggplot(aes(ct, lib)) + geom_boxplot() +
  labs(x = NULL, y = "total counts per cell")
```

12.55.7 3. Execution

Log-normalise, select variable genes, PCA, UMAP, cluster.

```
logX <- log1p(sweep(X, 1, rowSums(X), FUN = "/") * 1e4)
gene_var <- apply(logX, 2, var)
keep <- order(gene_var, decreasing = TRUE)[1:100]
pcs <- prcomp(logX[, keep])$x[, 1:10]
emb <- umap(pcs, n_neighbors = 20, min_dist = 0.1)

# A simple k-means cluster on the PCs as a stand-in.
km <- kmeans(pcs, centers = 3, nstart = 10)
tibble(x = emb[, 1], y = emb[, 2],
       cluster = factor(km$cluster), truth = cell_type) |>
  ggplot(aes(x, y, colour = cluster, shape = truth)) +
  geom_point(alpha = 0.8, size = 1) +
  labs(x = "UMAP1", y = "UMAP2")
```

A Seurat call pattern (not run).

```
library(Seurat)
obj <- CreateSeuratObject(counts = t(X))
obj <- NormalizeData(obj)
obj <- FindVariableFeatures(obj, nfeatures = 100)
obj <- ScaleData(obj)
obj <- RunPCA(obj, npcs = 10)
obj <- FindNeighbors(obj, dims = 1:10)
obj <- FindClusters(obj, resolution = 0.5)
obj <- RunUMAP(obj, dims = 1:10)
DimPlot(obj, label = TRUE)
```

12.55.8 4. Check

Contingency of cluster vs true cell type.

```
table(cluster = km$cluster, truth = cell_type)
```

12.55.9 5. Report

> On a simulated 300-cell × 500-gene dataset with three cell types, a small pipeline (log normalisation, top-100 variable genes, 10 PCs, UMAP, k-means) recovered the three types with high agreement. The same reasoning — preprocess, reduce, graph, cluster, embed — underlies the Seurat pipeline sketched in the accompanying code.

On a real dataset the critical choices are the normalisation (`SCTransform` vs `NormalizeData`), the number of PCs (elbow or `JackStraw`), and the clustering resolution. Perturb each and confirm that the biological conclusions do not.

12.55.10 Common pitfalls

- Clustering before QC; apoptotic cells and empty droplets cluster together.
- Forgetting that UMAP distances are not biological distances.
- Using marker genes from a tutorial on a different tissue for cell-type annotation.

12.55.11 Further reading

- Stuart T et al. (2019), *Comprehensive integration of single-cell data*.
- Luecken MD, Theis FJ (2019), *Current best practices in single-cell RNA-seq analysis: a tutorial*.

12.55.12 Session info

```
sessionInfo()
```

12.55.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 13

Machine Learning

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Cross-validation, regularisation (ridge/LASSO/elastic net), trees and ensembles, neural networks via `torch`/`keras`, SHAP and other interpretability tools, and the `tidymodels` workflow. False-discovery and knockoff-based selection are treated alongside CV as competing answers to the same question.

This chapter contains **46 method pages** and **8 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

13.1 Method pages

Method	Source slug
A Complete mlr3 Workflow	`mlr3-workflow`
A Complete tidymodels Workflow	`tidymodels-workflow`
Anomaly Detection	`anomaly-detection`
Bagging	`bagging`
Calibration Plots	`calibration-plots`
CatBoost	`catboost`
Class Weights for Imbalanced Data	`class-weights-imbalanced`
Convolutional Neural Networks: Introduction	`cnn-introduction`
Cross-Validation	`cross-validation`
DBSCAN as an ML Tool	`clustering-dbscan-ml`
Decision Trees (CART)	`decision-trees-cart`
Dropout and Early Stopping	`dropout-early-stopping`
Extremely Randomised Trees	`extremely-randomized-trees`
Feature Engineering	`feature-engineering`

Method	Source slug
Feature Selection	`feature-selection`
Feedforward Networks in torch	`feedforward-networks-torch`
Gradient Boosting	`gradient-boosting`
Isolation Forest	`isolation-forest`
Isotonic Regression Calibration	`isotonic-regression`
k-Means as an ML Tool	`clustering-kmeans-ml`
k-Nearest Neighbours Classification	`knn-classification`
k-Nearest Neighbours Regression	`knn-regression`
Kernel SVMs	`svm-kernel`
LightGBM	`lightgbm`
LIME Explanations	`lime-explanations`
Linear Discriminant as ML	`linear-discriminant-ml`
Linear Support Vector Machines	`svm-linear`
Logistic Regression as ML	`logistic-regression-ml`
Naive Bayes	`naive-bayes`
Nested Cross-Validation	`nested-cv`
Neural Networks: Introduction	`neural-networks-intro`
Partial Dependence Plots	`partial-dependence-plots`
Platt Scaling	`platt-scaling`
Preprocessing with recipes	`recipes-preprocessing`
Random Forests	`random-forest`
Regularisation in ML	`regularization-ridge-lasso-ml`
RNNs and LSTMs: Introduction	`rnn-lstm-intro`
SHAP Values	`shap-values`
SMOTE for Imbalanced Classes	`smote-imbalanced`
Stacking Ensembles	`stacking-ensembles`
Supervised Learning: Overview	`supervised-learning-overview`
The Bias-Variance Tradeoff	`bias-variance-tradeoff`
Train-Test Splits	`train-test-split`
Transformers: Overview	`transformer-overview`
Variable Importance	`variable-importance`
XGBoost	`xgboost`

13.2 Labs

Lab
Cross-validation, nested CV, bootstrap .632+
Regularisation: ridge, lasso, elastic net
Trees, random forests, gradient boosting
Interpretability and SHAP
Tabular neural networks with torch
Imaging and sequence models intro

Lab
tidymodels pipelines FDR, knockoffs, replication crisis

13.3 Anomaly Detection

13.3.1 Introduction

Anomaly detection flags observations that deviate from typical patterns. It is used for fraud detection, sensor fault monitoring, network intrusion, and adverse-event surveillance. Unlike classification, labels for anomalies are often unavailable at training time – methods are unsupervised or semi-supervised.

13.3.2 Prerequisites

Multivariate data; distance / density concepts.

13.3.3 Theory

Distance-based: kNN distance or average distance to neighbours; high = anomaly.

Density-based: LOF (Local Outlier Factor) compares local density to neighbours.

One-class SVM: learns a boundary enclosing most of the training data; points outside are anomalies.

Isolation Forest: exploits that anomalies are easier to isolate in random trees (shorter path length to terminal node).

Autoencoders: train a reconstruction network on normal data; anomalies have high reconstruction error.

13.3.4 Assumptions

Training data is mostly "normal"; anomaly pattern differs in some quantifiable way from normal.

13.3.5 R Implementation

```r
library(e1071)

set.seed(2026)
n <- 300
# Normal data is a ball; add some outliers
```

```r
df <- data.frame(
  x1 = c(rnorm(n, 0, 1), runif(20, -5, 5)),
  x2 = c(rnorm(n, 0, 1), runif(20, -5, 5))
)
labels <- c(rep("normal", n), rep("anomaly", 20))

# One-class SVM
oc <- svm(df, type = "one-classification", nu = 0.1, kernel = "radial")
pred <- predict(oc, df)                # TRUE = inlier, FALSE = anomaly

table(truth = labels, flagged = ifelse(pred, "inlier", "anomaly"))
```

13.3.6 Output & Results

Confusion matrix of the one-class SVM against ground truth; it correctly flags most of the 20 injected anomalies as outliers.

13.3.7 Interpretation

"One-class SVM detected 17 of 20 injected anomalies (sensitivity 0.85) with a 6 % false-positive rate on normal data. Isolation Forest gave similar results with less tuning."

13.3.8 Practical Tips

- Start with Isolation Forest; it is fast, robust, and has few hyperparameters.
- `nu` in one-class SVM approximates the expected anomaly fraction.
- Autoencoders are competitive for complex high-dimensional anomalies (images, signals).
- Validate with a held-out labelled anomaly set; unsupervised evaluation is hard.
- Combine multiple detectors for robustness; anomalies are rare and single-method false-positive rates compound.

13.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.3.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+

- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.4 Bagging

13.4.1 Introduction

Bagging — bootstrap aggregation, introduced by Breiman in 1996 — trains B models on B bootstrap samples of the training data and averages their predictions. The averaging dramatically reduces variance for high-variance, low-bias learners (deep decision trees, kNN with small k) without increasing their bias. Random forests are bagged trees with an additional feature-randomisation step that decorrelates the individual trees and amplifies the variance reduction.

13.4.2 Prerequisites

A working understanding of bootstrap resampling, the bias-variance trade-off, and decision trees as the canonical high-variance base learner.

13.4.3 Theory

For B predictions $\hat{y}_b$ each with variance σ^2 and pairwise correlation ρ, the variance of their mean is

$$\operatorname{Var}\left(\tfrac{1}{B}\sum_b \hat{y}_b\right) = \rho\sigma^2 + \frac{(1-\rho)\sigma^2}{B}.$$

As B grows, the second term vanishes; the first term is the residual variance limited by correlation. Bagging alone reduces variance via the $(1-\rho)/B$ contribution; adding decorrelation (feature subsampling, as in random forest) lowers ρ and produces further variance reduction.

Out-of-bag (OOB) observations — the roughly 37 % of rows not selected by each bootstrap — provide a free held-out validation set: for each row, average predictions from trees that did not include it during training.

13.4.4 Assumptions

The base learner benefits from variance reduction; bagging is most effective on high-variance learners (deep trees, kNN with small k, neural networks) and does essentially nothing for stable learners like linear regression.

13.4.5 R Implementation

```r
library(ipred); library(rpart)

set.seed(2026)
n <- 400
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n),
  y  = factor(ifelse(plogis(1 * rnorm(n) + 0.5 * rnorm(n)^2) > runif(n),
                     "yes", "no"))
)

# Bagged trees (no feature subsampling -- contrast with random forest)
m <- bagging(y ~ ., data = df, nbagg = 100, coob = TRUE)
print(m)
```

13.4.6 Output & Results

`ipred::bagging()` returns a fitted bagged ensemble with the OOB misclassification rate when `coob=TRUE`. The OOB error is the standard generalisation estimate; comparing it to a single tree's CV error quantifies the variance reduction from bagging.

13.4.7 Interpretation

A reporting sentence: "100 bagged trees achieved OOB misclassification rate of 0.21, an improvement over a single tree's CV error of 0.28; switching to a random forest with feature subsampling reduced OOB further to 0.18, illustrating that decorrelation contributes additional variance reduction beyond plain bagging." Always compare bagged ensembles against single-base-learner baselines.

13.4.8 Practical Tips

- Bagging primarily benefits high-variance learners (trees, kNN with small k); linear models are stable and gain little.
- Use $B = 100$ to 500 bootstrap replicates; returns diminish quickly beyond that, and computation grows linearly.
- OOB error is nearly unbiased for moderate to large n and free to compute alongside training; prefer it to cross-validation for rapid iteration.
- Random forests (bagging + feature subsampling) almost always outperform plain bagged trees; reach for `ranger` directly rather than tuning `ipred::bagging`.
- For classification, average class probabilities (soft voting) rather than class labels (hard voting); soft voting is consistently more accurate.
- Bagging extends beyond trees: any learner can be bagged via `ipred::bagging` with `bf` set to a custom training function.

13.4.9 R Packages Used

`ipred::bagging()` for canonical bagging on arbitrary base learners; `ranger` and `randomForest` for bagged trees with feature subsampling (random forests); `par snip::bag_tree()` for tidymodels integration; `mlr3pipelines::po("subsample")` for bagging within mlr3 pipelines.

13.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.4.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.5 The Bias-Variance Tradeoff

13.5.1 Introduction

Every supervised-learning error decomposes into structural bias (the wrong model class cannot fit the truth), variance (sampling noise in the fit), and irreducible noise. The bias-variance tradeoff is the single most important mental model for picking between a rigid and a flexible estimator.

13.5.2 Prerequisites

Regression basics; expected value and variance of estimators.

13.5.3 Theory

For squared loss and a target $f(x) + \varepsilon$ with $\mathbb{E}[\varepsilon] = 0$, $\mathrm{Var}(\varepsilon) = \sigma^2$:

$$\mathbb{E}[(y - \hat{f}(x))^2] = \underbrace{(\mathbb{E}[\hat{f}(x)] - f(x))^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]}_{\text{Variance}} + \sigma^2.$$

Complex models: low bias, high variance. Simple models: high bias, low variance. Regularisation, averaging (bagging), and dropout explicitly shift along this curve.

13.5.4 Assumptions

Squared loss and iid samples; trained estimator is a random function of the training set.

13.5.5 R Implementation

```
set.seed(2026)

true_f <- function(x) sin(x)
x_test <- seq(0, 2 * pi, length.out = 100)
f_true <- true_f(x_test)

n_reps  <- 200; n_train <- 30

simulate_fit <- function(degree) {
  preds <- replicate(n_reps, {
    x_train <- runif(n_train, 0, 2 * pi)
    y_train <- true_f(x_train) + rnorm(n_train, 0, 0.3)
    mdl <- lm(y_train ~ poly(x_train, degree))
    predict(mdl, newdata = data.frame(x_train = x_test))
  })
  bias <- rowMeans(preds) - f_true
  variance <- apply(preds, 1, var)
  list(bias2 = mean(bias^2), variance = mean(variance))
}

sapply(c(1, 3, 7, 15), simulate_fit)
```

13.5.6 Output & Results

As polynomial degree grows, bias^2 falls and variance rises; the sum (excess MSE beyond noise) is minimised at an intermediate flexibility.

13.5.7 Interpretation

“Polynomial degree 3 balanced bias (0.12) and variance (0.09), minimising excess MSE; degree 15 reduced bias to 0.001 but inflated variance to 0.45, hurting generalisation.”

13.5.8 Practical Tips

- Overfitting corresponds to high variance (the same structure fits differently across training resamples).
- Underfitting corresponds to high bias (structurally too restrictive to capture the signal).
- Ensembling (bagging, random forests) reduces variance without increasing bias.
- Regularisation (ridge, lasso) increases bias but reduces variance – tune by CV.
- Deep learning's "double descent" is a modern complication; the classical curve is the best starting intuition.

13.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.5.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.6 Calibration Plots

13.6.1 Introduction

A calibration plot (reliability diagram) plots predicted probability against observed frequency. A well-calibrated classifier's curve lies on the diagonal: when it predicts 0.7, events occur 70 % of the time. Poor calibration means probabilities cannot be trusted for decision thresholds or risk communication.

13.6.2 Prerequisites

Classification; predicted probabilities; binning.

13.6.3 Theory

Bin predictions by predicted probability; compute the observed frequency in each bin. Plot bin midpoints vs observed frequencies. A perfect classifier follows $y = x$; curves below indicate over-confidence, curves above indicate under-confidence.

Summary metrics: **Brier score** $= \mathbb{E}[(p - y)^2]$, **Expected Calibration Error (ECE)** $= \sum_k w_k |\text{acc}_k - \text{conf}_k|$.

13.6.4 Assumptions

Predictions are valid probabilities in $[0, 1]$; sufficient data per bin (10-20 bins for $n > 1000$).

13.6.5 R Implementation

```
library(probably); library(yardstick); library(dplyr); library(tibble);
 ↪  library(ggplot2)

set.seed(2026)
n <- 1000
# Simulate mis-calibrated predictions
true_y <- rbinom(n, 1, 0.4)
raw_p  <- plogis(-1 + 2 * true_y + rnorm(n, 0, 1.5))
# Inflate confidence to produce over-confident predictions
p_overconf <- raw_p^0.6
p_overconf <- pmin(pmax(p_overconf, 0.02), 0.98)

df <- tibble(truth = factor(true_y, levels = c(1, 0)),
             pred  = p_overconf)

cal_df <- cal_plot_breaks(df, truth, pred, num_breaks = 10)

# Manual reliability data + plot
df$bin <- cut(df$pred, seq(0, 1, 0.1), include.lowest = TRUE)
reliability <- df %>%
  group_by(bin) %>%
  summarise(mean_pred = mean(pred),
            obs_freq  = mean(as.integer(truth == "1")),
            n = n())

ggplot(reliability, aes(mean_pred, obs_freq)) +
  geom_point(size = 2, colour = "#2A9D8F") +
  geom_line() +
  geom_abline(linetype = 2) +
  labs(x = "Predicted probability", y = "Observed frequency",
```

```
       title = "Reliability diagram (simulated over-confident model)") +
  theme_minimal()

mean((df$pred - as.integer(df$truth == "1"))^2)   # Brier score
```

13.6.6 Output & Results

A reliability diagram (scatter of bin-means) and the Brier score. The simulated over-confident model lies above the diagonal at low predicted probabilities.

13.6.7 Interpretation

"The classifier over-predicts at low probabilities and under-predicts at high probabilities; Brier score 0.22 indicates moderate calibration error. Platt or isotonic recalibration on a validation set would flatten the curve toward the diagonal."

13.6.8 Practical Tips

- Tree ensembles (RF, XGBoost) are usually mis-calibrated; apply Platt scaling or isotonic regression on a validation set.
- Neural networks with modern training (high-capacity, low label smoothing) often over-confident; temperature scaling fixes it.
- Use stratified bins for reliable plots when probabilities concentrate near 0 or 1.
- Report Brier score and ECE alongside ROC AUC; AUC is rank-based and does not reveal calibration.
- For clinical applications, calibration matters more than discrimination; always verify on deployment-era data.

13.6.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.6.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro

- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.7 CatBoost

13.7.1 Introduction

CatBoost (Prokhorenkova et al., 2018) focuses on principled handling of categorical features via ordered target encoding – computing target statistics only from prior rows to avoid target leakage. It also uses **symmetric (oblivious) trees**: trees where every split at a given depth uses the same feature and threshold, yielding fast inference and natural regularisation.

13.7.2 Prerequisites

Gradient boosting; categorical encoding basics.

13.7.3 Theory

Ordered target encoding: for a categorical feature, the encoding of row i uses target statistics computed only over rows 1..i-1 (under a random permutation). This prevents target leakage present in ordinary mean-target encoding.

Ordered boosting: each tree is trained on residuals computed using an independent prefix of data, reducing prediction-shift bias.

Symmetric trees: fixed structure across layers; inference reduces to look-up tables, making CatBoost extremely fast at inference time.

13.7.4 Assumptions

Categorical features are correctly flagged; data are reasonably uniform (no temporal drift within a training set).

13.7.5 R Implementation

```
# library(catboost)  # install from
 ↪ https://catboost.ai/en/docs/installation/r-installation-binary-
 ↪ installation

set.seed(2026)
n <- 500
df <- data.frame(
  cat1 = sample(letters[1:5], n, replace = TRUE),
  cat2 = sample(c("a", "b", "c"), n, replace = TRUE),
  num1 = rnorm(n), num2 = rnorm(n),
```

```
  y    = factor(sample(c("0", "1"), n, replace = TRUE))
)

# pool <- catboost.load_pool(
#   data = df[, -5], label = as.integer(df$y) - 1,
#   cat_features = c(0, 1))
#
# model <- catboost.train(
#   learn_pool = pool,
#   params = list(iterations = 500, learning_rate = 0.05,
#                 depth = 6, loss_function = "Logloss",
#                 verbose = 0))
#
# importance <- catboost.get_feature_importance(model)
# importance

# Run summary for illustration (the fit itself needs the package install)
head(df)
```

13.7.6 Output & Results

A trained CatBoost model with feature-importance scores; categorical features are ordered-target-encoded internally.

13.7.7 Interpretation

"CatBoost achieved log-loss 0.36 on the held-out set. Feature importance ranked the categorical 'cat1' highest, leveraging ordered target encoding to extract signal that plain one-hot encoding misses."

13.7.8 Practical Tips

- Pass categorical columns as factor or integer; flag via `cat_features`.
- CatBoost handles missing values natively; no imputation needed.
- Symmetric trees are fast at inference; useful when deployment latency matters.
- Default hyperparameters are already strong; tune learning rate and depth first.
- On small data, CatBoost generally outperforms XGBoost/LightGBM on heavily categorical datasets.

13.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.7.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.8 Class Weights for Imbalanced Data

13.8.1 Introduction

Class weighting addresses imbalanced classification by scaling each observation's loss contribution inversely to its class frequency, giving the minority class more influence on the gradient or split decision. It achieves similar effects to SMOTE-style oversampling but without synthesising new data: simpler, deterministic, easier to reproduce, and natively supported by most classifiers (logistic regression, random forest, XGBoost, neural networks). For most imbalanced problems, class weights are the right starting point before reaching for resampling.

13.8.2 Prerequisites

A working understanding of classification loss functions, the accuracy paradox in imbalanced classification, and basic familiarity with resampling alternatives (oversampling, undersampling, SMOTE).

13.8.3 Theory

The weighted loss is

$$L = \sum_i w_{y_i} \cdot \ell(f(x_i), y_i),$$

where w_k is the weight for class k. Typical balanced weights use the inverse class frequency: $w_k = n/(Kn_k)$ where K is the number of classes and n_k is the count of class k. This makes the total weight per class equal, so each class contributes equally to the loss regardless of its sample size.

In tree-based methods, class weights enter the split-impurity computation; in gradient-boosted methods, they enter the gradient and Hessian; in neural networks, they multiply the per-example loss before summing.

13.8.4 Assumptions

The problem is genuinely a classification task with imbalanced classes; the imbalance reflects the population (not just an artefact of training-data collection that will not transfer to deployment).

13.8.5 R Implementation

```
library(ranger)

set.seed(2026)
n <- 500
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n),
  y  = factor(sample(c("pos", "neg"), n, replace = TRUE,
                     prob = c(0.05, 0.95)))
)

idx <- sample(n, n * 0.7)
train <- df[idx, ]; test <- df[-idx, ]

rf_unw <- ranger(y ~ ., data = train, num.trees = 200,
                 classification = TRUE)
pred_unw <- predict(rf_unw, test)$predictions

# Inverse-frequency class weights
cw <- setNames(as.numeric(1 / prop.table(table(train$y))),
               names(table(train$y)))
rf_w <- ranger(y ~ ., data = train, num.trees = 200,
               classification = TRUE, class.weights = cw)
pred_w <- predict(rf_w, test)$predictions

table(truth = test$y, unweighted = pred_unw)
table(truth = test$y, weighted   = pred_w)
```

13.8.6 Output & Results

Two confusion matrices: the unweighted random forest predicts almost exclusively the majority class (the rational behaviour given 95:5 imbalance and accuracy-maximising training); the weighted RF recovers more minority-class examples at the cost of a few additional false positives.

13.8.7 Interpretation

A reporting sentence: "Inverse-frequency class weighting on the 95:5 imbalanced dataset improved minority-class recall from 0.12 to 0.56 with a modest decrease in majority precision (0.97 to 0.93); the weighted classifier is more useful for screening applications where missing minority cases is more costly than false positives." Always state the weighting scheme and the trade-off in terms of the application's cost structure.

13.8.8 Practical Tips

- Always try class weights before SMOTE-style oversampling; they are deterministic, faster, and produce reproducible results.
- Tune weights by cross-validation; inverse-frequency is a sensible default but not necessarily optimal for the cost structure of every application.
- All major classifiers support class weights natively: `glmnet(weights=...)`, `ranger(class.weights=...)`, `xgboost(scale_pos_weight=...)`, `keras::compile(...loss_weights=...)`.
- Weighting shifts the decision boundary; inspect calibration plots and ROC curves to check probabilistic quality after weighting.
- For severely imbalanced problems (below 1 %), combine class weights with threshold tuning on a validation set; the default 0.5 cutoff is rarely optimal under heavy imbalance.
- For problems where the cost of errors is asymmetric and known, build a cost-sensitive classifier directly (`mlr3::msr("classif.costs")`) rather than tuning weights heuristically.

13.8.9 R Packages Used

`ranger` for random forest with native class weights; `glmnet` for penalised logistic regression with `weights=...`; `xgboost::xgboost(scale_pos_weight=...)` for gradient-boosted trees; `themis` for SMOTE-style alternatives within tidymodels; `parsnip::set_args(class_weights=...)` for class-weight support across multiple engines.

13.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.8.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+

- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.9 DBSCAN as an ML Tool

13.9.1 Introduction

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) groups points that are densely packed together and flags isolated points as noise. Unlike k-means it does not require specifying the number of clusters, handles non-convex cluster shapes, and produces an outlier-detection signal as a side effect — every point flagged as noise is a candidate anomaly. For machine-learning workflows where natural cluster discovery and outlier flagging are both useful, DBSCAN is the right starting point.

13.9.2 Prerequisites

A working understanding of distance metrics, density-based reasoning, and the limitations of partitioning algorithms (k-means) for non-convex cluster shapes.

13.9.3 Theory

DBSCAN has two hyperparameters: `eps` (the neighbourhood radius) and `minPts` (the minimum number of points required for a dense region). Three categories of point emerge from the algorithm:

- **Core point**: at least `minPts` neighbours within `eps`.
- **Border point**: within `eps` of a core point but not itself a core point.
- **Noise point**: neither core nor border.

Clusters are formed as maximally connected sets of core points (with their associated border points). Choosing `eps` is the practical challenge; the standard heuristic is the k-distance plot, which shows the distance to the k-th nearest neighbour for each point sorted ascending — a knee in the curve marks a sensible `eps`.

13.9.4 Assumptions

Cluster density is roughly uniform; the chosen `eps` matches the natural scale of the data; Euclidean distance is meaningful (or another distance has been substituted).

13.9.5 R Implementation

```r
library(dbscan)

set.seed(2026)
n <- 200
x1 <- c(rnorm(n/2, -2, 0.3), rnorm(n/2, 2, 0.3))
x2 <- c(rnorm(n/2, -2, 0.3), rnorm(n/2, 2, 0.3))
outliers_x1 <- runif(10, -4, 4)
outliers_x2 <- runif(10, -4, 4)
df <- data.frame(x1 = c(x1, outliers_x1),
                 x2 = c(x2, outliers_x2))

# Choose eps via the k-distance plot knee
kNNdistplot(df, k = 5); abline(h = 0.5, lty = 2, col = "red")

db <- dbscan(df, eps = 0.5, minPts = 5)
table(cluster = db$cluster)

plot(df, col = db$cluster + 1, pch = 16, asp = 1,
     main = "DBSCAN clustering (cluster 0 = noise)",
     xlab = "x1", ylab = "x2")
```

13.9.6 Output & Results

`dbscan()` returns a vector of cluster labels with `0` indicating noise. The k-distance plot guides `eps` selection; the cluster scatter shows densely-packed groups separated by noise points.

13.9.7 Interpretation

A reporting sentence: "DBSCAN with `eps=0.5` and `minPts=5` recovered the two natural clusters injected in the simulation (96 of 100 cluster-1 points and 98 of 100 cluster-2 points correctly assigned) and flagged 10 of the 10 injected outliers as noise (cluster 0); the remaining noise calls were 6 boundary points where the local density is genuinely lower." Always state `eps`, `minPts`, and the noise count.

13.9.8 Practical Tips

- Set `eps` via the k-distance plot's knee; it is the most sensitive hyperparameter and a poor choice fails the algorithm completely.
- 'minPts $= 2 \times$d$ (dimensions) is a sensible starting heuristic; inspect cluster counts and adjust if they are unreasonably few or many.
- For varying-density clusters, switch to HDBSCAN (`dbscan::hdbscan`); it adapts the density threshold per region and is more robust.

- Use the noise label (`cluster==0`) as a built-in outlier detector; for anomaly-detection tasks, this is often the simplest and most interpretable approach.
- Scale features before clustering (`scale()` or `recipes::step_normalize()`); DBSCAN's Euclidean distance is sensitive to feature scales.
- For high-dimensional data, distance concentration weakens DBSCAN; reduce dimensionality with PCA or UMAP first.

13.9.9 R Packages Used

`dbscan` for `dbscan()`, `hdbscan()`, `optics()`, and the k-distance heuristic; `fpc::dbscan()` for an alternative implementation; `cluster` for related clustering algorithms; `factoextra::fviz_cluster()` for ggplot-style visualisation of DBSCAN output.

13.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.9.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.10 k-Means as an ML Tool

13.10.1 Introduction

k-means is usually taught as a stand-alone unsupervised clustering method, but in ML pipelines it is also a versatile feature-engineering primitive. Pre-clustering observations and using cluster membership (or distance-to-centroid) as new features can substantially improve downstream supervised models, especially when the underlying signal is non-linear and a clustering captures local structure that linear models miss. k-means is also the workhorse for vector quantisation, image segmentation, and customer-segmentation tasks.

13.10.2 Prerequisites

A working understanding of Euclidean distances, centroids, the within-cluster sum-of-squares objective, and the bias-variance trade-off in feature engineering.

13.10.3 Theory

k-means minimises the within-cluster sum of squares (WCSS):

$$\min_{C} \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - \mu_k\|^2.$$

Lloyd's algorithm alternates assignment (each point to its nearest centroid) and update (each centroid to the mean of its assigned points) until convergence. The algorithm is non-convex; multiple random starts are needed to find a good local optimum. The number of clusters K is a hyperparameter chosen by the elbow, silhouette, or gap statistic methods.

For feature engineering, two derived quantities from a k-means fit are useful: - **Cluster membership** (one-hot or label): a categorical feature indicating which cluster each observation belongs to. - **Distance to each centroid**: K continuous features that often carry more predictive information than the cluster label alone.

13.10.4 Assumptions

Clusters are roughly spherical and equal-sized (the implicit Voronoi-cell geometry); features are scaled to comparable units; the chosen K matches a meaningful structure in the data.

13.10.5 R Implementation

```r
set.seed(2026)
n <- 300
df <- data.frame(x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n))

# Scale for kmeans
df_s <- scale(df)

# Fit with K=4 and use cluster membership as a feature
km <- kmeans(df_s, centers = 4, nstart = 20)
df$cluster <- factor(km$cluster)

# Distance to each centroid also becomes a feature
dist_to_centroids <- as.matrix(dist(rbind(df_s, km$centers)))
dist_features <- dist_to_centroids[1:n, (n + 1):(n + 4)]
```

```r
colnames(dist_features) <- paste0("dist_c", 1:4)

head(cbind(df, dist_features))
km$tot.withinss
```

13.10.6 Output & Results

The augmented data frame contains cluster membership plus four distance-to-centroid features. The model's WCSS quantifies clustering tightness; lower values indicate more compact clusters.

13.10.7 Interpretation

A reporting sentence: "k-means with $K = 4$ on the standardised features reduced WCSS from 900 (one cluster, equivalent to total variance) to 620 (four clusters); using cluster membership and distance-to-centroid as additional features in the downstream XGBoost classifier improved cross-validated RMSE by 3 %." Always state K, the scaling, and the number of random starts.

13.10.8 Practical Tips

- Scale features before k-means; unscaled features with different units dominate the Euclidean distance and bias the partition.
- Use `nstart=20` or higher to escape local minima; k-means is non-convex and a single start often produces sub-optimal solutions.
- k-means-style feature engineering works best when clusters actually exist in the data; on truly random data it adds noise features and can hurt downstream performance.
- Distance-to-centroid features are often more informative than the raw cluster label because they capture the continuous geometry that the discrete label discards.
- For high-dimensional data, reduce dimensionality with PCA before k-means; Euclidean distances concentrate and clusters lose meaning in high dimensions.
- For categorical or mixed data, k-means does not apply directly; use k-modes or PAM with Gower distance instead.

13.10.9 R Packages Used

Base R `kmeans()` for the canonical implementation; `factoextra::fviz_cluster()` for ggplot-style cluster visualisation; `klaR::kmodes()` for categorical alternatives; `recipes::step_kmeans_features()` (via `embed`) for tidymodels-flavoured pipeline integration; `cluster::clara()` for k-medoids alternatives that scale to large datasets.

13.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.10.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.11 Convolutional Neural Networks: Introduction

13.11.1 Introduction

Convolutional Neural Networks (CNNs; LeCun, 1989) replace fully-connected layers with local, weight-sharing convolutional filters. The shared weights encode translation invariance: a feature detector (edge, texture) learned once applies at every spatial location. CNNs dominate image classification, segmentation, and many signal-processing tasks.

13.11.2 Prerequisites

Feedforward networks; convolutions and cross-correlations; image tensors.

13.11.3 Theory

Convolutional layer: a set of learned filters K_c; each filter slides across the input, producing a feature map. Output channel count equals number of filters.

Pooling: max or average over a spatial window; reduces dimensionality and increases receptive field.

Typical architecture (classic): conv -> ReLU -> pool, repeated; then flatten to fully-connected for classification. Modern architectures (ResNet, EfficientNet) add residual connections and batch normalisation.

13.11.4 Assumptions

Inputs have spatial structure (images, spectrograms, time series); sufficient training data or a pretrained backbone.

13.11.5 R Implementation

```r
library(torch)

torch_manual_seed(2026)

# Simulate a small batch of "images"
x <- torch_randn(4, 1, 28, 28)         # 4 images, 1 channel, 28x28
y <- torch_randint(0, 10, size = list(4), dtype = torch_long())

# Simple CNN: 2 conv layers + FC head
cnn <- nn_module(
  initialize = function() {
    self$conv1 <- nn_conv2d(1, 16, kernel_size = 3, padding = 1)
    self$conv2 <- nn_conv2d(16, 32, kernel_size = 3, padding = 1)
    self$pool  <- nn_max_pool2d(2)
    self$fc1   <- nn_linear(32 * 7 * 7, 64)
    self$fc2   <- nn_linear(64, 10)
  },
  forward = function(x) {
    x <- self$pool(nnf_relu(self$conv1(x)))
    x <- self$pool(nnf_relu(self$conv2(x)))
    x <- x$view(c(x$size(1), -1))
    nnf_relu(self$fc1(x)) %>% self$fc2()
  }
)
model <- cnn()
logits <- model(x)
logits$shape               # [4, 10]
```

13.11.6 Output & Results

Logits of shape [batch, classes] ready for a cross-entropy loss during training.

13.11.7 Interpretation

“A two-layer CNN produced 10-class logits for a batch of 28x28 inputs. In a real MNIST pipeline this architecture would reach ~99 % accuracy after training.”

13.11.8 Practical Tips

- Pre-trained CNN backbones (ResNet50, EfficientNet) with fine-tuning outperform training from scratch for most tasks with < 1M images.
- Data augmentation (flip, rotate, crop) is essential for image classification; use `torchvision::transform_*`.
- Batch normalisation accelerates training and stabilises gradients; include after every conv layer.
- For 1-D signals (ECG, audio), use 1-D convolutions (`nn_conv1d`).
- Vision transformers (ViT) now rival or exceed CNNs on large datasets; CNNs remain preferred for small-to-moderate data.

13.11.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.11.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.12 Cross-Validation

13.12.1 Introduction

Cross-validation is the primary tool for estimating how well a predictive model will generalise to unseen data. The in-sample error – the error of the fitted model on the data used to train it – is almost always too optimistic. Cross-validation repeatedly splits the data into training and held-out portions, fits the model on the training portion, evaluates on the held-out portion, and averages the held-out performance as an estimate of true generalisation error.

13.12.2 Prerequisites

The reader should understand the concepts of overfitting, bias and variance, and the training vs. test-set distinction. Familiarity with `tidymodels` is helpful but

not assumed.

13.12.3 Theory

A model's **generalisation error** is its expected loss on a new observation drawn from the same population as the training data. The **training error** is its loss on the training data itself. The difference between them – the **optimism** of the training error – grows with model flexibility and shrinks with sample size.

13.12.3.1 k-fold cross-validation

The data are partitioned into k approximately equal folds. For each fold $j = 1, \dots, k$, the model is fitted on the other $k-1$ folds and evaluated on the held-out j-th fold. The k held-out losses are averaged. A common choice is $k = 5$ or $k = 10$, which balances bias (too few folds means each training set is noticeably smaller than the full data) against variance (too many folds means the estimates are correlated).

13.12.3.2 Leave-one-out cross-validation (LOOCV)

The extreme case $k = n$: each fold has exactly one observation. LOOCV has low bias (the training sets are nearly the full data) but high variance and high computational cost. It is the right choice when n is small and the model is fast to fit.

13.12.3.3 Stratified cross-validation

For classification with class imbalance, folds are constructed to preserve class proportions. This reduces variability in the estimated performance and avoids folds with zero minority-class examples.

13.12.3.4 Repeated k-fold

k-fold CV is repeated with different random partitions, and the results averaged. This reduces Monte Carlo variability in the estimate without the computational cost of LOOCV.

13.12.3.5 Nested cross-validation

When hyperparameters are tuned on CV, the tuned performance estimate is optimistically biased because the hyperparameters have been chosen to maximise the same metric. Nested CV uses an outer loop for honest generalisation estimation and an inner loop for tuning: the outer folds never see the data used to choose the hyperparameters.

13.12.4 Assumptions

CV is a general technique, but it assumes:

1. The splits are representative of the data-generating process – the training and test folds come from the same distribution.
2. The observations are exchangeable (or the CV scheme respects the dependence structure – see below).
3. The performance metric is well-estimated by an average over folds.

For time-series or clustered data, standard CV leaks information between folds and overstates performance. Use time-series CV (expanding window) or group-blocked CV (by subject, site, etc.) in those cases.

13.12.5 R Implementation

```r
library(tidymodels)

data(penguins, package = "palmerpenguins")
penguins <- tidyr::drop_na(penguins)

set.seed(2026)
split <- initial_split(penguins, prop = 0.75, strata = species)
train <- training(split)
test  <- testing(split)

folds <- vfold_cv(train, v = 10, strata = species)

rec <- recipe(species ~ bill_length_mm + bill_depth_mm +
              flipper_length_mm + body_mass_g, data = train) |>
  step_normalize(all_numeric_predictors())

spec <- multinom_reg() |> set_engine("nnet")

wf <- workflow() |> add_recipe(rec) |> add_model(spec)

cv_results <- fit_resamples(wf, resamples = folds,
                            metrics = metric_set(accuracy, roc_auc))

collect_metrics(cv_results)
```

This performs stratified 10-fold CV for a multinomial logistic regression on the `penguins` dataset. `fit_resamples()` handles the CV loop; `collect_metrics()` summarises the fold-level performance.

13.12.6 Output & Results

A typical output:

.metric	.estimator	mean	n	std_err
accuracy	multiclass	0.981	10	0.008
roc_auc	hand_till	0.999	10	0.000

The mean across folds is the point estimate; the standard error is the Monte Carlo uncertainty across folds.

13.12.7 Interpretation

The 10-fold cross-validated accuracy is 0.981 (SE 0.008), and the ROC AUC is 0.999 (SE < 0.001). These are estimates of out-of-sample performance on the penguin species classification task. The held-out test set is still unused; it is reserved for the final, honest evaluation after all modelling decisions are locked.

13.12.8 Practical Tips

- Preprocessing (scaling, imputation, encoding) must go *inside* the resampling loop to prevent leakage. `recipes` with `workflow()` does this correctly by default.
- Report both the mean and the standard error across folds so readers know how confident the estimate is.
- For hyperparameter tuning, use nested CV (or a train/validation/test split) – tuning on the same CV you report is biased.
- For time series, use `rolling_origin()` or `sliding_window()` in `rsample`; never use `vfold_cv()` on serial data.
- Small datasets have high-variance CV estimates – consider reporting a 95% CI across folds or using repeated CV.

13.12.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.12.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro

- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.13 Decision Trees (CART)

13.13.1 Introduction

Decision trees partition the feature space by recursive binary splits, fitting a constant prediction in each terminal region (leaf). The Classification and Regression Trees (CART) algorithm by Breiman, Friedman, Olshen and Stone (1984) is the dominant implementation. Trees are interpretable (a single fitted tree can be read as a sequence of if-else rules), handle mixed numeric and categorical data without preprocessing, require no feature scaling, and serve as the building blocks for random forests and gradient boosting. Their main weakness is high variance — small data perturbations produce very different trees — which motivates ensemble methods that aggregate many trees.

13.13.2 Prerequisites

A working understanding of classification and regression problems, impurity measures (Gini, entropy, variance), and the bias-variance trade-off in model complexity.

13.13.3 Theory

At each node, CART searches over all features and candidate split points, choosing the split that maximally reduces impurity:

- **Regression**: minimise within-node variance.
- **Classification**: minimise Gini $1 - \sum_k p_k^2$ or entropy $-\sum_k p_k \log p_k$.

Trees grow recursively until leaves are pure or a minimum-size constraint is hit, then are pruned back using cost-complexity pruning. The cost-complexity criterion is

$$R_\alpha(T) = R(T) + \alpha |T|,$$

where $R(T)$ is the training error and $|T|$ the number of terminal leaves. The pruning parameter α is chosen by cross-validation; the standard rule selects the smallest tree within one CV-standard-error of the minimum.

13.13.4 Assumptions

Splits are axis-aligned (a single tree captures interactions only by combining splits at different depths, which limits its ability to model rotated or oblique decision boundaries); observations are independent.

13.13.5 R Implementation

```
library(rpart); library(rpart.plot)

set.seed(2026)
n <- 400
df <- data.frame(
  x1 = runif(n), x2 = runif(n),
  y  = factor(ifelse(runif(n) < plogis(-2 + 3 * runif(n) + 2 * runif(n)),
                     "pos", "neg"))
)

tree <- rpart(y ~ x1 + x2, data = df,
              method = "class",
              control = rpart.control(cp = 0.001))

printcp(tree)                  # CV error vs complexity
tree_pruned <- prune(tree,
                     cp = tree$cptable[which.min(tree$cptable[,
                      ↪  "xerror"]), "CP"])
rpart.plot(tree_pruned, box.palette = c("#F4A261", "#2A9D8F"))
```

13.13.6 Output & Results

`rpart()` returns a fitted tree object; `printcp()` reports the cross-validation error at each candidate complexity parameter; `prune()` produces the cost-complexity-pruned tree at the chosen α. `rpart.plot()` visualises the tree with split conditions and leaf class proportions.

13.13.7 Interpretation

A reporting sentence: "CART with cost-complexity pruning at the CV-optimal α retained 5 leaves; the dominant split was on x_1 at 0.52, partitioning 78 % of positive cases into one branch and 80 % of negatives into the other; sub-splits on x_2 refined predictions within each branch." Always state the pruning method and the resulting tree size.

13.13.8 Practical Tips

- Always prune using the `xerror` column of the CP table; unpruned trees over-fit severely.
- A single tree has high variance; for predictive accuracy, prefer random forests, gradient boosting, or extra trees.
- Trees handle missing data via surrogate splits (rpart's default); this is one of the few methods that does so without explicit imputation.

- Visualise with `rpart.plot()` for `ggplot`-friendly output; `partykit::as.party()` provides a richer object with print and plot methods.
- For regression, use `method="anova"`; for classification, `method="class"`; for survival, `method="exp"` and the `rpart` package has built-in support.
- For interpretability with better accuracy, use a small ensemble (e.g., 50 trees in a random forest) and aggregate variable-importance measures from across trees.

13.13.9 R Packages Used

`rpart` for CART implementation with cost-complexity pruning; `rpart.plot` for tree visualisation; `partykit` for an alternative tree framework with richer output classes; `tree` for an older alternative implementation; `caret` and `tidymodels::decision_tree()` for cross-validated pruning within ML pipelines.

13.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.13.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.14 Dropout and Early Stopping

13.14.1 Introduction

Neural networks with many parameters easily overfit. Dropout (Srivastava et al., 2014) randomly zeros activations during training, acting as implicit ensembling. Early stopping halts training when validation loss stops improving, preventing the model from memorising training noise. Both are simple, cheap, and among the most effective regularisation techniques for deep nets.

13.14.2 Prerequisites

Feedforward networks; validation-based evaluation.

13.14.3 Theory

Dropout: during training, each neuron's activation is set to zero with probability p (typically 0.2-0.5); during inference, activations are scaled by $1 - p$ or equivalent. Acts like training an ensemble of 2^n sub-networks with shared weights.

Early stopping: train on training set, monitor a validation loss; halt when validation loss stops improving for a "patience" window. Implicitly regularises – the final model has parameters closer to the initialisation than full convergence would reach.

13.14.4 Assumptions

A held-out validation set exists; model is over-parameterised enough to overfit without regularisation.

13.14.5 R Implementation

```
library(torch)

torch_manual_seed(2026)

# Model with dropout after each hidden layer
net <- nn_module(
  initialize = function() {
    self$fc1  <- nn_linear(10, 64)
    self$drop <- nn_dropout(p = 0.3)
    self$fc2  <- nn_linear(64, 1)
  },
  forward = function(x) {
    x %>% self$fc1() %>% nnf_relu() %>% self$drop() %>% self$fc2()
  }
)
model <- net()
opt   <- optim_adam(model$parameters, lr = 1e-2)

n <- 500
x_tr <- torch_randn(n, 10); y_tr <- torch_randn(n, 1)
x_va <- torch_randn(200, 10); y_va <- torch_randn(200, 1)

best_val <- Inf; patience <- 5; counter <- 0
for (epoch in 1:60) {
```

```
  model$train()
  opt$zero_grad()
  loss <- nnf_mse_loss(model(x_tr), y_tr)
  loss$backward(); opt$step()

  model$eval()
  val <- nnf_mse_loss(model(x_va), y_va)$item()
  if (val < best_val - 1e-4) { best_val <- val; counter <- 0 }
  else { counter <- counter + 1; if (counter >= patience) break }
}
cat("Stopped at epoch", epoch, "  best_val=", round(best_val, 3), "\n")
```

13.14.6 Output & Results

Training halts when validation loss has stopped improving for 5 epochs; dropout noise in the forward pass further regularises the model.

13.14.7 Interpretation

"Combining 30 % dropout with early stopping (patience 5) halted training at epoch 23 with best validation MSE 0.98, preventing the over-fitting observed beyond epoch 30 without regularisation."

13.14.8 Practical Tips

- Dropout probability 0.2-0.5 for hidden layers; avoid dropout on output layers.
- Dropout is automatically disabled in `eval()` mode; don't forget to switch modes.
- Use weight decay (L2 on weights) jointly with dropout for stronger regularisation.
- Early stopping is the cheapest regulariser; always implement it for deep nets.
- For batch norm + dropout, place dropout after batch norm; order matters for training stability.

13.14.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.14.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.15 Extremely Randomised Trees

13.15.1 Introduction

Extremely Randomised Trees — usually shortened to "Extra Trees" — are a variant of random forests introduced by Geurts, Ernst, and Wehenkel in 2006. The defining innovation: instead of optimising the split threshold at each node, Extra Trees draws the threshold *at random* from the feature's range. The additional source of randomness increases bias slightly but reduces variance further, often producing models with accuracy competitive to random forests at a fraction of the training time.

13.15.2 Prerequisites

A working understanding of random forests, decision-tree splitting criteria (Gini, variance), and the bias-variance trade-off in ensemble methods.

13.15.3 Theory

At each node, Extra Trees:

1. Choose m random features (as in random forest).
2. For each chosen feature, draw a *random threshold* uniformly from its observed range — not the optimal split.
3. Split on whichever (feature, random-threshold) pair gives the lowest impurity among these candidates.

Trees are typically fit to the entire training data without bootstrap sampling (`replace=FALSE`); variance reduction comes from the random thresholds instead of the bootstrap. The combination of feature subsampling and threshold randomisation produces highly decorrelated trees, and averaging over many of them produces a low-variance ensemble.

13.15.4 Assumptions

Feature importance is approximately diffuse — random thresholds work poorly when a few dominant informative features need precise split points; in those settings, optimising thresholds (random forest) outperforms.

13.15.5 R Implementation

```
library(ranger)

set.seed(2026)
n <- 500
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n), x4 = rnorm(n),
  y  = sin(2 * rnorm(n)) + 0.3 * rnorm(n)
)

# ranger implements extra-trees via `splitrule = "extratrees"`
rf <- ranger(y ~ ., data = df,
             num.trees = 500, mtry = 2,
             splitrule = "variance")

et <- ranger(y ~ ., data = df,
             num.trees = 500, mtry = 2,
             splitrule = "extratrees",
             num.random.splits = 5)

c(rf_oob_rmse = sqrt(rf$prediction.error),
  et_oob_rmse = sqrt(et$prediction.error))
```

13.15.6 Output & Results

OOB-based RMSE for both random forest and Extra Trees on the same data. Extra Trees typically produces slightly lower RMSE on noisy datasets where variance reduction dominates; random forest typically wins on structured signals where threshold optimisation captures sharper boundaries.

13.15.7 Interpretation

A reporting sentence: "Extra Trees with `num.random.splits=5` produced marginally lower OOB RMSE (0.38) than random forest (0.41) on this noisy regression task; the variance reduction from random thresholds outweighed the small bias cost. Training was approximately 3 × faster than random forest." Always state the number of random splits and compare against random forest baseline.

13.15.8 Practical Tips

- Extra Trees can skip bootstrapping (`replace=FALSE`) for tighter variance; random forest keeps the bootstrap as its primary variance-reduction mechanism.
- With many informative features, tune `num.random.splits` upward (10–20) so the random-threshold candidates include the optimum more often; this approaches random forest behaviour.
- Training is typically 2–5× faster than random forest for the same tree count, because no threshold-optimisation search is performed at each split.
- Use random forest when a few features dominate the signal; use Extra Trees when signal is diffuse across many features.
- For variable importance, use permutation importance (`importance="permutation"`); impurity-based importance is biased in Extra Trees just as in random forests.
- For very large datasets, Extra Trees' speed advantage compounds; consider it as the default ensemble when training time matters.

13.15.9 R Packages Used

`ranger` with `splitrule="extratrees"` for the canonical implementation; `extraTrees` package for the Python-compatible reference implementation; `parsnip::rand_forest()` (with engine selection) for tidymodels integration; `caret` for cross-validated tuning of `num.random.splits`.

13.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.15.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.16 Feature Engineering

13.16.1 Introduction

Feature engineering – creating informative inputs – is often the largest driver of predictive performance. Even deep learning benefits from domain-informed features. Standard techniques: log/sqrt transforms, interactions, polynomial bases, categorical encodings, date features, and lag features for time series.

13.16.2 Prerequisites

Regression / classification basics; understanding the target's scale and distribution.

13.16.3 Theory

Key transformations: - **Log / sqrt** stabilise right-skewed variables. - **Box-Cox / Yeo-Johnson** learn the optimal power transform. - **Polynomial and spline bases** expose non-linear relationships to linear models. - **Interactions** $(x_1 x_2)$ surface effects whose direction depends on another variable. - **Target encoding** for high-cardinality categoricals: replace category with smoothed target mean. - **Lag / rolling features** for time series.

13.16.4 Assumptions

Transformations preserve monotonicity where needed; target encoding uses out-of-fold means to avoid leakage.

13.16.5 R Implementation

```
library(recipes); library(embed)

set.seed(2026)
n <- 300
df <- tibble::tibble(
  x1 = exp(rnorm(n)),                    # right-skewed
  x2 = runif(n, 0, 10),
  cat = sample(letters[1:20], n, replace = TRUE),
  date = as.Date("2024-01-01") + sample(0:364, n, replace = TRUE),
  y = runif(n)
)

rec <- recipe(y ~ ., data = df) %>%
  step_log(x1) %>%
  step_interact(~ x1:x2) %>%
  step_ns(x2, deg_free = 4) %>%
```

```
  step_date(date, features = c("dow", "month")) %>%
  step_holiday(date) %>%
  step_rm(date) %>%
  step_lencode_mixed(cat, outcome = vars(y))   # mixed-effects target
   ↪  encoding

baked <- prep(rec) %>% bake(new_data = NULL)
colnames(baked)[1:10]
```

13.16.6 Output & Results

A transformed data frame with log-x1, interaction term, natural spline basis for x2, day-of-week / month features, and smoothly target-encoded category.

13.16.7 Interpretation

"Feature engineering added 12 derived columns: log-transformed skew, spline bases, date features, and target-encoded category. The pipeline is reproducible and applied identically to train and test."

13.16.8 Practical Tips

- Do feature engineering inside a `recipe` (or equivalent) to avoid leakage.
- Log-transform right-skewed variables before linear models; trees handle skew fine.
- High-cardinality categoricals benefit hugely from target encoding (with proper smoothing / CV handling).
- Date features (day-of-week, hour, month, holiday) are often critical for temporal prediction.
- Don't over-engineer – more features means more noise; cross-validate additions.

13.16.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.16.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting

- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.17 Feature Selection

13.17.1 Introduction

Feature selection retains the predictors most relevant to the target while discarding the rest. The benefits are simpler models, reduced over-fitting, faster inference, improved interpretability, and reduced annotation cost in some applications. Three families of methods coexist: filter (univariate tests, cheap and fast), wrapper (fit-and-evaluate, slow but thorough), and embedded (selection happens during fitting, balancing speed and quality). Modern ML pipelines almost always use embedded methods (lasso, tree importance, Boruta) because they integrate naturally with cross-validation.

13.17.2 Prerequisites

A working understanding of regression and classification basics, regularisation (L1/lasso), and the bias-variance trade-off.

13.17.3 Theory

Filter methods rank features by univariate statistics — Pearson or Spearman correlation, mutual information, chi-squared, ANOVA F-statistic — and select the top k. Fast but ignore feature interactions and joint information.

Wrapper methods treat the model as a black box, fitting and evaluating across feature subsets. Forward selection adds features one at a time; backward elimination starts with all features and removes them; recursive feature elimination (RFE) cycles through both with cross-validation. Slow but capable of finding complementary feature combinations.

Embedded methods integrate selection with fitting. Lasso drives some coefficients to exactly zero (selection); tree-based methods (random forest, XGBoost) provide importance scores; Boruta compares each feature against random permuted shadows in many resamples to test for genuinely predictive features. Stability selection runs lasso on many subsamples and ranks features by selection frequency.

13.17.4 Assumptions

Selected features generalise from training to deployment data; selection is done within CV folds to prevent leakage of test-fold information into the chosen

feature set.

13.17.5 R Implementation

```r
library(Boruta); library(glmnet)

set.seed(2026)
n <- 300; p <- 20
X <- matrix(rnorm(n * p), n, p)
colnames(X) <- paste0("x", 1:p)
y <- X %*% c(rep(2, 5), rep(0, 15)) + rnorm(n)

# Embedded: Lasso selects non-zero features
lasso <- cv.glmnet(X, y, alpha = 1)
nonzero_lasso <- rownames(coef(lasso, s = "lambda.min"))[
  which(as.vector(coef(lasso, s = "lambda.min")) != 0)
][-1]
cat("Lasso selected:", length(nonzero_lasso), "\n")

# Wrapper-ish: Boruta compares features to shadow permutations
bor <- Boruta(X, y, maxRuns = 50, doTrace = 0)
confirmed <- getSelectedAttributes(bor, withTentative = FALSE)
cat("Boruta confirmed:", length(confirmed), "\n")
```

13.17.6 Output & Results

Lasso retains the features whose coefficients survive the L1 penalty at the cross-validation-optimal λ. Boruta confirms or rejects features based on whether their importance exceeds that of randomly permuted shadow features across many resampling runs. On the sparse-signal dataset (5 true features, 15 noise), both methods typically recover 4–5 true features with near-zero false positives.

13.17.7 Interpretation

A reporting sentence: "Lasso (cross-validated $\lambda = 0.08$) and Boruta (50 runs, BH-adjusted importance against shadow features) independently identified the same 5 predictive features matching the data-generating model, with no false positives in either; correlation-based filter selection (top 5 by absolute correlation with y) missed one feature because its marginal correlation was masked by noise from collinear predictors." Always describe the selection method and any cross-validation embedding.

13.17.8 Practical Tips

- Perform feature selection *inside* cross-validation folds; selection on the full dataset before CV leaks information from test folds into the chosen feature

set.
- Lasso is fast and principled for linear or near-linear relationships; random forest importance handles non-linear effects better.
- Boruta's multiple-run shadow-feature comparison is statistically cleaner than single-pass importance and provides confidence statements about selection.
- For $p > n$ regimes, lasso (or elastic net) is nearly mandatory; random forest importance alone becomes unreliable in very high-dimensional settings.
- Stability selection (`stabs::stabsel`) fits many lassos on data subsamples and ranks features by their selection frequency; this is a robust alternative when a single CV run is unstable.
- Pre-screen with cheap filter methods (variance, correlation) when p is in the millions; this reduces the candidate set before applying the more expensive wrapper or embedded methods.

13.17.9 R Packages Used

`glmnet` for lasso and elastic-net embedded selection; `Boruta` for shadow-feature-based confirmation; `stabs` for stability selection; `caret::rfe()` for recursive feature elimination; `vip` for permutation-based variable importance from any model; `recipes::step_select_*` for tidymodels-flavoured selection within pre-processing pipelines.

13.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.17.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.18 Feedforward Networks in torch

13.18.1 Introduction

`torch` for R is the native R port of PyTorch: an automatic-differentiation library with GPU support, a comprehensive layer library, and modern optimisers (Adam, AdamW). It enables building arbitrary neural-network architectures from scratch and training them with the same APIs Python users know. The companion package `luz` wraps `torch` with a scikit-learn-style `fit`/`predict` interface, abstracting away the manual training loop for routine cases.

13.18.2 Prerequisites

A working understanding of neural networks (forward propagation, backpropagation), gradient descent, the chain rule, and basic mini-batch training concepts.

13.18.3 Theory

A `torch` model is an `nn_module` — a composition of parameterised layers with a forward-pass method. Training iterates over epochs and mini-batches:

1. **Forward pass**: compute predictions and loss on a mini-batch.
2. **Backward pass**: compute gradients via reverse-mode automatic differentiation.
3. **Optimiser step**: update parameters using the gradient (Adam adjusts step sizes per parameter).

Modern training loops add learning-rate scheduling (cosine annealing, warmup), gradient clipping, weight decay, and early stopping. `luz` packages these patterns into named callbacks.

13.18.4 Assumptions

Sufficient data for the chosen model capacity (overparameterised models on tiny datasets memorise rather than generalise); features standardised before input; mini-batch size appropriate for available memory and the bias-variance trade-off.

13.18.5 R Implementation

```
library(torch)

set.seed(2026)
torch_manual_seed(2026)

# Simulate a simple regression task
n <- 1000
```

```
x <- torch_randn(n, 3)
y <- (x[, 1] * 2 + x[, 2]^2 - x[, 3] * 0.5)$unsqueeze(2)

net <- nn_module(
  initialize = function() {
    self$fc1 <- nn_linear(3, 16)
    self$fc2 <- nn_linear(16, 16)
    self$fc3 <- nn_linear(16, 1)
  },
  forward = function(x) {
    x %>% self$fc1() %>% nnf_relu() %>%
          self$fc2() %>% nnf_relu() %>%
          self$fc3()
  }
)

model <- net()
opt   <- optim_adam(model$parameters, lr = 1e-2)

for (epoch in 1:50) {
  opt$zero_grad()
  pred <- model(x)
  loss <- nnf_mse_loss(pred, y)
  loss$backward()
  opt$step()
  if (epoch %% 10 == 0)
    cat(sprintf("epoch %d  loss %.4f\n", epoch, loss$item()))
}
```

13.18.6 Output & Results

Per-epoch MSE loss should decrease monotonically as the network learns the non-linear regression. The 3-layer MLP with ReLU activations rapidly fits the simulated quadratic-in-x_2 structure that linear regression cannot capture.

13.18.7 Interpretation

A reporting sentence: "A 3-layer feedforward network (16-16-1 hidden units, ReLU activations, Adam optimiser, learning rate 0.01) trained for 50 epochs reduced MSE from 3.1 to 0.04 on the simulated regression with quadratic interactions; predictions closely tracked the noisy targets, with residuals consistent with the irreducible simulation noise level." Always state the architecture, optimiser, learning rate, and training epochs.

13.18.8 Practical Tips

- Use `luz::fit()` for a scikit-learn-style interface that handles the training loop, dataloaders, and standard callbacks (early stopping, model checkpointing, learning-rate scheduling).
- Move tensors and models to GPU with `$to(device="cuda")`; check `cuda_is_available()` first to handle CPU-only systems gracefully.
- Mini-batching via `dataloader()` is essential beyond toy problems; full-batch gradient descent is computationally inefficient and produces noisier optimisation trajectories than expected.
- Monitor training and validation loss separately; early-stop when validation loss plateaus or starts to rise to prevent over-fitting.
- `torchvision` and `torchaudio` add pre-trained models for transfer learning; these dramatically reduce training time and data requirements for common tasks (image classification, audio recognition).
- For tabular data, gradient-boosted trees (XGBoost, LightGBM) usually outperform feedforward networks; deep learning shines on raw modalities (images, text, audio) where structure is encoded in the data layout.

13.18.9 R Packages Used

`torch` for the canonical PyTorch-like API in R; `luz` for sklearn-style `fit`/`predict`; `torchvision`, `torchaudio`, `tabnet` for domain-specific models; `torcheval` for evaluation metrics; `coro` for Python-style generators (used internally by `dataloader`).

13.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.18.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.19 Gradient Boosting

13.19.1 Introduction

Gradient boosting, formalised by Friedman in 2001, builds an additive ensemble $F_M(x) = \sum_{m=1}^{M} \nu h_m(x)$ by fitting each weak learner h_m to the negative gradient of the loss at the current predictions. Decision trees are the canonical weak learner, the learning rate ν shrinks each tree's contribution to control regularisation, and the resulting framework dominates structured tabular-data benchmarks. Modern implementations (XGBoost, LightGBM, CatBoost) add second-order information, regularisation, missing-value handling, and other improvements over the original Friedman gradient boosting.

13.19.2 Prerequisites

A working understanding of decision trees, gradient descent, and the bias-variance trade-off in ensemble methods.

13.19.3 Theory

The generic gradient-boosting algorithm:

1. Initialise $F_0(x) = \arg\min_\gamma \sum_i L(y_i, \gamma)$ — the constant prediction minimising the loss.
2. For $m = 1, \dots, M$:
   - Compute pseudo-residuals $r_{im} = -\partial L(y_i, F(x_i))/\partial F(x_i)|_{F=F_{m-1}}$.
   - Fit a weak learner h_m to the pseudo-residuals.
   - Update $F_m(x) = F_{m-1}(x) + \nu h_m(x)$.

For squared-error loss the pseudo-residuals are ordinary residuals; for binary log-loss they are score residuals; for Huber loss they combine the two regimes. The learning rate ν trades fit quality for over-fit risk: smaller ν requires more trees but generalises better.

13.19.4 Assumptions

The chosen weak learner is appropriate for the loss (shallow trees with depth 3–6 for most losses); the learning rate is small enough for stable optimisation; sufficient sample size for the boosting interactions to be supported by the data.

13.19.5 R Implementation

```
library(gbm)

set.seed(2026)
n <- 500
df <- data.frame(
```

```
  x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n),
  y  = sin(rnorm(n)) + 0.5 * rnorm(n)^2
)

m <- gbm(y ~ ., data = df,
         distribution = "gaussian",
         n.trees = 2000,
         interaction.depth = 3,
         shrinkage = 0.01,
         cv.folds = 5,
         verbose = FALSE)

best_iter <- gbm.perf(m, method = "cv", plot.it = FALSE)
cat("Best iter:", best_iter, "\n")
summary(m, n.trees = best_iter, plotit = FALSE)
```

13.19.6 Output & Results

`gbm()` with `cv.folds` returns the cross-validated boosting iteration count along with the variable-importance summary. `gbm.perf()` finds the iteration that minimises CV error; predictions at that iteration count are typically used for deployment.

13.19.7 Interpretation

A reporting sentence: "Gradient boosting with `shrinkage=0.01`, `interaction.depth=3`, and 5-fold cross-validated early stopping converged at 1{,}240 trees (CV RMSE 0.34); variable-importance scores attributed 52 % of relative influence to x_1, 28 % to x_2, and 20 % to x_3, consistent with the simulated data structure." Always state the shrinkage, depth, CV-optimal iteration count, and importance scores.

13.19.8 Practical Tips

- Smaller learning rate (0.01–0.05) with many trees almost always generalises better than larger rate with few trees; this is the canonical regularisation trade-off in boosting.
- Use shallow trees (depth 3–6) as the standard weak learners; deeper trees approach a single-tree model and lose the boosting benefit.
- `cv.folds=5` in `gbm()` produces an early-stopping estimate without separate plumbing; `gbm.perf(method="cv")` finds the optimal iteration count.
- For structured prediction problems, prefer XGBoost, LightGBM, or CatBoost over base `gbm`; they add regularisation, missing-value handling, GPU support, and substantially faster training.
- Monotone constraints (supported by XGBoost and LightGBM) embed

domain knowledge into boosting; useful in regulated industries (credit, insurance, healthcare).
- Pair gradient boosting with feature-importance and partial-dependence plots for interpretation; raw boosted trees are otherwise opaque.

13.19.9 R Packages Used

`gbm` for the original Friedman gradient boosting; `xgboost`, `lightgbm`, `catboost` for modern alternatives with regularisation and speed; `parsnip::boost_tree()` for tidymodels integration; `mboost` for component-wise gradient boosting on additive models.

13.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.19.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.20 Isolation Forest

13.20.1 Introduction

Isolation Forest (Liu, Ting, and Zhou, 2008) is a tree-based unsupervised anomaly-detection method built on a single intuition: outliers are easier to isolate by random splits than densely packed normal points. The algorithm builds many random trees, partitions the data with random feature-and-split-value choices, and uses each observation's average path length to a terminal node as its anomaly score. Short paths mean easy isolation, which means the point is anomalous; long paths mean the point sits in dense regions that take many splits to isolate.

13.20.2 Prerequisites

A working understanding of decision trees, random partitioning, and basic anomaly-detection concepts.

13.20.3 Theory

Build T isolation trees: at each node, pick a random feature and a random split value uniformly in $[\min, \max]$ for that feature. Tree depth is capped at $\log_2 n$ for computational efficiency. For an observation x, the anomaly score is

$$s(x) = 2^{-\mathbb{E}[h(x)]/c(n)},$$

where $h(x)$ is the average path length across trees and $c(n)$ is a normalising constant equal to the average path length of an unsuccessful BST search on n points. Scores close to 1 indicate clear anomalies; scores around 0.5 indicate normal points; scores well below 0.5 are deep in dense regions.

13.20.4 Assumptions

Anomalies are few and distinct from the dense normal points; the subsampling size is sufficient to isolate them ($\psi = 256$ is the canonical default and works well in practice).

13.20.5 R Implementation

```
library(solitude)

set.seed(2026)
n <- 300
df <- data.frame(
  x1 = c(rnorm(n), runif(20, -5, 5)),
  x2 = c(rnorm(n), runif(20, -5, 5))
)

iforest <- isolationForest$new(num_trees = 200)
iforest$fit(df)

scores <- iforest$predict(df)$anomaly_score
flagged <- scores > quantile(scores, 0.9)

plot(df, col = ifelse(flagged, "red", "grey60"),
     pch = 16, asp = 1,
     main = "Isolation Forest: top-10% scored observations flagged",
     xlab = "x1", ylab = "x2")
```

13.20.6 Output & Results

`isolationForest$predict()` returns a tibble with anomaly scores per observation. Thresholding at a chosen quantile (e.g., the 90th percentile) flags the candidates; the scatter shows flagged points in red and normal points in grey.

13.20.7 Interpretation

A reporting sentence: "Isolation Forest with 200 trees and default subsampling scored 14 of 20 injected anomalies (sensitivity 0.70) at the 90th-percentile threshold, with 16 false positives among the 300 normal points (false-positive rate 5.3 %); the score distribution showed a clear bimodal structure separating anomalies from normal points." Always state the number of trees, threshold, and per-class accuracy.

13.20.8 Practical Tips

- Tune the contamination threshold (top-k percentile) to match the expected anomaly rate; a 5 % threshold for typical fraud detection, 1 % for very rare events.
- `num_trees=100-200` is usually sufficient; returns diminish beyond 200 and computational cost grows linearly.
- Isolation Forest handles high-dimensional data well because it does not rely on distance computations; this is one of its main advantages over distance-based methods (kNN-distance, LOF).
- Excellent default choice for anomaly detection; requires no feature scaling and has few hyperparameters to tune.
- For improved separation in non-axis-aligned anomaly structures, Extended Isolation Forest adds random oblique splits; available via the `isotree` package.
- Pair anomaly scores with domain expertise; flagged observations should be investigated, not automatically removed or actioned.

13.20.9 R Packages Used

`solitude` for the canonical R6-based isolation forest interface; `isotree` for Extended Isolation Forest with oblique splits and additional anomaly-detection variants; `dbscan::lof()` for Local Outlier Factor as a distance-based alternative; `mlr3` and `tidymodels` for cross-validated anomaly detection with labelled data.

13.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

13.20.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.21 Isotonic Regression Calibration

13.21.1 Introduction

Isotonic regression fits a monotone non-decreasing step function from classifier scores to observed probabilities. It is the non-parametric counterpart to Platt scaling: more flexible (no sigmoid assumption required) but requiring more validation data and more prone to over-fitting. It is the standard post-hoc calibration method for tree-ensemble classifiers (random forest, gradient boosting), whose mis-calibration patterns are often non-sigmoidal.

13.21.2 Prerequisites

A working understanding of calibration plots, the pool-adjacent-violators (PAV) algorithm, and Platt scaling as the parametric alternative.

13.21.3 Theory

Given sorted score-label pairs (f_i, y_i), isotonic regression finds the non-decreasing step function $\hat{g}$ minimising $\sum_i (y_i - \hat{g}(f_i))^2$. The pool-adjacent-violators algorithm solves this in $O(n)$ time: it walks through the sorted scores, pooling adjacent observations into single steps whenever monotonicity is violated, until the result is non-decreasing.

The fitted step function maps any future score to the observed probability rate within the matching score range from the validation set. The non-parametric flexibility allows fitting any monotone calibration curve, but at the cost of statistical efficiency relative to Platt's two-parameter sigmoid.

13.21.4 Assumptions

The mis-calibration is monotone (almost always the case for well-trained classifiers); enough validation data to estimate the step function without over-fitting (rule of thumb: at least 500 examples).

13.21.5 R Implementation

```
set.seed(2026)
n <- 1000
y <- rbinom(n, 1, 0.3)
f <- 2 * y + rnorm(n, 0, 1.2)

val_idx <- 1:500
te_idx  <- 501:1000

# Isotonic regression on validation; apply to test
iso <- isoreg(f[val_idx], y[val_idx])
iso_fn <- stepfun(sort(f[val_idx]), c(0, iso$yf))
iso_p  <- iso_fn(f[te_idx])
iso_p  <- pmin(pmax(iso_p, 0), 1)

naive_p <- plogis(f[te_idx])
platt   <- glm(y[val_idx] ~ f[val_idx], family = "binomial")
platt_p <- plogis(coef(platt)[1] + coef(platt)[2] * f[te_idx])

c(naive = mean((naive_p - y[te_idx])^2),
  platt = mean((platt_p - y[te_idx])^2),
  isotonic = mean((iso_p  - y[te_idx])^2))
```

13.21.6 Output & Results

Brier scores for the three methods: naive sigmoid (no calibration), Platt scaling (parametric sigmoid), and isotonic regression (non-parametric step function). Isotonic typically wins when the underlying calibration curve is non-sigmoidal.

13.21.7 Interpretation

A reporting sentence: "Isotonic regression on the held-out test set produced the lowest Brier score (0.168) compared with Platt scaling (0.181) and the naive sigmoid (0.213); isotonic captured a non-sigmoidal calibration curve that the parametric sigmoid could not. ROC AUC was unchanged across all three methods because the rescaling preserves rank." Always compare both calibration methods and report Brier score plus a reliability diagram.

13.21.8 Practical Tips

- Isotonic regression needs more validation data than Platt scaling (rule of thumb: at least 500 examples; for very imbalanced data, more for the minority class).
- Prone to over-fit when the validation set is small; the step function can track validation-specific noise, especially in the tails. Cross-validate the

calibration step itself for small validation sets.
- For tree-ensemble classifiers (random forest, GBM, XGBoost), isotonic regression is usually preferred over Platt; their mis-calibration is rarely sigmoidal.
- For SVMs, Platt scaling is usually sufficient; the SVM decision-value distribution is closer to logistic.
- Step-function output produces piecewise-constant probabilities, which can look strange in sensitive applications; spline-smoothed isotonic exists but is rarely implemented in production.
- Always pair calibration with a reliability diagram (`probably::cal_plot`, `gbm::calibrate.plot`) showing pre- and post-calibration curves; verify visually that the calibration improved.

13.21.9 R Packages Used

Base R `isoreg()` for the canonical PAV-based isotonic regression; `caret::calibrate()` and `probably::cal_estimate_isotonic()` for integrated tidymodels-style calibration; `betacal` for the parametric Beta calibration alternative; `mlr3calibration` for calibration extensions in mlr3.

13.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.21.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.22 k-Nearest Neighbours Classification

13.22.1 Introduction

k-nearest-neighbours classification assigns each test observation the majority class among its k nearest training points in feature space. It is the simplest

instance-based learning method: there is no training step beyond storing the data, and prediction is a search-and-vote operation. kNN is competitive on low-dimensional, well-separated data and serves as a useful baseline against which more complex methods are compared. Performance depends critically on the distance metric and on k, both of which require attention.

13.22.2 Prerequisites

A working understanding of distance metrics (Euclidean, Manhattan, Mahalanobis), train-test splits, and the bias-variance trade-off in supervised learning.

13.22.3 Theory

For a query point x^* and training set $\{(x_i, y_i)\}$, the prediction is the mode of the labels of the k nearest neighbours: $\hat{y}(x^*) = \text{mode}\{y_i : x_i \in N_k(x^*)\}$. Distance-weighted variants weight votes inversely by distance, giving closer neighbours more influence and often improving boundary smoothness.

The bias-variance trade-off in k is direct: small k produces high-variance, low-bias decisions (each prediction depends on a few neighbours and adapts to local noise); large k produces high-bias, low-variance decisions (each prediction averages over a larger neighbourhood and misses local structure). Cross-validation chooses the optimal k.

The choice of distance metric matters: Euclidean is the default, Manhattan is more robust to outliers, Mahalanobis accounts for feature covariance, and Gower distance handles mixed numeric-categorical inputs.

13.22.4 Assumptions

Features are appropriately scaled (distances are scale-sensitive); labels are sufficiently clean for local majority voting to be informative; the chosen distance metric reflects meaningful similarity.

13.22.5 R Implementation

```
library(class)

set.seed(2026)
n <- 400
x1 <- rnorm(n); x2 <- rnorm(n)
y <- factor(ifelse(x1^2 + x2^2 < 1, "in", "out"))
df <- data.frame(x1 = scale(x1)[, 1], x2 = scale(x2)[, 1], y = y)

idx <- sample(n, n * 0.7)
train <- df[idx, ]; test <- df[-idx, ]
```

```r
# kNN at different k
accs <- sapply(c(1, 5, 15, 51), function(k) {
  pred <- knn(train[, 1:2], test[, 1:2], train$y, k = k)
  mean(pred == test$y)
})
setNames(accs, paste0("k=", c(1, 5, 15, 51)))
```

13.22.6 Output & Results

Test accuracy across a grid of k values reveals the U-shaped bias-variance curve: very small k overfits training noise, very large k oversmooths the decision boundary, and a moderate k minimises test error.

13.22.7 Interpretation

A reporting sentence: "k-NN classification achieved best test accuracy of 0.91 at $k = 15$ on the circular-decision-boundary task; $k = 1$ over-fit to noise with accuracy 0.82, while $k = 51$ over-smoothed and dropped to 0.80. Cross-validation across k confirmed $k = 15$ as the optimum." Always state the chosen k, the distance metric, and the test-set accuracy.

13.22.8 Practical Tips

- Always standardise features before kNN; unscaled features with different units dominate the distance computation and bias predictions.
- Tune k by cross-validation; the U-shaped accuracy-vs-k curve has a clear optimum on most datasets.
- For high-dimensional data ($d > 20$), kNN suffers the curse of dimensionality: distances concentrate and discriminating power collapses. Reduce dimensionality with PCA or use a parametric / tree-based model.
- `kknn::train.kknn()` implements kernel-weighted kNN (closer neighbours weighted more heavily); often slightly more accurate than uniform-weight kNN.
- kNN is a lazy learner: prediction cost is $O(nd)$ per query for naive search; use `FNN` with kd-tree or ball-tree indexing for $O(\log n)$ queries on large training sets.
- For mixed numeric-categorical inputs, use Gower distance via `cluster::daisy()` instead of Euclidean; the standard distance breaks down on categorical features.

13.22.9 R Packages Used

`class::knn()` for the canonical implementation; `FNN::knn()` for fast kd-tree-based search; `kknn::train.kknn()` for weighted kNN with cross-validated k

and kernel selection; `caret` and `tidymodels` for cross-validated kNN within ML pipelines; `RANN` for approximate nearest-neighbour search at scale.

13.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.22.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.23 k-Nearest Neighbours Regression

13.23.1 Introduction

k-Nearest Neighbours regression predicts a continuous outcome at a query point as the mean (or weighted mean) of the k nearest training targets in feature space. It is the simplest non-parametric regression method, making no assumption about the global functional form and only requiring a distance metric. The trade-off is straightforward: small k produces high variance (each prediction depends on a few neighbours), while large k produces high bias (predictions average over a wide region and miss local structure).

13.23.2 Prerequisites

A working understanding of distance metrics, the bias-variance trade-off, and train-test evaluation in supervised learning.

13.23.3 Theory

The kNN regression prediction at query point x^* is

$$\hat{y}(x^*) = \frac{1}{k} \sum_{i \in N_k(x^*)} y_i,$$

where $N_k(x^*)$ is the set of indices of the k training points nearest to x^*. Distance-weighted variants use weights $w_i \propto 1/\|x^* - x_i\|$ to give closer neighbours more influence than far ones, producing smoother predictions and reducing the discontinuities that plain kNN exhibits at the boundaries between Voronoi cells.

The bias-variance trade-off in k is direct: $k = 1$ memorises training noise (high variance, low bias); $k = n$ predicts the global mean (low variance, high bias). Cross-validation chooses the optimal k for the data.

13.23.4 Assumptions

Features are appropriately scaled (unscaled features with different units dominate the distance computation); the regression surface is locally smooth (kNN cannot extrapolate or model abrupt changes well).

13.23.5 R Implementation

```
library(FNN)

set.seed(2026)
n <- 400
x <- runif(n, 0, 2 * pi)
y <- sin(x) + rnorm(n, 0, 0.3)
df <- data.frame(x = scale(x)[, 1], y = y)

idx <- sample(n, n * 0.7)
train <- df[idx, ]; test <- df[-idx, ]

rmses <- sapply(c(1, 5, 15, 51), function(k) {
  pred <- knn.reg(train = train[, "x", drop = FALSE],
                  test  = test[,  "x", drop = FALSE],
                  y = train$y, k = k)$pred
  sqrt(mean((test$y - pred)^2))
})
setNames(rmses, paste0("k=", c(1, 5, 15, 51)))
```

13.23.6 Output & Results

Test RMSE across a grid of k values reveals the U-shaped bias-variance curve: very small k overfits noise, very large k oversmooths the sinusoid, and a moderate k (around 15 in this example) minimises test error.

13.23.7 Interpretation

A reporting sentence: "k-NN regression with $k = 15$ achieved test RMSE of 0.31 on the held-out 30 % of the simulated sinusoid, matching the irreducible noise

level ($\sigma = 0.3$); $k = 1$ over-fitted to RMSE 0.42, $k = 51$ under-fitted to RMSE 0.45. Cross-validation across k confirmed $k = 15$ as the optimum." Always state how k was selected and the test-set RMSE.

13.23.8 Practical Tips

- Scale features (`scale()` or `recipes::step_normalize()`); unscaled features with different units dominate the distance computation and produce biased predictions.
- Use distance-weighted kNN (`kknn::train.kknn`) for smoother predictions and slightly better accuracy at the cost of one extra hyperparameter.
- Prediction cost is $O(nd)$ per query for naive search; use kd-trees (default in `FNN::knn.reg`) for faster lookups in low-dimensional feature spaces.
- For high-dimensional data ($d > 20$), kNN's accuracy collapses due to the curse of dimensionality; reduce dimensionality with PCA or use a parametric / tree-based model instead.
- kNN has no native handling of categorical predictors; encode as dummy variables or use Gower distance for mixed numeric-categorical inputs.
- For large datasets, consider approximate nearest-neighbour search (`RANN`, `Annoy`); exact k-NN becomes computationally prohibitive beyond a few million points.

13.23.9 R Packages Used

`FNN::knn.reg()` for fast kd-tree-based exact k-NN regression; `kknn::train.kknn()` for weighted k-NN with cross-validated kernel and k selection; `caret` and `tidymodels` for cross-validated k-NN within the broader ML workflow; `RANN` for approximate nearest-neighbour search at scale.

13.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.23.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines

- FDR, knockoffs, replication crisis

13.24 LightGBM

13.24.1 Introduction

LightGBM (Ke et al., 2017) is a gradient-boosting framework engineered for very large datasets. It introduces several optimisations over XGBoost: histogram-based split finding (substantially faster than sorted-value enumeration), leaf-wise tree growth (deeper trees per leaf count than level-wise), Gradient-based One-Side Sampling (GOSS) that prioritises high-gradient instances during training, and Exclusive Feature Bundling (EFB) that combines mutually exclusive sparse features. The result is typically 3–5× faster than XGBoost with comparable or slightly better accuracy on large tabular data.

13.24.2 Prerequisites

A working understanding of gradient boosting, XGBoost or similar boosted-tree workflow, and the bias-variance trade-off in tree depth and leaf count.

13.24.3 Theory

Three core innovations distinguish LightGBM from earlier gradient-boosting libraries:

- **Histogram binning**: pre-compute feature-value bins (default 255); split finding iterates over bins rather than sorted values, giving substantial speed and memory gains.
- **Leaf-wise growth**: at each iteration, expand the leaf with maximum loss reduction. Produces deeper, asymmetric trees given a fixed leaf count compared to level-wise growth, but at a higher over-fit risk.
- **GOSS**: sample instances by gradient magnitude, keeping the top-a fraction with largest gradients and a random sample of the rest. The classifier focuses on the hard examples that contribute most to the gradient.

13.24.4 Assumptions

Sufficient data for histogram bins to be informative; regularisation (`max_depth`, `num_leaves`, `min_data_in_leaf`) is tuned to prevent the over-fit risk inherent in leaf-wise growth.

13.24.5 R Implementation

```
library(lightgbm)

set.seed(2026)
```

```
n <- 600
X <- matrix(rnorm(n * 5), n, 5)
y <- as.numeric(plogis(X[, 1] + 0.5 * X[, 2]^2) > runif(n))

idx <- sample(n, n * 0.8)
d_tr <- lgb.Dataset(X[idx, ], label = y[idx])
d_te <- lgb.Dataset.create.valid(d_tr, X[-idx, ], label = y[-idx])

params <- list(objective = "binary", metric = "binary_logloss",
               learning_rate = 0.05, num_leaves = 31,
               feature_fraction = 0.8, bagging_fraction = 0.8,
               bagging_freq = 5, min_data_in_leaf = 20)

model <- lgb.train(params, d_tr,
                   nrounds = 500, valids = list(val = d_te),
                   early_stopping_rounds = 20, verbose = -1)

cat("Best iter:", model$best_iter,
    "   best logloss:", round(model$best_score, 4), "\n")

imp <- lgb.importance(model, percentage = TRUE)
head(imp)
```

13.24.6 Output & Results

`lgb.train()` returns a fitted model with the early-stopping iteration, best validation metric, and tree structure. `lgb.importance()` ranks features by gain (or split count, or cover) with optional percentage display.

13.24.7 Interpretation

A reporting sentence: "LightGBM with `learning_rate=0.05`, `num_leaves=31`, feature and bagging fractions 0.8 converged at 184 boosting rounds with early stopping (validation log-loss 0.39); feature 1 contributed 53 % of total gain, confirming it as the dominant predictor in line with the simulated data-generating process. Training was approximately 4× faster than XGBoost on the same data." Always state hyperparameters and early-stopping criteria.

13.24.8 Practical Tips

- Constrain `num_leaves` (default 31) or `max_depth` to prevent the over-fit risk of leaf-wise growth; deeper trees are more accurate per round but more prone to over-fitting.
- LightGBM is typically 3–5× faster than XGBoost for the same number of trees; the speed gap widens further for $>$ 1M rows.

- For categorical features, pass integer-encoded factors plus `categorical_feature` argument; LightGBM uses split-aware categorical handling that often outperforms one-hot encoding.
- A `learning_rate` of 0.01–0.05 is a sensible starting range; smaller values need more rounds but generalise better, larger values train faster.
- GPU training (`device_type="gpu"`) provides a drop-in 3–10× acceleration for very large data; CPU performance is already excellent.
- Compare LightGBM and XGBoost on the same data; results are usually very close, but the better model varies by task.

13.24.9 R Packages Used

`lightgbm` for the canonical R interface; `xgboost` for the closely-related alternative; `catboost` for the third major gradient-boosting library (with native categorical handling); `parsnip::boost_tree(engine="lightgbm")` for tidymodels integration; `bonsai` for the tidymodels-LightGBM bridge.

13.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.24.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.25 LIME Explanations

13.25.1 Introduction

LIME (Ribeiro, Singh, Guestrin 2016) explains a single prediction by fitting a sparse linear model locally around the point of interest. Samples are perturbed near the observation, labelled with the black-box model, and weighted by distance; the linear surrogate's coefficients describe the local behaviour of the model.

13.25.2 Prerequisites

Regression / classification; weighted linear regression.

13.25.3 Theory

For observation x: 1. Sample x'_k by perturbing x in feature space. 2. Predict with the black-box: $\hat{y}_k = f(x'_k)$. 3. Weight samples by proximity: $w_k = \exp(-d(x, x'_k)^2/\sigma^2)$. 4. Fit a sparse linear model g to $(x'_k, \hat{y}_k)$ with weights w_k. 5. Coefficients of g are the local explanation.

13.25.4 Assumptions

A local linear approximation captures the model's behaviour in a neighbourhood; perturbation scheme is appropriate for the feature type (Gaussian for continuous, resampling for categorical, word removal for text).

13.25.5 R Implementation

```
library(lime); library(ranger)

set.seed(2026)
n <- 500
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n),
  y  = factor(sample(c("a", "b"), n, replace = TRUE))
)
df$y <- factor(ifelse(plogis(0.7 * df$x1 - 0.4 * df$x2) > runif(n), "a",
 ↪  "b"))

rf <- ranger(y ~ ., data = df, num.trees = 200, probability = TRUE,
             classification = TRUE)

# LIME setup
explainer <- lime(df[1:450, -4], rf,
                  bin_continuous = TRUE, n_bins = 4)
expl      <- explain(df[451:452, -4],
                     explainer, labels = "a",
                     n_features = 3, n_permutations = 1000)

expl[, c("feature", "feature_weight", "feature_desc")]
```

13.25.6 Output & Results

Per-observation feature weights: how strongly each feature pushes the prediction toward the explained class, with a human-readable feature description.

13.25.7 Interpretation

"LIME attributed observation 1's predicted class 'a' to $x_1 > 0.8$ (weight 0.42) and $x_2 < -0.3$ (weight 0.18), providing an interpretable local rule."

13.25.8 Practical Tips

- LIME is unstable: different runs can produce different explanations due to random perturbations; set a seed and run several times.
- LIME's linear surrogate is a local approximation; for globally highly non-linear models, explanations drift across neighbourhoods.
- Compare LIME with SHAP on the same observations; large disagreements indicate explanation fragility.
- For text, LIME removes words and recomputes predictions; for tabular, use binned perturbation.
- Weighted sampling bandwidth σ strongly affects results; tune by inspection.

13.25.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.25.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.26 Linear Discriminant as ML

13.26.1 Introduction

Linear Discriminant Analysis (LDA), introduced by Ronald Fisher in 1936, is one of the oldest classification methods and remains a strong baseline. It models each class as a multivariate Gaussian with a class-specific mean but a *shared* covariance matrix, and assigns each observation to the class with the highest posterior probability under those Gaussians. The resulting decision boundary

is linear in the predictors. LDA is simple, well-calibrated by construction, and often competitive with logistic regression on small to moderate-sized datasets where class-conditional Gaussianity approximately holds.

13.26.2 Prerequisites

A working understanding of multivariate Gaussian distributions, Bayes' rule for posterior class probabilities, and basic linear algebra (matrix inversion).

13.26.3 Theory

Under class-conditional multivariate Normality with shared covariance Σ and class priors π_k, the posterior is

$$P(y = k \mid x) \propto \pi_k \, \phi(x; \mu_k, \Sigma).$$

Taking the log and dropping constants common to all classes yields the linear discriminant function

$$\delta_k(x) = x^\top \Sigma^{-1} \mu_k - \tfrac{1}{2} \mu_k^\top \Sigma^{-1} \mu_k + \log \pi_k,$$

which is linear in x. The decision boundary between any two classes is thus a hyperplane. Quadratic Discriminant Analysis (QDA) relaxes the shared-covariance assumption to give class-specific Σ_k, producing quadratic boundaries at the cost of more parameters.

Fisher's geometric derivation arrives at the same linear projection by maximising the ratio of between-class to within-class scatter; this connection makes LDA both a classifier and a supervised dimensionality-reduction tool.

13.26.4 Assumptions

Class-conditional features are multivariate Normal with a shared covariance; $n > p$ for stable covariance estimation (use shrinkage LDA when p approaches n).

13.26.5 R Implementation

```
library(MASS)

set.seed(2026)
n <- 300
x1 <- c(rnorm(n/2, 0), rnorm(n/2, 2))
x2 <- c(rnorm(n/2, 0), rnorm(n/2, 2))
y <- factor(rep(c("a", "b"), each = n/2))
```

```r
df <- data.frame(x1, x2, y)

idx <- sample(n, n * 0.7)
lda_fit <- lda(y ~ ., data = df[idx, ])
pred <- predict(lda_fit, df[-idx, ])$class
mean(pred == df[-idx, "y"])

# Coefficients of the linear discriminant function
lda_fit$scaling
```

13.26.6 Output & Results

`lda()` returns the prior probabilities, class means, scaling matrix (linear discriminant coefficients), and proportion of trace each axis explains. `predict()` provides class assignments and posterior probabilities. The linear coefficients, when interpreted geometrically, point in the direction of greatest between-class separation after within-class whitening.

13.26.7 Interpretation

A reporting sentence: "LDA achieved 89 % test accuracy on the two-class Gaussian problem with shared covariance ($n_{\text{train}} = 210$, $n_{\text{test}} = 90$); the linear-discriminant direction (0.82, 0.57) corresponds to the between-class mean difference after within-class scatter whitening; logistic regression gave 87 %, indicating LDA's Gaussian assumption is well-supported." Compare LDA with logistic regression as a sanity check on the Gaussian assumption.

13.26.8 Practical Tips

- When classes are clearly Gaussian with similar covariances, LDA is nearly Bayes-optimal and outperforms logistic regression in finite samples because of its stronger model.
- For p approaching n, use shrinkage LDA (`klaR::rda()` or the `sda` package); unregularised LDA has unstable covariance estimates with poor predictions.
- LDA's posterior probabilities are well-calibrated by construction when the model holds; no Platt scaling is typically needed.
- QDA relaxes the shared-covariance assumption but needs more data per class; use it when class covariances differ visibly (Box's M test gives a formal check).
- LDA also serves as a supervised dimensionality-reduction tool: project onto the top $K - 1$ discriminants for a low-dimensional class-aware visualisation.
- For high-dimensional sparse data (text, genomics), regularised LDA (`sparseLDA`, `penalizedLDA`) handles $p \gg n$ via L1 or grouped penalties.

13.26.9 R Packages Used

`MASS::lda()` and `qda()` for the canonical implementations; `klaR::rda()` for regularised discriminant analysis; `sda` for shrinkage LDA; `penalizedLDA` for high-dimensional sparse extensions; `caret` and `tidymodels::discrim_linear()` for cross-validated LDA within ML pipelines.

13.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.26.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.27 Logistic Regression as ML

13.27.1 Introduction

Logistic regression models class probability as $P(y = 1 \mid x) = \sigma(w^\top x + b)$, where σ is the logistic function, and fits parameters by minimising the cross-entropy loss. As an ML workhorse it is well-calibrated by construction, interpretable through coefficients on the log-odds scale, regularisable with L1, L2, or elastic-net penalties, and scales to millions of features — making it the standard baseline for text classification, click-through prediction, and medical risk scoring.

13.27.2 Prerequisites

A working understanding of linear models, the cross-entropy / log-loss objective, and regularisation (L1 vs L2 penalties).

13.27.3 Theory

The regularised log-loss objective is

$$L(w) = -\sum_i [y_i \log \sigma(w^\top x_i) + (1 - y_i) \log(1 - \sigma(w^\top x_i))] + \lambda R(w),$$

with $R(w) = \|w\|_2^2$ for L2 (ridge), $\|w\|_1$ for L1 (lasso), or a convex combination for elastic net. Optimisation uses Newton-Raphson (IRLS) for the unregularised case and coordinate descent (`glmnet`) for the L1/elastic-net case. The penalty parameter λ is tuned by cross-validation, with `lambda.min` minimising CV deviance and `lambda.1se` providing a more parsimonious choice within one SE of the minimum.

13.27.4 Assumptions

Log-odds are linear in the predictors; observations are independent; sufficient sample size per feature (rule of thumb: $n > 10p$ for unregularised, less for regularised models).

13.27.5 R Implementation

```r
library(glmnet)

set.seed(2026)
n <- 400; p <- 30
X <- matrix(rnorm(n * p), n, p)
beta <- c(rep(1, 5), rep(0, 25))
y <- rbinom(n, 1, plogis(X %*% beta - 1))

# Cross-validated lasso
cv_fit <- cv.glmnet(X, y, family = "binomial",
                    alpha = 1, nfolds = 10)
plot(cv_fit)

coef_min <- as.vector(coef(cv_fit, s = "lambda.min"))
head(coef_min, 10)
cat("# nonzero:", sum(coef_min != 0) - 1, "\n")
```

13.27.6 Output & Results

`cv.glmnet()` returns the cross-validation deviance curve over λ, the recommended `lambda.min` and `lambda.1se`, and the fitted coefficient paths. Lasso correctly recovered the 5 non-zero coefficients in the data-generating model; the deviance plot shows the U-shaped CV-error curve.

13.27.7 Interpretation

A reporting sentence: "Lasso logistic regression with cross-validated $\lambda_{\min} = 0.03$ retained 5 non-zero features that exactly match the data-generating β; AUC on a held-out test set was 0.84 with calibration intercept -0.02 and slope 0.97 — well-calibrated. The sparser `lambda.1se` solution retained 3 features with AUC 0.79." Always report the regularisation choice and a calibration check.

13.27.8 Practical Tips

- Standardise features before regularisation; `glmnet` does this internally and back-transforms, but explicit pre-scaling makes the workflow more transparent.
- Use `lambda.1se` for a sparser, more interpretable model; `lambda.min` is typically more accurate but less parsimonious.
- Logistic regression produces well-calibrated probabilities for correctly-specified models; SVMs, trees, and neural networks typically need Platt scaling or isotonic regression to recover calibration.
- For massively sparse data (text bags-of-words, click logs), `LiblineaR` is dramatically faster than `glmnet`; it handles millions of features routinely.
- For multi-class problems, `family="multinomial"` in `glmnet` extends the framework natively; alternatively, `nnet::multinom()` for unregularised multinomial regression.
- For very imbalanced classes, combine logistic regression with class weights (`weights=...`) or threshold tuning on a validation set.

13.27.9 R Packages Used

`glmnet::cv.glmnet()` for ridge, lasso, and elastic-net logistic regression; `LiblineaR` for fast sparse linear classifiers; base R `glm(family=binomial)` for unregularised logistic regression; `parsnip::logistic_reg()` for tidymodels-flavoured wrappers; `pROC` for ROC and calibration diagnostics.

13.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.27.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting

- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.28 A Complete mlr3 Workflow

13.28.1 Introduction

`mlr3` is the modern R6-based successor to `mlr`, a feature-complete machine-learning framework that abstracts around five core objects: Tasks (data + target + roles), Learners (algorithms), Resamplings (CV schemes), Tuners (search strategies), and Benchmarks (multi-learner comparisons). Like `tidymodels`, mlr3 provides a unified interface across dozens of underlying engines; unlike tidymodels, it uses R6 object-oriented classes rather than tibbles, giving more flexibility but a steeper learning curve.

13.28.2 Prerequisites

A working understanding of R6 classes, machine-learning pipeline concepts, and cross-validation-based hyperparameter tuning.

13.28.3 Theory

The core mlr3 workflow:

1. **Task**: wraps data with target and feature-role specifications.
2. **Learner**: an algorithm (e.g., `lrn("classif.ranger")` for random forest classification).
3. **Resampling**: a cross-validation scheme (`rsmp("cv",folds=5)` for 5-fold CV).
4. **Tuner with AutoTuner**: wraps a learner plus a search space, inner resampling, and search strategy into a self-tuning learner.
5. **Outer resampling**: evaluate the AutoTuner on outer folds for honest nested-CV performance estimation.

`mlr3pipelines` extends the framework with preprocessing graphs and meta-learners (stacking, bagging) as composable nodes — equivalent in role to tidymodels `recipes` and `workflows`.

13.28.4 Assumptions

Task roles (target, predictors, stratification variables) are correctly specified; hyperparameter search spaces cover plausible ranges; resampling matches the inferential question (random for IID, blocked for time series, spatial for geography).

13.28.5 R Implementation

```
library(mlr3); library(mlr3learners); library(mlr3tuning)

set.seed(2026)
df <- data.frame(
  x1 = rnorm(500), x2 = rnorm(500),
  x3 = rnorm(500), x4 = rnorm(500),
  y  = factor(sample(c("yes", "no"), 500, replace = TRUE))
)

task <- as_task_classif(df, target = "y")

# Tunable random forest
lrn_rf <- lrn("classif.ranger",
              mtry = to_tune(1, 4),
              min.node.size = to_tune(5, 20),
              num.trees = 300)

at <- AutoTuner$new(
  learner    = lrn_rf,
  resampling = rsmp("cv", folds = 3),
  measure    = msr("classif.ce"),
  search_space = NULL,
  terminator = trm("evals", n_evals = 10),
  tuner      = tnr("random_search")
)

outer <- rsmp("cv", folds = 3)
rr <- resample(task, at, outer, store_models = TRUE)
rr$aggregate(msr("classif.ce"))
```

13.28.6 Output & Results

`AutoTuner` performs hyperparameter tuning on inner resampling folds and selects the best configuration. `resample()` evaluates the AutoTuner on outer folds, producing nested-CV estimates of generalisation performance. `rr$aggregate()` returns the chosen metric (classification error here) averaged across outer folds with bootstrap-based standard errors.

13.28.7 Interpretation

A reporting sentence: "Nested 3 × 3 cross-validation with mlr3 AutoTuner (10-evaluation random-search inner tuning of random forest hyperparameters) estimated classification error of 0.25 ± 0.02 for the tuned random forest; the nested wrapping prevents the optimistic bias of tuning and evaluating on the

same folds." Always state the outer/inner CV scheme and the search budget.

13.28.8 Practical Tips

- `AutoTuner` is the right pattern for honest nested cross-validation; tuning and evaluating on the same resampling produces optimistically biased estimates.
- `mlr3pipelines` formalises preprocessing as pipeline operations (PipeOps); the equivalent of tidymodels `recipes`. Compose nodes into Graphs for complex pre/post-processing.
- `mlr3tuningspaces` provides sensible default search spaces per learner; useful when you don't know the right ranges.
- `benchmark_grid()` compares multiple learners on the same task under identical resampling, giving fair head-to-head performance comparisons.
- For big data, use `mlr3spatiotempcv` or `mlr3temporal` for time-aware resamplings; standard random CV leaks information for time-series problems.
- Parallelise tuning and resampling with `future::plan(multisession)`; mlr3 respects the `future` framework.

13.28.9 R Packages Used

`mlr3` for the core framework; `mlr3learners` for the curated learner library; `mlr3tuning` and `mlr3tuningspaces` for hyperparameter optimisation; `mlr3pipelines` for graph-based pipelines; `mlr3viz` for visualisation; `mlr3spatiotempcv` and `mlr3temporal` for time- and space-aware resampling.

13.28.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.28.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.29 Naive Bayes

13.29.1 Introduction

Naive Bayes is the simplest probabilistic classifier and one of the oldest. It applies Bayes' theorem under the strong assumption that features are conditionally independent given the class label. The independence assumption is almost always wrong in practice, yet Naive Bayes performs surprisingly well — often competitively with more sophisticated methods — on text classification, spam filtering, and other high-dimensional sparse problems. It trains in linear time and serves as a strong baseline for any classification pipeline.

13.29.2 Prerequisites

A working understanding of Bayes' theorem, conditional probability, and standard distributions (Gaussian, multinomial, Bernoulli) used as per-feature class-conditional models.

13.29.3 Theory

The classifier is

$$\hat{y}(x) = \arg\max_{y} P(y) \prod_{j=1}^{p} P(x_j \mid y),$$

where $P(y)$ is the class prior (estimated from class frequencies) and $P(x_j \mid y)$ is the per-feature class-conditional density. Three standard variants:

- **Gaussian Naive Bayes**: $P(x_j \mid y)$ is Normal with class-conditional mean and variance, suitable for continuous features.
- **Multinomial Naive Bayes**: for word counts and text classification.
- **Bernoulli Naive Bayes**: for binary features.

Training reduces to computing class frequencies and per-feature class-conditional statistics — extremely fast even on millions of examples.

13.29.4 Assumptions

Features are conditionally independent given the class (almost always violated but the classifier remains useful); the chosen distribution family matches each feature's behaviour.

13.29.5 R Implementation

```
library(e1071)
```

```r
set.seed(2026)
n <- 400
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n),
  x3 = sample(c("A", "B"), n, replace = TRUE),
  y  = factor(sample(c("pos", "neg"), n, replace = TRUE))
)

# Inject a relationship
df$y <- factor(ifelse(df$x1 + 0.5 * df$x2 +
                      as.numeric(df$x3) * 0.5 > runif(n, -1, 1),
                      "pos", "neg"))

idx <- sample(n, n * 0.7)
nb <- naiveBayes(y ~ ., data = df[idx, ])
pred <- predict(nb, df[-idx, ])

mean(pred == df[-idx, "y"])
nb$tables$x1         # Gaussian parameters for x1 per class
```

13.29.6 Output & Results

`naiveBayes()` returns a fitted classifier with per-class priors and per-feature conditional distribution parameters. `predict()` provides class labels and (with `type="raw"`) class probabilities. The accuracy on a held-out fold gives a baseline against which more complex models can be compared.

13.29.7 Interpretation

A reporting sentence: "Gaussian Naive Bayes (with Laplace smoothing for the categorical predictor) achieved 74 % test accuracy, competitive with logistic regression's 76 % at a fraction of the training cost; the per-class feature tables identify x_1 as the most discriminative feature (class-conditional mean difference 0.8)." Always state the variant used and the smoothing parameter for categorical features.

13.29.8 Practical Tips

- Naive Bayes is competitive on high-dimensional sparse data (text classification, spam filtering); often wins at small training sizes where complex models over-fit.
- Apply Laplace smoothing (`laplace=1`) for categorical features to avoid zero-probability cases that would otherwise propagate to make the entire posterior zero.

- Probability calibration is often poor (the independence assumption produces overconfident predictions); apply Platt scaling or isotonic regression if calibrated probabilities are needed.
- For continuous features that are not Gaussian, transform (log, Box-Cox) or bin to ordinal before fitting; alternatively, use a kernel-density NB variant.
- The `naivebayes` package provides faster implementations and supports kernel density estimation for non-Gaussian continuous features.
- For text classification specifically, multinomial NB on TF-IDF features remains a strong baseline; only neural methods reliably beat it on large enough corpora.

13.29.9 R Packages Used

`e1071::naiveBayes()` for the classical implementation; `naivebayes` for faster alternatives with kernel-density and Poisson variants; `klaR::NaiveBayes()` for an interface with cross-validation hooks; `parsnip::naive_Bayes()` for tidymodels integration; `text2vec` and `quanteda` for text feature extraction before NB classification.

13.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.29.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.30 Nested Cross-Validation

13.30.1 Introduction

When hyperparameters are tuned on the same data used for evaluation, performance estimates are optimistically biased — the chosen hyperparameters fit

not only the underlying signal but also the noise in the validation folds, producing CV errors lower than the model achieves on truly new data. Nested cross-validation separates the two concerns: an inner loop tunes hyperparameters on each training fold, and an outer loop evaluates the resulting tuned model on held-out test folds. The outer-loop mean is an honest, low-bias estimate of generalisation error.

13.30.2 Prerequisites

A working understanding of k-fold cross-validation, hyperparameter tuning, and the optimism bias that arises from tuning and evaluating on the same data.

13.30.3 Theory

The standard nested-CV scheme:

- **Outer loop**: K-fold CV. Each fold splits data into outer-training and outer-test sets.
- **Inner loop**: on each outer-training set, run K'-fold CV to pick the best hyperparameter configuration.
- **Outer evaluation**: refit the chosen configuration on the full outer-training set; predict and score on the outer-test set.
- **Aggregate**: average the K outer-test scores; the mean is the generalisation estimate, and the spread quantifies uncertainty.

Typical values: outer $K = 5$ to 10, inner $K' = 5$. Computational cost scales as $K \times K'$ times the cost of fitting one model.

13.30.4 Assumptions

Observations are iid (or the resampling respects the dependence structure for time-series, clustered, or grouped data); all preprocessing (imputation, scaling, feature engineering) is refit within each inner fold to prevent test-set leakage.

13.30.5 R Implementation

```
library(rsample); library(purrr); library(dplyr); library(tidyr)

set.seed(2026)
n <- 300
df <- tibble(x1 = rnorm(n), x2 = rnorm(n),
             y  = 1 + 2 * rnorm(n) + 0.5 * rnorm(n)^2)

outer_folds <- vfold_cv(df, v = 5)
lambdas <- c(0.001, 0.01, 0.1, 1, 10)

fit_ridge <- function(train, lambda) {
```

```
  X <- model.matrix(y ~ x1 + x2, train)[, -1]
  solve(crossprod(X) + lambda * diag(ncol(X))) %*% crossprod(X, train$y)
}
pred_ridge <- function(beta, newdata) {
  as.vector(model.matrix(y ~ x1 + x2, newdata)[, -1] %*% beta)
}

outer_rmse <- map_dbl(outer_folds$splits, function(osp) {
  tr_o <- analysis(osp); te_o <- assessment(osp)

  inner <- vfold_cv(tr_o, v = 5)
  inner_rmse <- map_dfr(lambdas, function(l) {
    errs <- map_dbl(inner$splits, function(isp) {
      b <- fit_ridge(analysis(isp), l)
      sqrt(mean((assessment(isp)$y - pred_ridge(b, assessment(isp)))^2))
    })
    tibble(lambda = l, rmse = mean(errs))
  })
  best_l <- inner_rmse$lambda[which.min(inner_rmse$rmse)]
  b <- fit_ridge(tr_o, best_l)
  sqrt(mean((te_o$y - pred_ridge(b, te_o))^2))
})

c(mean_rmse = mean(outer_rmse), sd_rmse = sd(outer_rmse))
```

13.30.6 Output & Results

A vector of K outer-fold RMSE values; the mean estimates generalisation error and the standard deviation quantifies how much that estimate varies across resampling.

13.30.7 Interpretation

A reporting sentence: "Nested 5×5 cross-validation estimated test RMSE of 1.04 ± 0.08 for the ridge regression with hyperparameter λ tuned on inner folds; single-CV tuning on the same hyperparameter grid would have produced RMSE 0.92 — under-estimating by approximately 12 %, the optimism cost of tuning and evaluating on the same folds." Always report the outer-fold mean and standard deviation, never a single CV estimate when tuning is involved.

13.30.8 Practical Tips

- Fit *all* preprocessing (imputation, scaling, feature engineering, target encoding) inside each inner fold; preprocessing on the full outer-training set leaks information into validation.

- `tidymodels::nested_cv()` and `mlr3::resampling_nested_cv()` formalise the structure and prevent the manual loop errors that nested CV is famous for.
- For small datasets ($n < 200$), repeated nested CV (repeating the outer split with different random seeds and averaging) reduces the outer-fold variance that small samples introduce.
- The final deployment model is tuned once on the *full* dataset with the chosen hyperparameter; nested CV informs uncertainty about generalisation, not the deployed model itself.
- Parallelise the inner loop with `future` or `doParallel`; the outer loop is typically fast enough to leave serial.
- Computational budget can be substantial: 5×5 nested CV with 5-point hyperparameter grid runs $5 \times 5 \times 5 = 125$ model fits; account for this when planning.

13.30.9 R Packages Used

`rsample` for the outer-loop splits; `tidymodels::tune` and `tune_grid()` for nested-CV-aware hyperparameter optimisation; `mlr3::resampling_nested_cv` and `AutoTuner` for mlr3-style nested CV; `caret::train` with `trainControl(method="repeatedcv")` for an older alternative.

13.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.30.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.31 Neural Networks: Introduction

13.31.1 Introduction

Neural networks are compositions of linear maps and non-linear activation functions. A feed-forward network with L layers computes $f(x) = g_L(W_L g_{L-1}(\cdots W_2 g_1(W_1 x + b_1)\cdots) + b_L)$ where W_l are weight matrices, b_l biases, and g_l activation functions. The Universal Approximation Theorem (Cybenko, 1989) shows that even a single hidden layer with enough width can approximate any continuous function. Modern deep learning stacks hundreds of such layers to learn powerful representations for images, text, audio, and tabular data with implicit feature engineering.

13.31.2 Prerequisites

A working understanding of linear algebra (matrix multiplication), gradient descent, the chain rule for backpropagation, and basic machine-learning concepts (loss functions, train-test evaluation).

13.31.3 Theory

Perceptron: $\hat{y} = \text{sign}(w^\top x + b)$. Linear decision boundary; provably cannot solve XOR.

Multi-layer perceptron (MLP): two or more layers of linear + non-linear transformations. Universal approximator with one hidden layer of sufficient width.

Activation functions: - **Sigmoid**: $\sigma(x) = 1/(1 + e^{-x})$, smooth but vanishing gradients in deep nets. - **tanh**: zero-centred sigmoid; better than sigmoid for hidden layers but still suffers vanishing gradients. - **ReLU**: $\max(0, x)$, the modern default for hidden layers; faster training and avoids vanishing gradients in positive regions. - **Leaky ReLU**, **GELU**, **SiLU**: variants that address ReLU's "dead-neuron" problem.

Training: minimise loss (MSE for regression, cross-entropy for classification) by gradient descent (or modern variants Adam, AdamW). Backpropagation computes gradients efficiently via the chain rule.

13.31.4 Assumptions

Sufficient data relative to model capacity (overparameterised models on tiny datasets memorise rather than generalise); features are normalised; loss is differentiable almost everywhere.

13.31.5 R Implementation

```
library(nnet)

set.seed(2026)
n <- 400
x1 <- rnorm(n); x2 <- rnorm(n)
y  <- factor(ifelse(x1^2 + x2^2 > 1.5, "out", "in"))
df <- data.frame(x1 = scale(x1)[, 1], x2 = scale(x2)[, 1], y = y)

idx <- sample(n, n * 0.7)
nn <- nnet(y ~ ., data = df[idx, ],
           size = 5, decay = 1e-3, maxit = 200,
           trace = FALSE)

pred <- predict(nn, df[-idx, ], type = "class")
mean(pred == df[-idx, "y"])
```

13.31.6 Output & Results

`nnet()` fits a small MLP with one hidden layer of `size` neurons. The test accuracy on the concentric-rings task demonstrates the network's ability to learn a non-linear decision boundary that linear classifiers cannot represent.

13.31.7 Interpretation

A reporting sentence: "A 5-neuron single-hidden-layer MLP with weight decay 1e-3 trained for 200 iterations achieved 91 % test accuracy on the concentric-rings classification task; logistic regression on the same data managed only 52 %, confirming that the non-linear decision boundary requires non-linear modelling capacity." Always state architecture, regularisation, and a linear baseline.

13.31.8 Practical Tips

- Always standardise inputs; unscaled features with different units make training unstable and slow.
- Combine multiple regularisers: weight decay (L2), dropout, early stopping based on validation loss.
- Initialise weights carefully: Xavier (for tanh) or He (for ReLU) for deep networks; `nnet` uses small uniform initialisation that suffices for shallow nets.
- For tabular data, tree-based ensembles (XGBoost, LightGBM) usually outperform MLPs; neural networks dominate where structure matters (images, text, audio, time series).
- Modern R neural-network workflows use `torch` or `keras` rather than `nnet`; the latter is fine for shallow nets and pedagogy but lacks the GPU support

and modern optimisers needed for production.
- For very small datasets, regularise heavily (high decay, small networks, dropout) or fall back to linear / tree models; deep nets need data to generalise.

13.31.9 R Packages Used

`nnet::nnet()` for shallow MLPs; `torch` and `luz` for modern deep learning with GPU support; `keras` and `tensorflow` as the alternative TensorFlow-based stack; `parsnip::mlp()` for tidymodels integration; `neuralnet` for an alternative implementation with visual graph rendering.

13.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.31.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.32 Partial Dependence Plots

13.32.1 Introduction

Partial dependence plots (PDPs), introduced by Friedman in 2001, show the average model prediction as a feature varies while holding other features at their observed values. They provide a visual answer to "how does the model's prediction change with x_j?" — exposing non-linear effects and interactions in black-box models that variable-importance scores alone cannot reveal. Combined with variable importance (which features matter) and SHAP values (per-prediction explanations), PDPs are the third pillar of interpretation for opaque ML models.

13.32.2 Prerequisites

A working understanding of trained black-box classifiers or regressors (random forest, gradient boosting, neural networks), variable importance, and the difference between marginal and conditional effects.

13.32.3 Theory

Partial dependence at value $x_j = v$ is

$$\mathrm{PD}_j(v) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(v, x_{-j,i}),$$

i.e., set the feature of interest to v across all rows in the dataset, predict, and average. The PDP is the curve of $\mathrm{PD}_j(v)$ as v varies over the observed range of x_j.

Individual Conditional Expectation (ICE) plots show one curve per observation rather than the average; the PDP is the mean of the ICE curves. ICE plots expose heterogeneity in feature effects: if all ICE curves are roughly parallel, the effect of x_j is homogeneous; if they diverge, an interaction with another feature is implied.

Accumulated Local Effects (ALE) plots address PDP's extrapolation problem when features are correlated; they integrate the prediction's local derivative rather than averaging predictions at potentially out-of-distribution feature combinations.

13.32.4 Assumptions

For PDP interpretation, features are approximately independent; otherwise the average over x_{-j} at fixed $x_j = v$ may include low-density regions of the joint distribution and produce misleading curves. For correlated features, use ALE plots instead.

13.32.5 R Implementation

```
library(pdp); library(ranger)

set.seed(2026)
n <- 500
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n),
  y  = sin(rnorm(n)) + 0.3 * rnorm(n)^2
)
```

```
rf <- ranger(y ~ ., data = df, num.trees = 300)
pd <- partial(rf, pred.var = "x1", plot = FALSE, train = df)

plot(pd$x1, pd$yhat, type = "l", lwd = 2, col = "#2A9D8F",
     xlab = "x1", ylab = "PD(x1)",
     main = "Partial dependence of x1")
```

13.32.6 Output & Results

`pdp::partial()` returns a data frame with the feature value and the partial-dependence prediction. Plotting it as a curve shows how the average prediction varies with the feature; non-linear shapes reveal effects that linear models would miss.

13.32.7 Interpretation

A reporting sentence: "The partial-dependence plot revealed a non-monotone sinusoidal relationship between x_1 and the outcome (peak prediction near $x_1 = -0.5$, trough near $x_1 = 1.5$); this structure was not visible from variable importance alone, which simply ranked x_1 as the strongest predictor. ICE curves were broadly parallel, indicating no major interaction with other features." Always pair PDP with ICE for heterogeneity detection.

13.32.8 Practical Tips

- Pair PDP with variable importance: importance answers "which features matter?", PDP answers "how do they affect the prediction?".
- Use ICE plots to detect interactions: ICE curves diverging from the PDP indicate heterogeneous effects driven by another feature.
- PDPs extrapolate into low-density regions when features are correlated; for correlated-feature contexts, use Accumulated Local Effects (ALE) plots via `iml::FeatureEffect` or `ALEPlot`.
- Two-dimensional PDPs (pair of features) reveal interactions but are computationally expensive; use a sampled feature grid for large datasets.
- `DALEX::model_profile()` offers a unified interpretation interface across tree, linear, and deep models with consistent API.
- For per-prediction explanations rather than global ones, use SHAP values (`shapviz`, `iml`); SHAP and PDP answer related but distinct interpretation questions.

13.32.9 R Packages Used

`pdp` for the canonical PDP implementation; `iml::FeatureEffect()` for PDP, ICE, and ALE within a unified interpretation framework; `DALEX::model_profile()` for

model-agnostic PDPs; `ALEPlot` for ALE specifically; `shapviz` and `fastshap` for SHAP-based per-prediction explanations.

13.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.32.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.33 Platt Scaling

13.33.1 Introduction

Platt scaling (Platt, 1999) is a post-hoc calibration method that maps a classifier's raw scores onto well-calibrated probabilities by fitting a logistic regression from scores to true labels on a held-out validation set. It is the simplest and often best calibration method for SVMs, neural networks, and other classifiers whose mis-calibration is approximately monotone. The idea is to leave the classifier's ranking untouched while rescaling the score axis so that the predicted probability of class 1 matches the empirical class-1 rate.

13.33.2 Prerequisites

A working understanding of calibration plots, the difference between discrimination and calibration, logistic regression, and the use of held-out validation sets.

13.33.3 Theory

Given validation pairs (f_i, y_i) where f_i is the classifier score (decision value, logit, or raw probability), fit

$$P(y = 1 \mid f) = \frac{1}{1 + \exp(Af + B)},$$

by maximum likelihood. New predictions are $P(y = 1 \mid f(x))$ from this fitted sigmoid. The transformation has only two parameters and preserves the rank order — and therefore the ROC curve and AUC — of the original classifier. Label smoothing $(y+1)/(N+2)$ avoids numerical issues near 0 and 1 when the validation set is small.

13.33.4 Assumptions

Mis-calibration is monotone (a sigmoid family is appropriate); a separate validation set is available for fitting the calibration; the classifier's scores rank observations meaningfully (Platt scaling cannot fix a classifier with poor discrimination).

13.33.5 R Implementation

```
set.seed(2026)
n <- 1000
# Well-ranked but mis-calibrated scores
y <- rbinom(n, 1, 0.3)
f <- 2 * y + rnorm(n, 0, 1.2)         # decision function

# Naive sigmoid of f does not match observed frequencies
naive_p <- plogis(f)

# Platt scaling: fit logistic regression y ~ f on validation
val_idx <- 1:500
te_idx  <- 501:1000
fit <- glm(y[val_idx] ~ f[val_idx], family = "binomial")
A <- coef(fit)[2]; B <- coef(fit)[1]
platt_p <- plogis(A * f[te_idx] + B)

# Brier scores
c(naive = mean((naive_p[te_idx] - y[te_idx])^2),
  platt = mean((platt_p - y[te_idx])^2))
```

13.33.6 Output & Results

Brier score (mean squared error of probability vs label) drops after Platt scaling, reflecting improved calibration. AUC remains unchanged because the rescaling is monotone and preserves rank.

13.33.7 Interpretation

A reporting sentence: "Platt scaling reduced the Brier score from 0.21 (naive sigmoid of SVM decision values) to 0.18 (Platt-rescaled) on the held-out test set; ROC AUC was unchanged at 0.81 because the rescaling preserves rank order. The reliability diagram shifted toward the diagonal at every probability decile after recalibration." Always report Brier score and a reliability diagram before and after calibration.

13.33.8 Practical Tips

- Platt scaling works best when mis-calibration is smooth and monotone; for wiggly or non-monotone calibration curves, isotonic regression is preferred.
- Use a separate validation set (not training data) to fit the calibration; in-sample calibration is self-reinforcing and biased.
- Platt scaling adds only 2 parameters and is extremely data-efficient — a few hundred validation points are usually sufficient.
- For multi-class problems, fit one-vs-rest Platt scaling per class and renormalise the resulting probabilities to sum to 1.
- Pair with a reliability diagram (`ggcalibrate` or hand-built) before and after calibration to verify the visual improvement.
- Platt scaling cannot improve discrimination; if AUC is poor, fix the underlying classifier rather than chasing better calibration.

13.33.9 R Packages Used

Base R `glm(family="binomial")` for fitting the Platt sigmoid; `caret::calibrate()` for an integrated calibration workflow; `probably` for tidymodels-flavoured calibration utilities; `betacal` for Beta calibration as an alternative parametric form; `mlr3calibration` for calibration extensions in the mlr3 ecosystem.

13.33.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.33.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch

- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.34 Random Forests

13.34.1 Introduction

Random forests, introduced by Breiman in 2001, average predictions from many bootstrapped decision trees, each trained with a random subset of features considered at every split. Two sources of randomness — bootstrap row sampling and feature subsampling — decorrelate the trees and dramatically reduce variance compared with a single tree. The result is a strong off-the-shelf baseline for tabular data: minimal hyperparameter tuning, robust to outliers and irrelevant features, and providing free out-of-bag (OOB) error estimation as a side effect of the bootstrap.

13.34.2 Prerequisites

A working understanding of decision trees, bagging (bootstrap aggregating), and the bias-variance trade-off in ensemble methods.

13.34.3 Theory

For each of B trees:

1. Draw a bootstrap sample (sampling with replacement, same size as original) from the training data.
2. Grow a deep tree on this sample; at each split, consider a random subset of m features (typically $m = \sqrt{p}$ for classification, $m = p/3$ for regression).
3. Predict on each tree; aggregate predictions by averaging (regression) or majority voting (classification).

Out-of-bag (OOB) error uses observations not selected by each bootstrap as a held-out validation set; aggregating these per-tree predictions across the forest gives a nearly unbiased generalisation estimate at zero additional computational cost.

13.34.4 Assumptions

Observations are independent and identically distributed; features are informative (random forests are robust to irrelevant features but cannot synthesise signal that is not there); the default m value is a starting point but may need tuning for specific datasets.

13.34.5 R Implementation

```
library(ranger)

set.seed(2026)
n <- 500
df <- data.frame(x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n),
                 y  = factor(sample(c("pos", "neg"), n, replace = TRUE)))
df$y <- factor(ifelse(plogis(0.8 * df$x1 - 0.6 * df$x2) > runif(n),
 ↪  "pos", "neg"))

rf <- ranger(y ~ ., data = df,
             num.trees = 500,
             mtry = 1,
             importance = "impurity",
             classification = TRUE)

rf$prediction.error         # OOB misclassification error
rf$variable.importance
```

13.34.6 Output & Results

`ranger()` returns a fitted random forest with OOB error (`prediction.error`), per-feature variable importance (`variable.importance`), and the underlying tree structures. The OOB error is the standard generalisation estimate; variable importance ranks features by their average contribution to prediction quality.

13.34.7 Interpretation

A reporting sentence: "A 500-tree random forest with `mtry=1` achieved OOB classification error of 0.17; variable importance identified x_1 (impurity decrease 32) and x_2 (decrease 18) as the top predictors, consistent with the data-generating logistic process. A held-out test set confirmed the OOB-based estimate (test error 0.16)." Always report OOB error or held-out test error and the top variable importances.

13.34.8 Practical Tips

- Use `ranger` instead of the original `randomForest` package for new analyses; it is dramatically faster and lower-memory while producing essentially identical results.
- Tune `mtry` on 5-fold cross-validation; the default ($\sqrt{p}$ for classification, $p/3$ for regression) is reasonable but not always optimal.
- `min.node.size` controls leaf size; larger values give more bias and less variance — useful for very noisy datasets.

- For unbiased variable importance, use `importance="permutation"` instead of `"impurity"`; it is slower but less biased toward continuous features and high-cardinality categoricals.
- OOB error is nearly unbiased for moderate to large n; use it instead of cross-validation for rapid hyperparameter screening, then validate the final choice on a held-out test set.
- For class imbalance, use `class.weights` or `case.weights` (in `ranger`); both are natively supported and more efficient than over- or under-sampling.

13.34.9 R Packages Used

`ranger` for fast random forests with OOB and permutation importance; `randomForest` for the original Breiman implementation; `parsnip::rand_forest()` for the tidymodels-flavoured wrapper; `caret` for cross-validated tuning; `vip` for variable-importance visualisation; `pdp` for partial-dependence plots.

13.34.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.34.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.35 Preprocessing with recipes

13.35.1 Introduction

The `recipes` package in tidymodels defines preprocessing pipelines as a sequence of named steps — impute missing values, normalise numeric predictors, encode categoricals, apply PCA — that learn their parameters from training data and apply them consistently at prediction time. The principal benefit is leak-free preprocessing: imputation means, scaling parameters, and PCA loadings are estimated only on the training set, never on test or validation folds. Applying pre-

processing globally before splitting (a frequent mistake) leaks information from test into training and produces optimistically biased performance estimates.

13.35.2 Prerequisites

A working understanding of tidymodels, train-test splits, cross-validation, and the concept of data leakage.

13.35.3 Theory

A recipe has two phases:

1. **Define**: chain `recipe()` with `step_*()` functions in pipeline order.
2. **Prep / bake**: `prep()` learns parameters from training data; `bake()` applies them to any new data.

All statistics (centring means, scaling SDs, imputation values, PCA loadings) come from the data passed to `prep()`. When the recipe is then applied to test data via `bake()` or via a `workflow()`, the same parameters are used — preserving the principle that test data must not influence preprocessing.

Steps are executed in declared order; ordering matters substantively (impute before scale, not after; encode categoricals before any operation that requires numeric inputs).

13.35.4 Assumptions

Training data is representative enough to estimate transform parameters reliably; step ordering is correct for the chosen transformations.

13.35.5 R Implementation

```
library(recipes); library(dplyr)

set.seed(2026)
n <- 200
df <- tibble(
  num1 = rnorm(n),
  num2 = rnorm(n),
  cat  = sample(c("a", "b", "c"), n, replace = TRUE),
  y    = rnorm(n)
)
df$num1[sample(n, 10)] <- NA           # introduce missingness

rec <- recipe(y ~ ., data = df) %>%
  step_impute_mean(num1) %>%
  step_normalize(all_numeric_predictors()) %>%
```

```
  step_dummy(all_nominal_predictors())

prepped <- prep(rec, training = df[1:150, ])
test_baked <- bake(prepped, new_data = df[151:200, ])
head(test_baked)
```

13.35.6 Output & Results

`prep()` returns a fitted recipe with all learnt parameters stored. `bake()` applies the prepared recipe to new data, returning a transformed tibble with imputed, normalised, and one-hot-encoded predictors. The transformations on the held-out 50 rows use parameters learned only from the first 150.

13.35.7 Interpretation

A reporting sentence: "The recipe imputed missing values in `num1` with the training-set mean (-0.02), centred and scaled all numeric predictors, and one-hot encoded the `cat` factor with three levels; all parameters were learned on the 150-row training set and applied identically to the 50-row test set, avoiding the data leakage that occurs when preprocessing is applied globally before splitting." Always describe the preprocessing steps in the methods section.

13.35.8 Practical Tips

- Place imputation steps before scaling; you cannot meaningfully standardise a column that contains NAs.
- `step_normalize()` centres and scales; `step_range()` rescales to $[0, 1]$ via min-max.
- For tree-based models (random forest, XGBoost), skip normalisation — trees are scale-invariant — but keep imputation and one-hot encoding.
- Use `step_zv()` and `step_nzv()` to remove zero-variance and near-zero-variance predictors before modelling; they contribute no information and can destabilise some algorithms.
- Pair a recipe with a model specification via `workflow()` so preprocessing and prediction happen as one pipeline; this prevents accidental leakage at predict time.
- For PCA dimensionality reduction, `step_pca()` learns loadings on training data; the same loadings are applied to test data, preserving train-test independence.

13.35.9 R Packages Used

`recipes` for the core preprocessing framework; `workflows` for pairing recipes with models; `embed` for embedding steps (target encoding, UMAP); `themis` for class-imbalance steps (SMOTE, ROSE); `textrecipes` for text-specific preprocessing;

`tidymodels` as the umbrella package that bundles all of these.

13.35.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.35.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.36 Regularisation in ML

13.36.1 Introduction

Regularisation adds a complexity penalty to the training loss, preventing overfitting by shrinking coefficients. Ridge (L2) shrinks all coefficients toward zero; lasso (L1) also performs variable selection; elastic net combines both. All three are standard tools across regression and classification.

13.36.2 Prerequisites

Linear / logistic regression; convex optimisation.

13.36.3 Theory

Objective: $\min_\beta \sum_i L(y_i, x_i^\top \beta) + \lambda \cdot P(\beta)$, with - $P(\beta) = \|\beta\|^2$ (ridge, L2) - $P(\beta) = \|\beta\|_1$ (lasso, L1) - $P(\beta) = \alpha\|\beta\|_1 + (1-\alpha)\|\beta\|^2/2$ (elastic net)

Lasso produces sparse solutions (many $\hat{\beta}_j = 0$); ridge shrinks but rarely zeros; elastic net balances sparsity with the grouping effect for correlated features.

13.36.4 Assumptions

Features are standardised so the penalty is comparable across predictors; λ is tuned by CV.

13.36.5 R Implementation

```r
library(glmnet)

set.seed(2026)
n <- 300; p <- 50
X <- matrix(rnorm(n * p), n, p)
beta <- c(rep(2, 5), rep(0, 45))
y <- X %*% beta + rnorm(n)

cv_ridge <- cv.glmnet(X, y, alpha = 0,   nfolds = 10)   # L2
cv_lasso <- cv.glmnet(X, y, alpha = 1,   nfolds = 10)   # L1
cv_enet  <- cv.glmnet(X, y, alpha = 0.5, nfolds = 10)

nz <- function(fit) sum(as.vector(coef(fit, s = "lambda.min")) != 0) - 1
cat("nonzero: ridge=", nz(cv_ridge),
    "  lasso=", nz(cv_lasso),
    "  elastic=", nz(cv_enet), "\n")
c(ridge = cv_ridge$cvm[cv_ridge$lambda == cv_ridge$lambda.min],
  lasso = cv_lasso$cvm[cv_lasso$lambda == cv_lasso$lambda.min],
  enet  = cv_enet $cvm[cv_enet $lambda == cv_enet $lambda.min])
```

13.36.6 Output & Results

Non-zero counts: ridge retains all 50, lasso selects ~5, elastic net selects ~7. Elastic net combines sparsity (lasso-like) with stability (ridge-like) for correlated features.

13.36.7 Interpretation

"Lasso with CV-selected λ retained the 5 true non-zero features and zero-ed out 45 irrelevant ones; elastic net gave similar sparsity but handled correlated predictors more gracefully."

13.36.8 Practical Tips

- Always standardise features (`glmnet` does this internally); otherwise the penalty is scale-dependent.
- Use `lambda.1se` for a sparser, more parsimonious model within one SE of the CV minimum.
- Lasso picks one from correlated-feature groups arbitrarily; elastic net picks all – often more stable.
- For classification, extend via `family="binomial"`; same rules apply.
- Gradient-boosting frameworks expose L1/L2 regularisation on leaf weights; not identical to glmnet's lasso/ridge but same motivation.

13.36.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.36.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.37 RNNs and LSTMs: Introduction

13.37.1 Introduction

Recurrent Neural Networks (RNNs) process sequences by maintaining a hidden state that updates at every time step. Long Short-Term Memory units (LSTM; Hochreiter & Schmidhuber, 1997) add gating mechanisms to combat vanishing gradients, enabling long-range dependencies. GRUs (Cho, 2014) are a simpler gated variant.

13.37.2 Prerequisites

Feedforward networks; gradient-based training; sequence tasks.

13.37.3 Theory

Vanilla RNN: $h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$. Gradients through time vanish or explode, limiting long-range dependence.

LSTM adds input, forget, and output gates, plus a memory cell:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \quad h_t = o_t \odot \tanh(c_t).$$

Gates control what is written, read, and forgotten in the memory cell.

Bidirectional RNNs concatenate forward and backward hidden states – useful for sequence classification.

13.37.4 Assumptions

Inputs are sequential and order-dependent; padding/masking handles variable-length sequences correctly; sufficient context in the hidden state.

13.37.5 R Implementation

```
library(torch)

torch_manual_seed(2026)

# Batch of 4 sequences, each length 10 and feature dim 5
x <- torch_randn(4, 10, 5)

lstm <- nn_lstm(input_size = 5, hidden_size = 8, num_layers = 2,
                batch_first = TRUE)
out  <- lstm(x)             # list: output tensor, (h_n, c_n)

out[[1]]$shape              # [4, 10, 8]
out[[2]][[1]]$shape         # h_n: [2, 4, 8]

# Simple forecasting model: LSTM + linear head
forecaster <- nn_module(
  initialize = function() {
    self$lstm <- nn_lstm(1, 16, batch_first = TRUE)
    self$fc   <- nn_linear(16, 1)
  },
  forward = function(x) {
    out <- self$lstm(x)[[1]]
    self$fc(out[, -1, ])       # predict from last time step
  }
)

model <- forecaster()
y_seq <- torch_randn(4, 10, 1)
model(y_seq)$shape          # [4, 1]
```

13.37.6 Output & Results

The LSTM returns per-timestep hidden states (shape [batch, seq, hidden]) plus final hidden and cell states.

13.37.7 Interpretation

“A 2-layer LSTM with 8 hidden units processed 10-step sequences; the final hidden state feeds a linear classifier. Transformers now exceed LSTMs on most

NLP tasks, but LSTMs remain strong for short-window forecasting and low-data regimes."

13.37.8 Practical Tips

- Use LSTMs or GRUs over vanilla RNNs; the latter are hard to train for any real-world sequence length.
- Pack sequences (`nn_utils_rnn_pack_padded_sequence`) to skip padding computations.
- Gradient clipping (`nn_utils_clip_grad_norm_`) is almost always needed for RNN training stability.
- For short tasks, a 1-D CNN over the sequence often matches or beats an LSTM with faster training.
- For long sequences with global dependencies, transformers beat LSTMs; but they need more data.

13.37.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.37.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.38 SHAP Values

13.38.1 Introduction

SHAP (SHapley Additive exPlanations; Lundberg & Lee, 2017) attributes each prediction to input features via the Shapley value from cooperative game theory. It is the only attribution method satisfying the combined axioms of local accuracy, consistency, and missingness. SHAP has become the standard for black-box model interpretability.

13.38.2 Prerequisites

Variable importance; cooperative game theory basics helpful.

13.38.3 Theory

For prediction $f(x)$, each feature j gets a SHAP value ϕ_j satisfying

$$f(x) = \mathbb{E}[f(X)] + \sum_j \phi_j.$$

Shapley values average the marginal contribution of feature j across all possible subsets of features, producing a fair attribution. Exact computation is exponential; Tree SHAP (Lundberg, 2020) computes it in polynomial time for tree ensembles.

13.38.4 Assumptions

Features have sensible "absence" interpretations (TreeSHAP integrates over feature distributions); additive decomposition meaningfully captures model behaviour.

13.38.5 R Implementation

```
library(xgboost); library(shapviz)

set.seed(2026)
n <- 500
X <- matrix(rnorm(n * 4), n, 4); colnames(X) <- paste0("x", 1:4)
y <- as.numeric(X[, 1] > 0) + 0.5 * as.numeric(X[, 2] > 0) +
     rnorm(n, 0, 0.5)

dtrain <- xgb.DMatrix(X, label = y)
model <- xgb.train(list(objective = "reg:squarederror",
                        max_depth = 3, eta = 0.1),
                   data = dtrain, nrounds = 200, verbose = 0)

sv <- shapviz(model, X_pred = X, X = X)

# Summary (importance ranking)
sv_importance(sv)

# Per-observation waterfall (first prediction)
sv_waterfall(sv, row_id = 1)
```

13.38.6 Output & Results

An importance plot (mean |SHAP| per feature) and a waterfall plot showing how each feature pushes the specific prediction above or below the baseline.

13.38.7 Interpretation

"SHAP ranked x_1 as the dominant driver (mean |SHAP| = 0.36); the waterfall showed observation 1's prediction was pushed up by 0.42 from baseline primarily due to $x_1 = 1.8$."

13.38.8 Practical Tips

- Use **Tree SHAP** (fast, exact) for tree models; fallback to **Kernel SHAP** for arbitrary models (approximate, slower).
- SHAP dependence plots (SHAP value vs feature value) reveal non-linear effects and interactions.
- SHAP values sum to $f(x) - \mathbb{E}[f(X)]$, so per-observation explanations are additive.
- For deep learning, use `DeepSHAP` or integrated gradients; TreeSHAP does not apply.
- Global importance = mean |SHAP| across observations; usually close to permutation importance on tree ensembles.

13.38.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.38.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.39 SMOTE for Imbalanced Classes

13.39.1 Introduction

On imbalanced classification problems, classifiers trained with standard loss optimise for the majority class and under-predict the minority. SMOTE (Chawla et al., 2002) addresses this by synthesising new minority examples via interpolation between minority neighbours, balancing the training set without duplicating samples.

13.39.2 Prerequisites

Classification; understanding of imbalanced data; kNN.

13.39.3 Theory

SMOTE algorithm: 1. For each minority sample x_i, find its k nearest minority neighbours. 2. Select a random neighbour x_{nn}. 3. Generate a synthetic sample $x_{\text{new}} = x_i + \lambda(x_{nn} - x_i)$, $\lambda \sim U(0,1)$.

Apply within the training set only (never the test set). Variants: Borderline-SMOTE, ADASYN (focus on hard minority regions).

13.39.4 Assumptions

Feature space is continuous (SMOTE works poorly for categoricals without adaptation); minority class has enough samples to define meaningful neighbourhoods.

13.39.5 R Implementation

```
library(themis); library(recipes); library(dplyr)

set.seed(2026)
n <- 500
df <- tibble(
  x1 = rnorm(n), x2 = rnorm(n),
  y  = factor(sample(c("pos", "neg"), n, replace = TRUE,
                     prob = c(0.05, 0.95)))
)
table(df$y)

rec <- recipe(y ~ ., data = df) %>%
  step_smote(y, over_ratio = 1, neighbors = 5)

baked <- prep(rec) %>% bake(new_data = NULL)
table(baked$y)
```

13.39.6 Output & Results

Original class counts (~475 neg, 25 pos); after SMOTE the minority is upsampled via synthetic examples to match the majority.

13.39.7 Interpretation

"SMOTE upsampled the 5 % minority class to 50 % synthetic examples, enabling the downstream logistic regression to learn a better boundary. Test AUC improved from 0.74 to 0.83 on the held-out set."

13.39.8 Practical Tips

- Apply SMOTE **only** to the training set (inside the CV fold), never to test/validation.
- `over_ratio=1` creates a fully balanced training set; smaller ratios undersample less.
- SMOTE works on numeric features; for mixed data, use SMOTE-NC or other mixed-type variants.
- Alternatives: class weighting in the loss function is often as effective and simpler.
- Evaluate using class-balanced metrics (balanced accuracy, F1, AUC, PR-AUC) – plain accuracy misleads under imbalance.

13.39.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.39.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.40 Stacking Ensembles

13.40.1 Introduction

Stacking, also called stacked generalisation (Wolpert 1992), combines predictions from several diverse base learners using a second-level meta-learner trained to predict the target from the base-model predictions. Stacking consistently wins competitive ML benchmarks when base models are individually decent and sufficiently diverse — the meta-learner captures complementary strengths that no single model can match. It generalises bagging (one type of model averaged) and boosting (one type of model sequentially trained) to arbitrary heterogeneous ensembles.

13.40.2 Prerequisites

A working understanding of cross-validation, the bias-variance trade-off in ensembling, and the difference between in-sample and out-of-fold predictions.

13.40.3 Theory

The standard stacking workflow:

1. **Train base learners** $M_1, M_2, \dots, M_K$ on the training data; compute *out-of-fold* (OOF) predictions $\hat{y}_k^{\text{oof}}$ via cross-validation.
2. **Train a meta-learner** g (typically linear or regularised regression / logistic regression) to predict the target from the OOF predictions: $\hat{y} = g(\hat{y}_1^{\text{oof}}, \dots, \hat{y}_K^{\text{oof}})$.
3. **Refit base learners** on the full training set; produce final predictions as $\hat{y}(x) = g(M_1(x), \dots, M_K(x))$.

Using OOF predictions for the meta-learner prevents leakage that would occur if the meta-learner trained on in-sample base predictions.

13.40.4 Assumptions

Base learners are sufficiently diverse (a trees + linear + kNN combination usually outperforms five different tree variants); the meta-learner is simple enough that it does not over-fit the small set of base predictions.

13.40.5 R Implementation

```
library(tidymodels); library(stacks)

set.seed(2026)
n <- 500
df <- tibble(x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n),
             y  = factor(sample(c("pos", "neg"), n,
```

```
                              replace = TRUE, prob = c(0.3, 0.7))))

split <- initial_split(df, strata = y)
train <- training(split); test <- testing(split)

folds <- vfold_cv(train, v = 5)
ctrl <- control_stack_grid()

rec <- recipe(y ~ ., data = train)
rf_spec  <- rand_forest(mtry = 2, trees = 300) %>%
  set_engine("ranger") %>% set_mode("classification")
log_spec <- logistic_reg(penalty = 0.01) %>%
  set_engine("glmnet") %>% set_mode("classification")

rf_wf  <- workflow() %>% add_recipe(rec) %>% add_model(rf_spec)
log_wf <- workflow() %>% add_recipe(rec) %>% add_model(log_spec)

rf_res  <- fit_resamples(rf_wf,  folds, control = ctrl)
log_res <- fit_resamples(log_wf, folds, control = ctrl)

st <- stacks() %>% add_candidates(rf_res) %>% add_candidates(log_res)
st_blend <- blend_predictions(st)
st_fit   <- fit_members(st_blend)

st_fit
```

13.40.6 Output & Results

`fit_members()` returns the trained stack with meta-learner coefficients weighting each base model. Inspecting `st_blend` shows the chosen blending weights — typically a non-negative least-squares or penalised regression solution that emphasises the base models with the most distinctive predictive signal.

13.40.7 Interpretation

A reporting sentence: "Stacking a random forest and a regularised logistic regression via a penalised linear meta-learner achieved test ROC AUC of 0.89, exceeding either base model alone (RF 0.85, logistic 0.80); the meta-learner weighted RF at 0.62 and logistic at 0.38, with both contributions retained at non-zero values." Always report base-model performances, the meta-learner choice, and the blending weights.

13.40.8 Practical Tips

- Base-model diversity (tree + linear + kNN + neural net) matters more than having many similar learners; a stack of five XGBoost variants is

rarely better than a single well-tuned XGBoost.
- Use a regularised linear meta-learner (lasso or non-negative LS); unregularised meta-learners over-fit the relatively small set of OOF predictions.
- Pass *out-of-fold* predictions to the meta-learner, not in-sample predictions; this is the central correctness guarantee of stacking.
- Stacking plus bagging hybrids (super learner of van der Laan, Polley, Hubbard) perform best on moderate-to-large data.
- The `stacks` package (tidymodels) automates the workflow and integrates with the broader `tune` ecosystem for cross-validated base-model tuning.
- `SuperLearner` is the classical alternative outside tidymodels; both produce equivalent results given the same base models.

13.40.9 R Packages Used

`stacks` for tidymodels-based stacking; `SuperLearner` for the classical implementation; `tune::fit_resamples()` for the cross-validated base-learner predictions; `parsnip` for unified base-model specifications; `glmnet` and `ranger` for the standard base-learner choices.

13.40.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.40.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.41 Supervised Learning: Overview

13.41.1 Introduction

Supervised learning uses labelled data $(x_i, y_i)_{i=1}^n$ to learn a function f that predicts y from new x. The enterprise is defined by three choices: a **hypothesis**

class (tree, linear model, neural net), a **loss function** (squared error, cross-entropy), and an **optimisation** procedure. Generalisation – performance on unseen data – is the goal, and the bias-variance decomposition is the classical lens for understanding why models fail.

13.41.2 Prerequisites

Probability, linear algebra, basic statistics.

13.41.3 Theory

Empirical risk minimisation (ERM): $\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_i L(y_i, f(x_i))$.

Loss functions: - Regression: squared loss $(y - f)^2$, absolute loss $|y - f|$. - Classification: 0/1 (intractable), log loss (cross-entropy), hinge (SVMs).

Bias-variance decomposition for squared loss:

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2 + \text{Variance} + \sigma^2_{\text{noise}}.$$

Flexible models have low bias, high variance; rigid models the reverse. Regularisation trades one for the other.

13.41.4 Assumptions

(x, y) pairs are drawn iid from some joint distribution; the deployment distribution matches the training distribution (no distribution shift).

13.41.5 R Implementation

```r
library(tidymodels)

set.seed(2026)
n <- 300
x <- runif(n, 0, 2 * pi)
y <- sin(x) + rnorm(n, 0, 0.3)
df <- tibble(x = x, y = y)

split <- initial_split(df, prop = 0.7)
train <- training(split); test <- testing(split)

# Two models of different flexibility
m_rigid <- lm(y ~ x, data = train)
m_flex  <- loess(y ~ x, data = train, span = 0.2)

rmse <- function(yhat, y) sqrt(mean((yhat - y)^2))
rmse(predict(m_rigid, test), test$y)
```

```
rmse(predict(m_flex,  test), test$y)
```

13.41.6 Output & Results

The flexible loess model has lower test RMSE on this non-linear signal; the rigid linear model under-fits.

13.41.7 Interpretation

"The sinusoidal signal is captured by loess (test RMSE 0.31) but missed by linear regression (test RMSE 0.68), illustrating how bias dominates when the hypothesis class is too restrictive."

13.41.8 Practical Tips

- Always report test performance, never training performance, as a generalisation estimate.
- The no-free-lunch theorem: no single model wins everywhere; match model to the data-generating process.
- Stratify classification splits to preserve class proportions.
- Use cross-validation for small-to-moderate data; a single train/test split is high-variance.
- Track multiple metrics: RMSE and MAE together are more informative than either alone.

13.41.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.41.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.42 Kernel SVMs

13.42.1 Introduction

Kernel SVMs extend the linear SVM framework to non-linear decision boundaries via the kernel trick: replace every inner product $x_i^\top x_j$ in the dual formulation with a kernel function $K(x_i, x_j)$ that corresponds to an inner product in a high-dimensional (often infinite-dimensional) implicit feature space. The Radial Basis Function (RBF) kernel is the workhorse, capable of representing essentially any smooth decision boundary given enough training data and appropriate hyperparameter tuning.

13.42.2 Prerequisites

A working understanding of linear SVMs, inner-product spaces, the dual formulation of constrained optimisation, and Mercer's theorem (the conditions a function must satisfy to be a valid kernel).

13.42.3 Theory

Common kernels:

- **RBF (Gaussian)**: $K(x, x') = \exp(-\gamma \|x - x'\|^2)$. Universal approximator; the default choice for most non-linear problems.
- **Polynomial**: $K(x, x') = (x^\top x' + c)^d$. Useful when polynomial structure is expected from domain knowledge.
- **Linear**: $K(x, x') = x^\top x'$. Reduces to the standard linear SVM.
- **Sigmoid**: $K(x, x') = \tanh(\alpha x^\top x' + c)$, related to single-layer perceptrons.

Two hyperparameters dominate: the cost parameter C (margin tightness, inherited from linear SVM) and a kernel-specific parameter (γ for RBF, d and c for polynomial). RBF γ trades bias for variance: small γ produces a smoother decision boundary, large γ produces a wigglier one with high variance.

13.42.4 Assumptions

Features are appropriately scaled; sufficient data to support the kernel's implicit dimensionality (RBF in particular grows in effective complexity with γ); C and γ are tuned jointly because they are correlated.

13.42.5 R Implementation

```
library(e1071)

set.seed(2026)
n <- 300
theta <- runif(n, 0, 2 * pi)
```

```
r <- c(rep(1, n/2), rep(2.5, n/2)) + rnorm(n, 0, 0.3)
df <- data.frame(x1 = r * cos(theta), x2 = r * sin(theta),
                 y  = factor(rep(c("inner", "outer"), each = n/2)))
df[, 1:2] <- scale(df[, 1:2])

svm_rbf <- svm(y ~ ., data = df,
               kernel = "radial", cost = 1, gamma = 0.5,
               scale = FALSE)
table(predict(svm_rbf, df), df$y)

# Grid-search C x gamma via tune()
tune_out <- tune(svm, y ~ ., data = df, kernel = "radial",
                 ranges = list(cost = c(0.1, 1, 10),
                               gamma = c(0.1, 0.5, 1)))
tune_out$best.parameters
```

13.42.6 Output & Results

The RBF-kernel SVM separates the two concentric classes — a non-linear pattern that no linear classifier can solve. `tune()` performs grid-search cross-validation over C and γ and returns the best combination by CV accuracy.

13.42.7 Interpretation

A reporting sentence: "The RBF-kernel SVM correctly separated the concentric-rings classes with grid-CV-selected $C = 1$ and $\gamma = 0.5$ (CV accuracy 0.96, test accuracy 0.94); the linear-kernel baseline managed only 0.51, confirming the non-linear structure cannot be captured without the kernel trick." Always state both hyperparameters and how they were tuned.

13.42.8 Practical Tips

- Always standardise features; kernel distances are scale-sensitive and unscaled features bias the geometry.
- For RBF, γ trades bias (small γ, smoother boundary) for variance (large γ, wigglier boundary); tune jointly with C on a logarithmic 2D grid.
- C and γ are correlated; larger C paired with smaller γ often gives similar performance to smaller C with larger γ.
- High-degree polynomial kernels are rarely useful in practice; RBF subsumes most cases and is more numerically stable.
- For large n ($> 100{,}000$), kernel SVMs become computationally expensive ($O(n^2)$ to $O(n^3)$ training); consider Nyström approximation (`kernlab::kha`) or fall back to primal linear SVM on engineered features.
- For probability output, set `probability=TRUE` in `e1071::svm()`; it fits a Platt-scaling layer on top of the SVM decision values.

13.42.9 R Packages Used

`e1071::svm()` for the canonical kernel SVM with RBF, polynomial, and sigmoid options; `kernlab::ksvm()` for an alternative with richer kernel customisation; `LiblineaR` for the primal linear-only fast variant; `parsnip::svm_rbf()` and `svm_poly()` for tidymodels integration; `caret` for cross-validated joint tuning.

13.42.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.42.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.43 Linear Support Vector Machines

13.43.1 Introduction

Linear support vector machines (SVMs) find the hyperplane that maximally separates classes with the largest possible margin. The soft-margin formulation introduced by Cortes and Vapnik in 1995 added slack variables to tolerate misclassified or margin-violating points, governed by a regularisation parameter C. Linear SVMs scale to very large sparse feature spaces (common in text classification, where bag-of-words gives millions of features) and are competitive with logistic regression on most tabular classification tasks.

13.43.2 Prerequisites

A working understanding of linear classification, convex optimisation, and the geometric concept of margin in supervised learning.

13.43.3 Theory

The soft-margin primal optimisation problem is

$$\min_{w,b,\xi} \tfrac{1}{2}\|w\|^2 + C\sum_i \xi_i \quad \text{s.t.} \quad y_i(w^\top x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0.$$

The first term maximises the margin (minimising $\|w\|$); the second penalises slack. Larger C penalises slack more, favouring low training error at the cost of higher variance; smaller C allows more margin violations and produces a smoother boundary. The dual formulation exposes the support-vector structure: only points on or inside the margin contribute to w, so the trained model is effectively determined by a sparse subset of training points.

13.43.4 Assumptions

Classes are approximately linearly separable for the chosen C; features are appropriately scaled (SVMs are distance-based and unscaled features dominate the geometry).

13.43.5 R Implementation

```
library(e1071)

set.seed(2026)
n <- 200
x1 <- rnorm(n); x2 <- rnorm(n)
y <- factor(ifelse(x1 + x2 + rnorm(n, 0, 0.5) > 0, "pos", "neg"))
df <- data.frame(x1, x2, y)

# Scale the features
df[, 1:2] <- scale(df[, 1:2])

svm_fit <- svm(y ~ ., data = df,
               kernel = "linear",
               cost = 1,
               scale = FALSE)

table(pred = predict(svm_fit, df), truth = df$y)

# Vary C and observe support-vector count
cs <- c(0.01, 0.1, 1, 10, 100)
sapply(cs, function(c) {
  f <- svm(y ~ ., data = df, kernel = "linear", cost = c, scale = FALSE)
  c(C = c, n_sv = f$tot.nSV)
})
```

13.43.6 Output & Results

`svm()` returns a fitted classifier with the support vectors, dual coefficients, and a learned weight vector. Varying C shows the trade-off: small C retains many support vectors (soft boundary), large C retains few (tight boundary, high variance).

13.43.7 Interpretation

A reporting sentence: "A linear SVM with cross-validated $C = 1$ achieved 92 % test accuracy with 124 support vectors out of 200 training points; the support-vector count indicates the classes overlap moderately and a soft boundary is appropriate. Higher C values reduced support-vector counts but increased test error, confirming the bias-variance trade-off." Always report C, support-vector counts, and how C was tuned.

13.43.8 Practical Tips

- Standardise features before fitting; unscaled features with different units bias the geometry of the margin.
- Tune C by cross-validation over a logarithmic grid (typical range 10^{-3} to 10^{3}); the U-shaped CV-error curve has a clear optimum.
- For class imbalance, use `class.weights` to up-weight the minority class; alternatively, balance via under-sampling.
- For very large sparse feature spaces (text classification with millions of features), `LiblineaR` is dramatically faster than `e1071::svm` and equally accurate.
- Linear SVM and logistic regression often produce nearly identical decision boundaries on the same data; the SVM uses hinge loss while logistic uses log-loss, but for most tasks the choice is convention rather than substance.
- For non-linearly separable data, switch to kernel SVM (RBF, polynomial); the linear SVM cannot represent non-linear boundaries regardless of C.

13.43.9 R Packages Used

`e1071::svm()` for the canonical SVM implementation; `LiblineaR` for fast linear SVMs on large sparse data; `kernlab::ksvm()` for an alternative with richer kernel support; `parsnip::svm_linear()` for tidymodels integration; `caret` for cross-validated C tuning.

13.43.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

13.43.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.44 A Complete tidymodels Workflow

13.44.1 Introduction

`tidymodels` is the R meta-package that wraps `rsample`, `recipes`, `parsnip`, `workflows`, `tune`, and `yardstick` into a consistent, tidyverse-friendly interface for machine-learning pipelines. A complete tidymodels workflow composes a data split, a preprocessing recipe, a model specification, hyperparameter tuning on cross-validation resamples, and final evaluation on a held-out test set. The framework's strength is that the same syntax handles regression, classification, and time-series tasks across dozens of underlying engines (lm, glm, random forest, xgboost, neural networks).

13.44.2 Prerequisites

A working knowledge of R basics, the tidyverse, and standard model-fitting concepts (training-test split, cross-validation, hyperparameter tuning).

13.44.3 Theory

The canonical tidymodels pipeline:

1. **Split data** with `rsample::initial_split()` into training and test sets.
2. **Define preprocessing** with `recipes`: imputation, normalisation, dummy encoding, feature engineering.
3. **Specify a model** with `parsnip`, choosing an engine and which hyperparameters to tune.
4. **Combine** into a `workflow` that bundles recipe and model so preprocessing is applied consistently at prediction time.
5. **Tune** hyperparameters on cross-validation resamples (`tune::tune_grid` or Bayesian `tune_bayes`) using a metric set (`yardstick`).
6. **Finalise** on the best hyperparameters and refit on all training data.
7. **Evaluate** on the test set, never touching it before this step.

13.44.4 Assumptions

Data are tidy (one row per observation, one column per variable); roles (outcome vs predictor) are clearly identified; cross-validation splits are appropriate to the problem (random for IID, stratified for imbalance, blocked for time series).

13.44.5 R Implementation

```
library(tidymodels)

set.seed(2026)
df <- tibble(
  x1 = rnorm(500), x2 = rnorm(500),
  x3 = rnorm(500), x4 = rnorm(500),
  y  = factor(sample(c("yes", "no"), 500, replace = TRUE,
                     prob = c(0.3, 0.7)))
)

split <- initial_split(df, strata = y)
train <- training(split); test <- testing(split)
folds <- vfold_cv(train, v = 5, strata = y)

rec <- recipe(y ~ ., data = train) %>%
  step_normalize(all_numeric_predictors())

spec <- rand_forest(trees = 300, mtry = tune(), min_n = tune()) %>%
  set_engine("ranger") %>% set_mode("classification")

wf <- workflow() %>% add_recipe(rec) %>% add_model(spec)

grid <- expand_grid(mtry = c(1, 2, 3), min_n = c(5, 15))
tuned <- tune_grid(wf, folds, grid = grid, metrics = metric_set(roc_auc))

best <- select_best(tuned, metric = "roc_auc")
final_fit <- finalize_workflow(wf, best) %>% last_fit(split)

collect_metrics(final_fit)
```

13.44.6 Output & Results

`tune_grid()` returns a tibble with one row per combination of resample fold and hyperparameter setting, including the metric value (here ROC AUC). `last_fit()` refits the workflow on the full training set with the chosen hyperparameters and evaluates it on the held-out test set in one call.

13.44.7 Interpretation

A reporting sentence: "tidymodels random-forest workflow with 5-fold stratified cross-validation selected `mtry=2`, `min_n=5` (best CV ROC AUC 0.81 across the 6-point grid); on the held-out test set the final model achieved ROC AUC 0.79, the honest generalisation estimate." Always state the resampling scheme, the tuning metric, and the test-set performance.

13.44.8 Practical Tips

- Use `workflow()` rather than direct `fit()` calls; the workflow ensures preprocessing is applied consistently at prediction time without leaking test-set information into transformations.
- `last_fit()` refits on the full training set and evaluates on the test set in a single command; this is the cleanest pattern for the final-model step.
- Stratify splits and CV folds by outcome (`strata=y`) for imbalanced classification; otherwise rare classes can be missing in some folds.
- `hardhat::extract_workflow()`, `extract_fit_engine()`, and `extract_recipe()` retrieve internals from a fitted workflow for inspection or downstream use.
- Parallelise tuning with `doParallel::registerDoParallel(cores=8)` or the `future` framework; tuning is often embarrassingly parallel across resamples and hyperparameter combinations.
- For Bayesian hyperparameter optimisation, use `tune_bayes()` instead of `tune_grid()`; it converges to the optimum in fewer evaluations on smooth performance landscapes.

13.44.9 R Packages Used

`tidymodels` as the umbrella package; `rsample` for splits and resampling; `recipes` for preprocessing; `parsnip` for unified model specifications; `workflows` for bundling; `tune` for hyperparameter optimisation; `yardstick` for performance metrics; `dials` for hyperparameter ranges; `finetune` for racing-based tuning that prunes poor candidates early.

13.44.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.44.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+

- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.45 Train-Test Splits

13.45.1 Introduction

A train-test split reserves a random subset of the data for honest evaluation of model generalisation, separating it cleanly from the data used for fitting and tuning. It is the simplest possible evaluation protocol and the foundation of every supervised-learning workflow. The trade-off is straightforward: a larger test set gives more stable error estimates but leaves less data for fitting; for small datasets, cross-validation extracts more value from the available data than a single fixed split.

13.45.2 Prerequisites

A working understanding of supervised learning, the train-validate-test paradigm, and the difference between hyperparameter tuning and final model evaluation.

13.45.3 Theory

With n observations, reserve n_{test} for test. Typical ratios are 80/20, 70/30, or 90/10 depending on dataset size. The test-set error is a point estimate of generalisation with variance $O(1/n_{\text{test}})$ — for stable error estimates the test set should be at least a few hundred observations.

Stratified splits sample within each class so class proportions match between train and test; this is essential for imbalanced classification, where random sampling can leave the minority class severely under- or over-represented in either fold.

Grouped splits keep all observations from the same subject, site, or sample together to prevent leakage; ignoring grouping when it exists silently inflates performance estimates.

13.45.4 Assumptions

Observation pairs (x_i, y_i) are independent and identically distributed (or the dependence structure is respected by the chosen split scheme); deployment distribution matches the training distribution.

13.45.5 R Implementation

```
library(rsample); library(dplyr)

set.seed(2026)
n <- 1000
df <- tibble(x = rnorm(n),
             y = factor(sample(c("pos", "neg"), n,
                               replace = TRUE, prob = c(0.1, 0.9))))

# Random 80/20 split
split1 <- initial_split(df, prop = 0.8)

# Stratified split preserving y proportions
split2 <- initial_split(df, prop = 0.8, strata = y)

tbl_noStrata <- table(training(split1)$y) / nrow(training(split1))
tbl_strata   <- table(training(split2)$y) / nrow(training(split2))

round(rbind(noStrata = tbl_noStrata, strata = tbl_strata), 3)
```

13.45.6 Output & Results

The proportion table shows that stratified splitting preserves the 10 % minority-class proportion in both train and test; random splitting can drift by 2–3 percentage points, especially with smaller test sets. `training(split)` and `testing(split)` extract the two folds.

13.45.7 Interpretation

A reporting sentence: "An 80/20 stratified split preserved the 10 % minority-class proportion in both training and test folds (training: 10.1 %, test: 9.9 %); we tuned hyperparameters on the training set via 5-fold cross-validation and reserved the test set for a single final evaluation, never touching it during model development." Always state the split ratio, stratification, and how the test set was used.

13.45.8 Practical Tips

- For time series, split chronologically (`rsample::initial_time_split()`); random splitting leaks future information into training and produces optimistic estimates.
- For grouped data (multiple observations per subject), use `rsample::group_initial_split()` to prevent leakage; ignoring grouping when it exists silently inflates performance estimates.

- Test sets should be large enough for stable error estimates; below 200 observations the test error has high variance and a single split may not generalise.
- The test set is touched *once*: after model development is complete. Repeated use during development implicitly tunes against it and produces optimism that no single number can capture.
- For final deployment, refit the best model on the training and test sets combined; the test set served its purpose at evaluation time and there is no reason to discard its information.
- For small datasets, prefer cross-validation over a single train-test split; CV uses the data more efficiently and gives a less variable error estimate.

13.45.9 R Packages Used

`rsample::initial_split()` and `initial_time_split()` for splits; `group_initial_split()` and `group_vfold_cv()` for grouped data; `caret::createDataPartition()` for an older alternative; `mlr3::partition()` for mlr3-style splits; base R `sample()` for ad-hoc splitting.

13.45.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.45.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.46 Transformers: Overview

13.46.1 Introduction

The Transformer architecture (Vaswani et al., 2017) replaced recurrent networks as the dominant sequence model. Its core mechanism is **self-attention**: each

token attends to every other token in the sequence with learned weights. Transformers underpin BERT, GPT, T5, and modern LLMs, and have expanded into vision (ViT), audio (wav2vec2), and tabular domains.

13.46.2 Prerequisites

Neural networks; attention concept; tokenisation basics.

13.46.3 Theory

Scaled dot-product attention:

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V,$$

where Q, K, V are query, key, and value projections of the input.

Multi-head attention runs h parallel attention heads and concatenates outputs.

Transformer block = multi-head self-attention + feedforward + residual + layer norm, stacked many times.

Positional encoding injects sequence order (sinusoidal or learned).

13.46.4 Assumptions

Sufficient data (transformers are data-hungry); proper tokenisation (BPE or WordPiece for text); positional encoding matches task.

13.46.5 R Implementation

```
library(torch)

torch_manual_seed(2026)

# Minimal transformer block: multi-head attention + feedforward
block <- nn_module(
  initialize = function(d_model = 32, n_heads = 4, d_ff = 64) {
    self$mha <- nn_multihead_attention(embed_dim = d_model,
                                       num_heads = n_heads,
                                       batch_first = TRUE)
    self$ln1 <- nn_layer_norm(d_model)
    self$ff  <- nn_sequential(nn_linear(d_model, d_ff),
                              nn_relu(),
                              nn_linear(d_ff, d_model))
    self$ln2 <- nn_layer_norm(d_model)
```

```
  },
  forward = function(x) {
    attn <- self$mha(x, x, x)[[1]]
    x <- self$ln1(x + attn)
    x <- self$ln2(x + self$ff(x))
    x
  }
)

model <- block()
seq <- torch_randn(2, 10, 32)        # batch=2, seq=10, d_model=32
out <- model(seq)
out$shape                             # [2, 10, 32]
```

13.46.6 Output & Results

Output sequence of the same shape as the input; the block transforms representations by letting each token attend to the full context.

13.46.7 Interpretation

"A single transformer block with 4-head self-attention and feedforward sublayers transformed a (10-token, 32-dim) sequence. Stacking 6-48 such blocks gives modern NLP models; pretrained backbones (BERT, RoBERTa) are the practical starting point."

13.46.8 Practical Tips

- Transformers need a lot of data; for small-data tasks, fine-tune a pretrained backbone rather than training from scratch.
- Use efficient attention variants (FlashAttention, sparse attention) for long sequences.
- Masking (causal for GPT, padding for BERT) is essential; forgetting it silently breaks training.
- Positional encoding: sinusoidal for classical transformers, rotary (RoPE) for modern LLMs.
- For R, practical transformer pipelines often call HuggingFace via `reticulate`; `torch` in R implements building blocks but lacks a huge pretrained-model ecosystem.

13.46.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.46.10 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.47 Variable Importance

13.47.1 Introduction

Variable importance ranks features by contribution to model accuracy. It is the most common first-pass interpretability tool for black-box models. Tree ensembles offer two native measures: impurity-based (mean decrease in Gini or variance) and permutation-based (drop in accuracy when the feature is shuffled).

13.47.2 Prerequisites

Decision trees; random forests; cross-validation.

13.47.3 Theory

Impurity importance: sum the impurity decrease over all splits using feature j. Biased toward high-cardinality features and correlated features.

Permutation importance: shuffle feature j in held-out data; measure the drop in performance. Unbiased for correlated features as a group but can split importance between correlated pairs.

Conditional permutation and SHAP values handle correlation more rigorously (at higher cost).

13.47.4 Assumptions

Sufficient held-out data for permutation; features correctly treated as continuous or categorical.

13.47.5 R Implementation

```r
library(ranger); library(vip)

set.seed(2026)
n <- 500
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n), x3 = rnorm(n),
  x4 = rnorm(n), x5 = rnorm(n),
  y  = sin(rnorm(n)) + 0.3 * rnorm(n)^2
)

rf <- ranger(y ~ ., data = df, num.trees = 500,
             importance = "permutation")
imp <- sort(rf$variable.importance, decreasing = TRUE)
print(round(imp, 3))

vip(rf, num_features = 5) +
  ggplot2::labs(title = "Permutation importance (ranger RF)")
```

13.47.6 Output & Results

Sorted importance scores; the top feature dominates, others are close to zero on this mostly-noise data.

13.47.7 Interpretation

"Permutation importance identified x_1 as the dominant predictor (12 % RMSE increase when shuffled); other features contributed negligibly, consistent with the data-generating process."

13.47.8 Practical Tips

- Use permutation importance when correctness matters; impurity importance is cheap but biased.
- Drop correlated features or use grouped importance if groups of correlated variables share the signal.
- Importance alone does not reveal direction of effect; pair with partial dependence or SHAP.
- Normalise importance by the baseline model error to express as a relative increase.
- For deep learning, use integrated gradients or SHAP; RF-style permutation is not standard.

13.47.9 Reporting

When reporting variable importance, always state which flavour was used (impurity, permutation, drop-column, or SHAP), the loss against which importance is measured, and whether the held-out test set or the training data drove the calculation. Permutation importance computed on training data overstates the apparent role of variables that the model overfits to, so the test-set version is the more defensible default. In a manuscript, include a small bar chart or table for the top features and note explicitly that importance is a measure of model reliance, not of causal effect or biological mechanism. Where features are correlated, mention that shared signal can split importance across columns and that grouped-permutation summaries may be more interpretable.

13.47.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.47.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.48 XGBoost

13.48.1 Introduction

XGBoost (eXtreme Gradient Boosting), introduced by Chen and Guestrin in 2016, extends classical gradient boosting with L1 and L2 regularisation of leaf weights, a second-order Taylor expansion of the loss, sparsity-aware split finding, and histogram-based methods that scale to billions of rows. It has dominated competitive machine-learning benchmarks on structured tabular data for nearly a decade and remains the strongest single off-the-shelf model for most regression and classification problems on tables.

13.48.2 Prerequisites

A working understanding of gradient boosting, decision trees, regularisation (L1/L2 penalties), and cross-validation for hyperparameter tuning.

13.48.3 Theory

The XGBoost objective is

$$\mathcal{L} = \sum_i L(y_i, \hat{y}_i) + \sum_k \Omega(f_k), \qquad \Omega(f) = \gamma T + \tfrac{1}{2}\lambda \|w\|^2,$$

where L is the per-observation loss (squared error, log-loss, etc.), f_k is the k-th regression tree, T is the number of leaves, w are the leaf weights, γ penalises tree complexity, and λ regularises leaf weights. A second-order Taylor expansion of L yields analytic optimal leaf weights and a tractable split score.

Key hyperparameters: `eta` (learning rate), `max_depth`, `min_child_weight`, `lambda` and `alpha` (L2 and L1 leaf-weight regularisation), `gamma` (minimum loss reduction for a split), `subsample` (row sampling), `colsample_bytree` (column sampling at tree level).

13.48.4 Assumptions

Dataset is tabular; labels suit the chosen objective (binary, multiclass, count, regression); sufficient sample size to support tree-based interactions and depth.

13.48.5 R Implementation

```
library(xgboost)

set.seed(2026)
n <- 600
X <- matrix(rnorm(n * 5), n, 5)
y <- as.numeric(plogis(X[, 1] + 0.5 * X[, 2]^2) > runif(n))

idx <- sample(n, n * 0.8)
d_tr <- xgb.DMatrix(X[idx, ], label = y[idx])
d_te <- xgb.DMatrix(X[-idx, ], label = y[-idx])

params <- list(objective = "binary:logistic", eta = 0.05,
               max_depth = 4, subsample = 0.8,
               colsample_bytree = 0.8,
               eval_metric = "logloss")

bst <- xgb.train(params = params, data = d_tr,
```

```
                  nrounds = 500,
                  watchlist = list(val = d_te),
                  early_stopping_rounds = 20,
                  verbose = 0)

cat("Best iter:", bst$best_iteration,
    "  best logloss:", round(bst$best_score, 4), "\n")

xgb.importance(model = bst)[1:5, ]
```

13.48.6 Output & Results

`xgb.train()` returns a fitted `xgb.Booster` with the early-stopping iteration count, best validation metric, and tree structure. `xgb.importance()` ranks features by gain (the average loss reduction attributed to splits on that feature), cover, or frequency.

13.48.7 Interpretation

A reporting sentence: "XGBoost with `eta=0.05`, `max_depth=4`, row and column subsampling at 0.8, converged after 217 boosting rounds with early stopping at 20-round patience; validation log-loss 0.38, ROC AUC 0.92. Feature-gain importance identified x_1 and x_2 as the strongest contributors, consistent with the simulated data-generating process." Always state the hyperparameters, early-stopping settings, and validation metrics.

13.48.8 Practical Tips

- Always use early stopping (`early_stopping_rounds`) to avoid over-training; without it, XGBoost trains until `nrounds` regardless of whether the validation loss has plateaued.
- Choose `objective` to match the target: `"reg:squarederror"` for regression, `"binary:logistic"` for binary classification, `"multi:softprob"` for multi-class, `"count:poisson"` for counts, `"survival:cox"` for time-to-event.
- Tune with Bayesian optimisation (`ParBayesianOptimization`, `tune::tune_bayes()`) rather than grid search; XGBoost has many correlated knobs and grid search wastes evaluations.
- `tree_method="hist"` accelerates training 5–10× on large data; `"gpu_hist"` adds another order of magnitude on GPU.
- Monotone constraints (`monotone_constraints`) enforce domain-knowledge directions for predictor-outcome relationships; useful in regulated industries (credit risk, healthcare).
- For interpretation beyond feature importance, use SHAP values (`xgboost::predict(...,predcontrib=TRUE)` or the `shapviz` package); SHAP gives per-prediction explanations.

13.48.9 R Packages Used

`xgboost` for the canonical R interface; `parsnip::boost_tree()` for the tidymodels wrapper; `lightgbm` and `catboost` as competitive alternatives; `shapviz` and `iml` for SHAP-based interpretation; `ParBayesianOptimization` for efficient hyperparameter tuning.

13.48.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

13.48.11 See also — labs in this chapter

- Cross-validation, nested CV, bootstrap .632+
- Regularisation: ridge, lasso, elastic net
- Trees, random forests, gradient boosting
- Interpretability and SHAP
- Tabular neural networks with torch
- Imaging and sequence models intro
- tidymodels pipelines
- FDR, knockoffs, replication crisis

13.49 Week 1, Session 1 — Cross-validation, nested CV, bootstrap .632+

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.49.1 Learning objectives

- Implement k-fold and repeated k-fold cross-validation from scratch and with a package, and report an honest generalisation error.
- Explain why selecting a model *and* tuning a hyperparameter on the same fold inflates performance, and use nested CV to fix it.
- Compute a bootstrap .632+ estimate and say when it is preferable to CV.

13.49.2 Prerequisites

Sampling variability; comfort with `caret` or base-R model fitting.

13.49.3 Background

Every predictive model reports an error on the data it was fit to. That number is almost always optimistic, because the fit has had a chance to memorise noise as well as signal. Cross-validation and the bootstrap are two resampling strategies that simulate the experience of scoring the model on data it has not seen. They are not interchangeable: they make different trade-offs between bias and variance, and they behave differently under high-dimensional settings.

A particular failure mode that recurs in the applied literature is *double dipping*: using the same held-out fold both to choose a hyperparameter and to report the final error. The fold is then no longer held out, and the reported error slides back toward the training error. Nested cross-validation is the clean solution: the inner loop picks hyperparameters, the outer loop reports error.

The bootstrap gives a different window onto generalisation. The plain bootstrap resamples with replacement; the .632 and .632+ variants weight the in-bag and out-of-bag errors to reduce bias when the learner overfits. In very small samples the .632+ estimator is often the most stable, and it is worth carrying in your toolbox alongside CV.

A practical note: when you write up a modelling study, the generalisation error is the headline number. If you tuned, you must disclose how. If the inner tuning cost was hidden in the outer fold, a careful reader will notice that the reported error is too good and cannot be reproduced. Nested CV is not a cosmetic fix; it is the difference between a number that generalises and a number that does not.

13.49.4 Setup

```
library(tidyverse)
library(glmnet)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

13.49.5 1. Goal

Estimate the generalisation error of a lasso on a moderate p, small n simulated regression problem, first naively, then with a proper nested cross-validation.

13.49.6 2. Approach

We simulate `n=120` observations with `p=40` features, of which only 5 are true signal. We will (i) fit a single `cv.glmnet`, (ii) report the CV-minimising error, then (iii) wrap that in an outer 5-fold loop to show that the inner CV error is optimistic.

```r
n <- 120; p <- 40
X <- matrix(rnorm(n * p), n, p)
beta <- c(rep(1.5, 5), rep(0, p - 5))
y <- as.numeric(X %*% beta + rnorm(n, sd = 1.0))

tibble(yhat_naive = as.numeric(X %*% beta), y = y) |>
  ggplot(aes(yhat_naive, y)) +
  geom_point(alpha = 0.6) +
  geom_abline(slope = 1, intercept = 0, colour = "grey50") +
  labs(x = "True linear predictor", y = "Observed y")
```

13.49.7 3. Execution

Naive single-layer CV.

```r
cv1 <- cv.glmnet(X, y, alpha = 1, nfolds = 5)
naive_mse <- min(cv1$cvm)
naive_mse
```

Nested CV. The inner `cv.glmnet` picks `lambda`; the outer 5 folds produce the reported MSE.

```r
K <- 5
folds <- sample(rep(1:K, length.out = n))
outer_mse <- numeric(K)
for (k in 1:K) {
  tr <- folds != k; te <- folds == k
  fit_k <- cv.glmnet(X[tr, ], y[tr], alpha = 1, nfolds = 5)
  yhat  <- as.numeric(predict(fit_k, newx = X[te, ], s = "lambda.min"))
  outer_mse[k] <- mean((y[te] - yhat)^2)
}
nested_mse <- mean(outer_mse)
c(naive = naive_mse, nested = nested_mse)
```

13.49.8 4. Check

Bootstrap .632+ error as a cross-check.

```r
B <- 50
errs_in <- errs_oob <- numeric(B)
for (b in 1:B) {
  idx <- sample.int(n, n, replace = TRUE)
  oob <- setdiff(seq_len(n), unique(idx))
  fit_b <- cv.glmnet(X[idx, ], y[idx], alpha = 1, nfolds = 5)
  yhat_in  <- as.numeric(predict(fit_b, newx = X[idx, ],  s =
  ↪  "lambda.min"))
```

```
  yhat_oob <- as.numeric(predict(fit_b, newx = X[oob, ],  s =
↪  "lambda.min"))
  errs_in[b]  <- mean((y[idx] - yhat_in)^2)
  errs_oob[b] <- mean((y[oob] - yhat_oob)^2)
}
err_app <- mean(errs_in); err_oob <- mean(errs_oob)
err_632 <- 0.368 * err_app + 0.632 * err_oob
c(apparent = err_app, oob = err_oob, boot632 = err_632,
  nested = nested_mse)
```

13.49.9 5. Report

> On simulated data ($n = 120$, $p = 40$, five true signals), a lasso with tuning parameter chosen by 5-fold CV achieved an out-of-sample MSE of `round(nested_mse,2)` under nested 5×5 CV. The naive single-layer CV understated the error at `round(naive_mse,2)`; the bootstrap .632 estimator gave `round(err_632,2)`, which bracketed the nested estimate.

The naive number looks tighter, but it reflects the choice of `lambda` as well as the fit. Only the nested number is a fair estimate of what this procedure — tune-then-fit — will do on new data.

13.49.10 Common pitfalls

- Tuning on the test fold and reporting the resulting error.
- Using a single 80/20 split in high dimensions; variance across splits can exceed the effect of interest.
- Bootstrapping without tracking in-bag vs out-of-bag, which inflates the estimate back to the apparent error.

13.49.11 Further reading

- Hastie, Tibshirani, Friedman, *Elements of Statistical Learning*, §7.10–7.11.
- Varma S, Simon R (2006), *Bias in error estimation when using cross-validation for model selection.*
- Efron B, Tibshirani R (1997), *Improvements on cross-validation: the .632+ bootstrap method.*

13.49.12 Session info

```
sessionInfo()
```

13.49.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.50 Week 4, Session 4 — FDR, knockoffs, replication crisis

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.50.1 Learning objectives

- Simulate p-values under a mixture of null and alternative hypotheses and apply Benjamini–Hochberg FDR control.
- Describe the knockoff filter in words and explain what "exchangeable" means in that context.
- Connect FDR discipline to the replication crisis in biomedicine.

13.50.2 Prerequisites

Multiple-testing basics.

13.50.3 Background

Multiple-testing corrections come in two families: family-wise error rate (Bonferroni, Holm) control the probability of *any* false discovery; false-discovery rate methods (Benjamini–Hochberg, Storey's q-value) control the *proportion* of false discoveries among the discoveries. FDR is the appropriate target for the typical screen with thousands of simultaneous tests and a tolerance for some false positives on the shortlist.

Knockoffs are a newer idea that replace the classical p-value route entirely. For each feature a "knockoff" variable is constructed that matches the feature's joint distribution with other features but is conditionally independent of the outcome. A statistic comparing the original and the knockoff then controls the FDR without distributional assumptions on the test statistic, provided the knockoffs are exchangeable.

The replication crisis in biomedicine is not usually a failure of FDR in the technical sense; it is usually a failure to control *researcher degrees of freedom* — the multiple analyses that precede the reported test. No post-hoc correction can rescue a flexible, undocumented workflow.

Pre-registration, code availability, and transparent reporting discipline the part of the pipeline that FDR cannot. They are not optional extras.

13.50.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

13.50.5 1. Goal

Simulate 1 000 tests, most null and a few alternative, and compare uncorrected, Bonferroni, and BH discovery sets.

13.50.6 2. Approach

```
m <- 1000
m1 <- 50
is_alt <- c(rep(TRUE, m1), rep(FALSE, m - m1))
mu <- ifelse(is_alt, 3, 0)
z <- rnorm(m, mu, 1)
p <- 2 * (1 - pnorm(abs(z)))

tibble(p = p, is_alt = is_alt) |>
  ggplot(aes(p, fill = is_alt)) +
  geom_histogram(bins = 40, position = "identity", alpha = 0.7) +
  labs(x = "p-value", y = "count")
```

13.50.7 3. Execution

```
bonf <- p.adjust(p, method = "bonferroni")
bh   <- p.adjust(p, method = "BH")

res <- tibble(
  method = c("raw p<0.05", "Bonferroni<0.05", "BH q<0.05"),
  discovered  = c(sum(p < 0.05), sum(bonf < 0.05), sum(bh < 0.05)),
  true_pos    = c(sum(p[is_alt] < 0.05),
                  sum(bonf[is_alt] < 0.05),
                  sum(bh[is_alt] < 0.05)),
  false_pos   = c(sum(p[!is_alt] < 0.05),
                  sum(bonf[!is_alt] < 0.05),
                  sum(bh[!is_alt] < 0.05))
) |> mutate(fdp = false_pos / pmax(discovered, 1))
res
```

13.50.8 4. Check

Knockoff discussion — the idea, in words.

```
library(knockoff)
# For a linear regression y ~ X with X gaussian, construct
# Xk matching the second-order structure of X but conditionally
# independent of y given X. Then fit a penalised regression on
# [X, Xk]; the per-feature statistic W_j = |beta_j| - |beta_{j+p}|
# controls the FDR among selected features.
ko <- knockoff.filter(X, y, fdr = 0.1)
ko$selected
```

13.50.9 5. Report

> On 1 000 simulated tests (50 alternative, 950 null), raw p-value cut-off at 0.05 produced `res$discovered[1]` discoveries with a false-discovery proportion of `round(res$fdp[1],2)`. Bonferroni correction produced `res$discovered[2]` discoveries; the BH procedure at $q < 0.05$ produced `res$discovered[3]` with a false-discovery proportion of `round(res$fdp[3],2)`.

Knockoffs would give a similar FDR guarantee via a different route, and would remain valid under correlation between features provided the joint distribution can be modelled.

13.50.10 Common pitfalls

- Reporting nominal $p < 0.05$ cut-offs in a high-dimensional screen.
- Applying BH to a set of tests that are not approximately independent or positive-dependent.
- Treating knockoffs as a drop-in for p-values without checking the exchangeability assumption.

13.50.11 Further reading

- Benjamini Y, Hochberg Y (1995), *Controlling the false discovery rate: A practical and powerful approach to multiple testing.*
- Barber RF, Candès EJ (2015), *Controlling the false discovery rate via knockoffs.*
- Ioannidis JPA (2005), *Why most published research findings are false.*

13.50.12 Session info

```
sessionInfo()
```

13.50.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.51 Week 2, Session 4 — Imaging and sequence models intro

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.51.1 Learning objectives

- Describe how a 2-D convolution extracts spatial features from an image.
- Describe how a 1-D convolution or a recurrent layer processes a sequence.
- Recognise when to use a CNN, RNN, or transformer for biomedical data.

13.51.2 Prerequisites

A neural-network training loop in `torch`.

13.51.3 Background

Convolutional neural networks (CNNs) exploit the spatial locality of image data: a small filter is slid across the image, producing a feature map for each filter. Stacking convolutional layers lets the network learn increasingly abstract features, from edges to textures to whole tissue structures. In biomedical imaging — histopathology, radiology, dermatology — CNNs underpin the majority of deep-learning applications.

For sequence data — gene sequences, electronic health record time series, wearable signals — the relevant operations are 1-D convolutions and attention. Transformers have become the dominant architecture for long sequences, but 1-D CNNs and gated recurrent networks remain competitive on shorter inputs and are easier to train. This lab illustrates the *shape* of the computation with a tiny hand-built 2-D convolution and a tiny 1-D convolution; the real training happens with frameworks that we mark `#|eval:false`.

The conceptual point: image and sequence models share the training tabular `torch` training loop. What differs is the architecture, the data pipeline, and the inductive biases (locality for CNNs, ordered context for RNNs and attention).

13.51.4 Setup

```
library(tidyverse)
set.seed(42)
```

```
theme_set(theme_minimal(base_size = 12))
```

13.51.5 1. Goal

Show manually computed convolutions on a small image and sequence, then sketch the `torch` code that would perform the same operations inside a trained model.

13.51.6 2. Approach

Create a toy 16×16 image with a bright square.

```
img <- matrix(0, 16, 16)
img[5:10, 5:10] <- 1
img <- img + matrix(rnorm(256, 0, 0.1), 16, 16)

image(t(img[16:1, ]), col = grey.colors(100), main = "toy image")
```

13.51.7 3. Execution

A manual 3×3 edge filter convolved with the image.

```
kernel <- matrix(c(-1, -1, -1, -1, 8, -1, -1, -1, -1), 3, 3)
conv2d <- function(img, k) {
  kh <- nrow(k); kw <- ncol(k)
  out <- matrix(0, nrow(img) - kh + 1, ncol(img) - kw + 1)
  for (i in seq_len(nrow(out))) for (j in seq_len(ncol(out)))
    out[i, j] <- sum(img[i:(i + kh - 1), j:(j + kw - 1)] * k)
  out
}
feat <- conv2d(img, kernel)
image(t(feat[nrow(feat):1, ]), col = grey.colors(100),
      main = "edge-filter feature map")
```

A 1-D convolution on a toy sequence.

```
seq1 <- c(rep(0, 10), rep(1, 5), rep(0, 10))
conv1d <- function(x, k) {
  out <- numeric(length(x) - length(k) + 1)
  for (i in seq_along(out)) out[i] <- sum(x[i:(i + length(k) - 1)] * k)
  out
}
feat1d <- conv1d(seq1, c(-1, 1))
tibble(i = seq_along(feat1d), v = feat1d) |>
  ggplot(aes(i, v)) + geom_line() +
  labs(x = "position", y = "edge response")
```

A tiny CNN in `torch` (sketch only).

```
library(torch)
cnn <- nn_sequential(
  nn_conv2d(1, 8, kernel_size = 3, padding = 1),
  nn_relu(),
  nn_max_pool2d(2),
  nn_conv2d(8, 16, kernel_size = 3, padding = 1),
  nn_relu(),
  nn_max_pool2d(2),
  nn_flatten(),
  nn_linear(16 * 4 * 4, 10)
)
```

A 1-D convolutional sequence model sketch.

```
library(torch)
seq_model <- nn_sequential(
  nn_conv1d(1, 8, kernel_size = 5, padding = 2),
  nn_relu(),
  nn_adaptive_avg_pool1d(1),
  nn_flatten(),
  nn_linear(8, 1)
)
```

13.51.8 4. Check

Inspect the convolution output shape and magnitude.

```
dim(feat)
summary(as.numeric(feat))
```

13.51.9 5. Report

> A 3×3 edge filter applied to a 16×16 toy image produced a feature map of shape `paste(dim(feat),collapse="×")`, with responses concentrated at the boundary of the bright square. Analogous 1-D convolutions applied to a binary step sequence produce sharp derivative-like responses at the transitions.

Real CNNs and sequence models compose many such filters, learned from data. The lab establishes the building block so the architectures in the literature read as stacks of familiar operations, not black boxes.

13.51.10 Common pitfalls

- Applying a CNN to tabular data; locality inductive bias is wasted.

- Forgetting input channel and batch dimensions in `torch`.
- Comparing accuracy of sequence models without specifying the tokenisation used.

13.51.11 Further reading

- LeCun Y, Bengio Y, Hinton G (2015), *Deep learning*.
- Vaswani A et al. (2017), *Attention is all you need*.

13.51.12 Session info

```
sessionInfo()
```

13.51.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.52 Week 2, Session 2 — Interpretability and SHAP

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.52.1 Learning objectives

- Produce permutation importance, partial-dependence, and Shapley values for a fitted black-box model, and interpret each plot.
- State the limitations of feature-attribution methods under feature correlation.
- Wrap a model with `DALEX` and `iml` for a standardised explanation pipeline.

13.52.2 Prerequisites

Trees and forests.

13.52.3 Background

Interpretability is not a single method; it is a cluster of tools that answer different questions. *Permutation importance* answers: how much worse does the model score if I scramble this feature? *Partial dependence* answers: averaging over all the other features, how does the prediction change as this feature varies? *SHAP* answers: for this specific prediction, how much did each feature contribute relative to the average prediction?

All three can mislead under feature correlation. Permuting a feature breaks its correlation with others, producing unrealistic inputs. Partial dependence averages over a marginal distribution that may not represent the data. SHAP — in particular TreeSHAP for trees — has several variants (interventional vs conditional) that answer subtly different causal questions. The robust practice is to use more than one method and to check that they tell a consistent story.

Interpretability is not a substitute for a causal model. A feature can have large SHAP values and large permutation importance and still be downstream of the outcome, a proxy for a confounder, or a leakage artefact. The explanation tells you what the model uses; whether that is what the world uses is a separate, harder question.

13.52.4 Setup

```
library(tidyverse)
library(ranger)
library(DALEX)
library(iml)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

13.52.5 1. Goal

Fit a random forest on the `palmerpenguins` dataset and produce three complementary interpretability artefacts.

13.52.6 2. Approach

```
d <- palmerpenguins::penguins |>
  tidyr::drop_na() |>
  mutate(species = factor(species))
ggplot(d, aes(bill_length_mm, body_mass_g, colour = species)) +
  geom_point(alpha = 0.7)
```

13.52.7 3. Execution

```
rf <- ranger(species ~ bill_length_mm + bill_depth_mm +
               flipper_length_mm + body_mass_g,
             data = d, probability = TRUE, num.trees = 500)

explainer <- DALEX::explain(
  model = rf,
  data  = d[, c("bill_length_mm", "bill_depth_mm",
```

```
                "flipper_length_mm", "body_mass_g")],
  y     = as.integer(d$species == "Adelie"),
  predict_function = function(m, newdata) predict(m,
   ↪  newdata)$predictions[, "Adelie"],
  label = "rf-Adelie",
  verbose = FALSE
)
vi <- DALEX::model_parts(explainer, type = "variable_importance")
plot(vi)
```

```
pdp <- DALEX::model_profile(explainer, variables = "bill_length_mm")
plot(pdp)
```

13.52.8 4. Check

A Shapley value for one instance.

```
pred_fun <- function(model, newdata) predict(model,
 ↪  newdata)$predictions[, "Adelie"]
predictor <- iml::Predictor$new(rf, data = d[, 3:6],
                                y = as.integer(d$species == "Adelie"),
                                predict.function = pred_fun)
sh <- iml::Shapley$new(predictor, x.interest = d[1, 3:6])
plot(sh)
```

13.52.9 5. Report

> On the penguins dataset, a random forest predicting the probability of *Adelie* species had permutation-importance values dominated by bill length and flipper length. Partial-dependence profiles showed that Adelie probability decreases sharply above a bill length of about 44 mm. For the first instance in the dataset, the Shapley attribution confirmed bill length as the primary contributor.

Report all three artefacts together, and note the correlation structure among the features when you do.

13.52.10 Common pitfalls

- Treating SHAP values as causal effects.
- Running permutation importance on correlated features without a conditional variant.
- Presenting partial-dependence plots outside the data's support.

13.52.11 Further reading

- Lundberg SM, Lee S-I (2017), *A unified approach to interpreting model predictions*.
- Molnar C, *Interpretable Machine Learning* (online book).

13.52.12 Session info

```
sessionInfo()
```

13.52.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.53 Week 1, Session 2 — Regularisation: ridge, lasso, elastic net

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.53.1 Learning objectives

- Fit ridge, lasso, and elastic-net regressions with `glmnet`, and explain what each penalty does to the coefficient path.
- Select `lambda` and `alpha` by cross-validation, and describe the bias–variance trade-off the choice implies.
- Interpret a coefficient path figure and a set of standardised coefficients.

13.53.2 Prerequisites

OLS and GLMs; the cross-validation lab.

13.53.3 Background

The ordinary least-squares solution is unbiased but, when p is close to or larger than n, it is unstable: small perturbations of the data move the fitted coefficients a long way. Regularisation trades a little bias for a lot of variance reduction by adding a penalty to the fitting criterion. Ridge shrinks all coefficients toward zero, lasso sets some exactly to zero, and elastic net blends the two. Each lives on the `glmnet` coefficient path and is selected by the same basic mechanism: cross-validation over `lambda` (and optionally `alpha`).

In biomedical data the motivation is rarely pure prediction. A lasso that selects 7 of 400 genes is also a shortlist for the next experiment. A ridge that keeps

every variable but shrinks their effects is an honest admission that signal is diffuse and that no small subset will carry the story. Elastic net is the default when variables come in correlated groups, because the lasso tends to pick one at random from a correlated cluster while ridge keeps them all.

Standardise before fitting. The penalty is scale-sensitive, and a variable in raw units will be penalised differently from the same variable in standardised units. `glmnet` standardises by default, but it is worth knowing what the coefficients refer to when you plot them.

13.53.4 Setup

```
library(tidyverse)
library(glmnet)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

13.53.5 1. Goal

Compare ridge, lasso, and elastic net on a simulated high-dimensional regression problem and pick a final model.

13.53.6 2. Approach

`n=100`, `p=200`, block-correlated design with a handful of true signals, to exercise both the sparsity of lasso and the group-handling of elastic net.

```
n <- 100; p <- 200
Z <- matrix(rnorm(n * (p / 10)), n, p / 10)
X <- Z[, rep(seq_len(p / 10), each = 10)] + 0.2 * matrix(rnorm(n * p), n,
 ↪  p)
beta <- c(rep(2, 5), rep(-1.5, 5), rep(0, p - 10))
y <- as.numeric(X %*% beta + rnorm(n))

tibble(i = seq_along(beta), beta = beta) |>
  ggplot(aes(i, beta)) + geom_segment(aes(xend = i, yend = 0)) +
  labs(x = "feature index", y = "true coefficient")
```

13.53.7 3. Execution

```
fit_ridge <- glmnet(X, y, alpha = 0)
fit_lasso <- glmnet(X, y, alpha = 1)
fit_enet  <- glmnet(X, y, alpha = 0.5)
par(mfrow = c(1, 3), mar = c(4, 4, 2, 1))
plot(fit_ridge, xvar = "lambda"); title("ridge")
```

```r
plot(fit_lasso, xvar = "lambda"); title("lasso")
plot(fit_enet,  xvar = "lambda"); title("elastic net")
```

13.53.8 4. Check

Pick `lambda` by CV for each; compare minimum CV MSE.

```r
cv_ridge <- cv.glmnet(X, y, alpha = 0,   nfolds = 5)
cv_lasso <- cv.glmnet(X, y, alpha = 1,   nfolds = 5)
cv_enet  <- cv.glmnet(X, y, alpha = 0.5, nfolds = 5)
tibble(
  model = c("ridge", "lasso", "enet"),
  min_cvmse = c(min(cv_ridge$cvm), min(cv_lasso$cvm), min(cv_enet$cvm)),
  nonzero   = c(sum(coef(cv_ridge, s = "lambda.min") != 0),
                sum(coef(cv_lasso, s = "lambda.min") != 0),
                sum(coef(cv_enet,  s = "lambda.min") != 0))
)
```

13.53.9 5. Report

> On a simulated `n=100`, `p=200` regression with 10 true signals, elastic net ($\alpha = 0.5$) achieved the lowest 5-fold CV MSE (`round(min(cv_enet$cvm),2)`) and retained `sum(coef(cv_enet,s="lambda.min")!=0)-1` features. Lasso was comparable but dropped one member of each correlated signal pair; ridge was smoothest but produced no sparse shortlist.

The choice is problem-dependent: if the next step is validation of a candidate biomarker panel, lasso or elastic net is preferable for the interpretable sparsity; if the next step is prediction only, ridge is often more stable.

13.53.10 Common pitfalls

- Interpreting lasso coefficients as unbiased effect estimates.
- Forgetting that correlated features share shrinkage: lasso may pick any one of them.
- Reporting CV-minimum error as generalisation error (see the cross-validation lab).

13.53.11 Further reading

- Friedman J, Hastie T, Tibshirani R (2010), *Regularization paths for generalized linear models via coordinate descent.*
- Zou H, Hastie T (2005), *Regularization and variable selection via the elastic net.*

13.53.12 Session info

```
sessionInfo()
```

13.53.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.54 Week 2, Session 3 — Tabular neural networks with torch

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.54.1 Learning objectives

- Sketch the architecture of a small feed-forward network for tabular regression using `torch`.
- Write a training loop with manual batching and an optimiser.
- Compare a neural network to a linear baseline and be honest about when the network is worth the complexity.

13.54.2 Prerequisites

Regularisation; basic gradient descent intuition.

13.54.3 Background

Neural networks rarely beat gradient-boosted trees on small tabular biomedical problems. They come into their own when the feature representation itself must be learned (images, sequences, free text) or when the sample size is large enough to feed a flexible model. On a feature matrix of hundreds of rows and tens of columns, a logistic or linear regression with appropriate regularisation is a better baseline and often a better final model.

That said, understanding the training loop — forward pass, loss, backward pass, optimiser step — is a prerequisite for everything in imaging, sequence, and modern generative modelling. This lab walks through that loop on a small simulated regression so the shapes and operations are transparent.

We mark the full training loop with `#|eval:false` because installation of `torch` downloads a library that may not be present in every rendering environment. A tiny parallel example using base R gradient descent is shown first so the ideas run end-to-end no matter what is installed.

13.54.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

13.54.5 1. Goal

Fit a small regression by manual gradient descent, then show the equivalent `torch` code with `#|eval:false`.

13.54.6 2. Approach

```
n <- 500; p <- 10
X <- matrix(rnorm(n * p), n, p)
beta <- c(2, -1, 0.5, rep(0, p - 3))
y <- as.numeric(X %*% beta + rnorm(n, 0, 0.5))
tibble(i = seq_len(n), y = y) |>
  ggplot(aes(i, y)) + geom_point(alpha = 0.3, size = 0.6)
```

13.54.7 3. Execution

Base-R gradient descent on the same problem.

```
w <- rep(0, p); lr <- 0.01
losses <- numeric(200)
for (step in 1:200) {
  yhat <- X %*% w
  resid <- as.numeric(y - yhat)
  grad <- -2 * t(X) %*% resid / n
  w <- w - lr * grad
  losses[step] <- mean(resid^2)
}
tibble(step = 1:200, loss = losses) |>
  ggplot(aes(step, loss)) + geom_line() +
  labs(x = "iteration", y = "mean squared error")
```

The equivalent `torch` code.

```
library(torch)

x_t <- torch_tensor(X, dtype = torch_float())
y_t <- torch_tensor(matrix(y, ncol = 1), dtype = torch_float())

net <- nn_sequential(
  nn_linear(p, 16), nn_relu(),
```

```
  nn_linear(16, 1)
)
optimizer <- optim_adam(net$parameters, lr = 1e-2)
loss_fn <- nn_mse_loss()

for (epoch in 1:200) {
  optimizer$zero_grad()
  pred <- net(x_t)
  loss <- loss_fn(pred, y_t)
  loss$backward()
  optimizer$step()
}
```

13.54.8 4. Check

Compare our gradient-descent weights against the OLS solution.

```
lm_coef <- as.numeric(coef(lm(y ~ X - 1)))
tibble(var = seq_along(w), gd = as.numeric(w), lm = lm_coef, truth =
↪  beta) |>
  pivot_longer(-var) |>
  ggplot(aes(var, value, colour = name)) +
  geom_point() + geom_line(alpha = 0.5) +
  labs(x = "feature index", y = "coefficient")
```

13.54.9 5. Report

> A small feed-forward network and a linear model both recover the true coefficients on a simulated `n=500`, `p=10` regression. Under this signal-to-noise ratio and sample size, the linear model is the appropriate baseline; the neural network is shown here for machinery, not as the recommended estimator.

The training-loop pattern — zero grads, forward, loss, backward, step — is the same whether you are fitting a 3-layer MLP on penguins or a ResNet on histology. The difference is the architecture, the data pipeline, and the compute.

13.54.10 Common pitfalls

- Fitting a deep MLP on a small tabular problem and then reporting training loss as generalisation error.
- Forgetting to standardise inputs before training; learning rates are calibrated on normalised scales.
- Ignoring reproducibility: set both the R seed and the torch seed.

13.54.11 Further reading

- Goodfellow I, Bengio Y, Courville A, *Deep Learning*, ch. 6.
- `torchforR` book by Keydana (online).

13.54.12 Session info

```
sessionInfo()
```

13.54.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.55 Week 2, Session 5 — tidymodels pipelines

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.55.1 Learning objectives

- Build a `tidymodels` recipe, workflow, and tuning grid for a binary classification task.
- Split data for CV properly and collect metrics across resamples.
- Finalise a model on the full training data and evaluate on a test split.

13.55.2 Prerequisites

Cross-validation and regularisation.

13.55.3 Background

`tidymodels` organises the modelling workflow into composable objects: a *recipe* describes preprocessing, a *model specification* describes the algorithm and its tunable parameters, a *workflow* bundles them, and *tune* functions drive resampling. The value of this separation is reproducibility: the same recipe runs on training, tuning, and test data, and it is hard to leak information across the split.

This lab builds a full pipeline on `MASS::Pima.tr`: logistic regression with elastic net and tuned `mixture` and `penalty`. The same pattern extends to random forests, boosting, neural networks, or any model with a parsnip wrapper.

The convention to remember: tune on the training folds, collect metrics, finalise with the best parameters, fit once to all of the training data, then evaluate once on the test set. The test set is touched exactly once.

13.55.4 Setup

```
library(tidyverse)
library(tidymodels)
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

13.55.5 1. Goal

Fit an elastic-net logistic regression to predict diabetes status on `Pima.tr`, using a proper tidymodels pipeline.

13.55.6 2. Approach

```
d <- as_tibble(MASS::Pima.tr)
ggplot(d, aes(glu, bmi, colour = type)) + geom_point(alpha = 0.7)
```

13.55.7 3. Execution

Split, recipe, model, workflow, tune.

```
d_split <- initial_split(d, prop = 0.75, strata = type)
d_train <- training(d_split); d_test <- testing(d_split)
folds <- vfold_cv(d_train, v = 5, strata = type)

rec <- recipe(type ~ ., data = d_train) |>
  step_normalize(all_numeric_predictors())

mod <- logistic_reg(penalty = tune(), mixture = tune()) |>
  set_engine("glmnet")

wf  <- workflow() |> add_recipe(rec) |> add_model(mod)

grid <- grid_regular(penalty(), mixture(), levels = 5)
res  <- tune_grid(wf, resamples = folds, grid = grid,
                  metrics = metric_set(roc_auc, accuracy))
collect_metrics(res) |>
  filter(.metric == "roc_auc") |>
  arrange(desc(mean)) |>
  head(5)
```

13.55.8 4. Check

Select best, finalise, evaluate on the test set.

```
best <- select_best(res, metric = "roc_auc")
wf_final <- finalize_workflow(wf, best)
fit_final <- fit(wf_final, data = d_train)
preds <- predict(fit_final, d_test, type = "prob") |>
  bind_cols(predict(fit_final, d_test), d_test)
roc_auc(preds, truth = type, .pred_Yes, event_level = "second")
accuracy(preds, truth = type, estimate = .pred_class)
```

```
roc_curve(preds, truth = type, .pred_Yes, event_level = "second") |>
  autoplot()
```

13.55.9 5. Report

> An elastic-net logistic regression tuned on 5-fold CV, using a 5×5 grid over `penalty` and `mixture`, achieved a test ROC-AUC of `round(roc_auc(preds,truth=type,.pred_Yes,event_level="second")$.estimate,3)` on `MASS::Pima.tr`.

The pipeline pattern — recipe + workflow + tune + finalise — is the template we will reuse for every predictive-model lab hereafter.

13.55.10 Common pitfalls

- Normalising the entire dataset before splitting; leakage is then baked in.
- Reporting the mean CV metric as if it were a test-set metric.
- Tuning without stratifying folds on an imbalanced outcome.

13.55.11 Further reading

- Kuhn M, Silge J, *Tidy Modeling with R* (online book).

13.55.12 Session info

```
sessionInfo()
```

13.55.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

13.56 Week 2, Session 1 — Trees, random forests, gradient boosting

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

13.56.1 Learning objectives

- Fit a single decision tree, a random forest, and a gradient-boosted model on the same data, and compare their error.
- Read variable-importance output critically and know its biases.
- Explain when a tree ensemble is preferable to a linear model and when it is not.

13.56.2 Prerequisites

Cross-validation; regularisation.

13.56.3 Background

Tree-based models divide feature space by axis-aligned splits and fit a constant within each region. A single tree is interpretable but high variance; averaging many trees with resampling and random feature subsets gives random forests, which are among the strongest off-the-shelf methods on tabular data. Gradient boosting takes a different route: it fits trees sequentially, each correcting the residuals of the last, and usually wins on tabular benchmarks when carefully tuned.

Tree methods make few assumptions about the functional form of the response and handle interactions automatically. Their limitations are extrapolation (flat beyond the training range), variable-importance measures that can be misleading when features are correlated, and a general tendency to look better than they generalise unless honest resampling is used.

When reporting a random forest or gradient-boosted model, report the tuning procedure, the out-of-bag or CV error, the variable-importance method, and — if the goal is scientific — a permutation-based importance to cross-check the default Gini/gain importance.

13.56.4 Setup

```
library(tidyverse)
library(rpart)
library(ranger)
library(xgboost)
set.seed(42)
```

```
theme_set(theme_minimal(base_size = 12))
```

13.56.5 1. Goal

Predict `mpg` from `mtcars` with a single tree, a random forest, and gradient boosting; compare errors and importances.

13.56.6 2. Approach

```
d <- mtcars
ggplot(d, aes(wt, mpg, colour = factor(cyl))) +
  geom_point() + labs(colour = "cyl")
```

13.56.7 3. Execution

```
tree_fit <- rpart(mpg ~ ., data = d, cp = 0.01)
rf_fit   <- ranger(mpg ~ ., data = d, importance = "permutation",
                   num.trees = 500)
xg_fit   <- xgboost(
  data = as.matrix(d[, -1]), label = d$mpg,
  nrounds = 100, eta = 0.05, max_depth = 3,
  objective = "reg:squarederror", verbose = 0
)
```

13.56.8 4. Check

Honest error: CV for the tree and xgboost, OOB for ranger.

```
K <- 5
folds <- sample(rep(1:K, length.out = nrow(d)))
mse <- function(y, yhat) mean((y - yhat)^2)

get_err <- function() {
  err_tree <- err_rf <- err_xg <- numeric(K)
  for (k in 1:K) {
    tr <- folds != k; te <- folds == k
    f_tr <- d[tr, ]; f_te <- d[te, ]
    t1 <- rpart(mpg ~ ., data = f_tr, cp = 0.01)
    r1 <- ranger(mpg ~ ., data = f_tr, num.trees = 500)
    x1 <- xgboost(
      data = as.matrix(f_tr[, -1]), label = f_tr$mpg,
      nrounds = 100, eta = 0.05, max_depth = 3,
      objective = "reg:squarederror", verbose = 0
    )
```

```
    err_tree[k] <- mse(f_te$mpg, predict(t1, f_te))
    err_rf[k]   <- mse(f_te$mpg, predict(r1, f_te)$predictions)
    err_xg[k]   <- mse(f_te$mpg, predict(x1, as.matrix(f_te[, -1])))
  }
  c(tree = mean(err_tree), rf = mean(err_rf), xgb = mean(err_xg))
}
errs <- get_err()
errs
```

```
tibble(var = names(rf_fit$variable.importance),
       imp = as.numeric(rf_fit$variable.importance)) |>
  arrange(desc(imp)) |>
  ggplot(aes(reorder(var, imp), imp)) + geom_col() +
  coord_flip() + labs(x = NULL, y = "permutation importance")
```

13.56.9 5. Report

> On `mtcars` ($n = 32$), 5-fold CV MSEs were `round(errs["tree"],2)` (single tree), `round(errs["rf"],2)` (random forest), and `round(errs["xgb"],2)` (gradient boosting). Permutation importance from the random forest identified weight and displacement as the dominant predictors.

The CV errors are close because the sample is tiny and the problem is close to linear. On larger tabular problems the gap opens and gradient boosting typically wins after tuning; but the tiny-sample regime is where trees and boosting most often look like magic and almost always lie.

13.56.10 Common pitfalls

- Reporting the training error of a forest; always use OOB or CV.
- Trusting Gini/gain importance for correlated predictors.
- Letting xgboost overfit with low `eta` and high `nrounds` without early stopping.

13.56.11 Further reading

- Breiman L (2001), *Random forests*.
- Chen T, Guestrin C (2016), *XGBoost: A scalable tree boosting system*.

13.56.12 Session info

```
sessionInfo()
```

13.56.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 14

Clinical Biostatistics

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Diagnostic test accuracy, agreement (Bland-Altman, kappa, ICC), biomarker development, prediction-model reporting under TRIPOD-AI, and fairness audits. The chapter sits at the intersection of ML and biostatistics for a reason: that is where most regulatory submissions live.

This chapter contains **36 method pages** and **4 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

14.1 Method pages

Method	Source slug
Adaptive Trial Designs	`adaptive-design`
Alpha-Spending Functions	`alpha-spending`
Baseline Adjustment with ANCOVA	`baseline-adjustment-ancova`
Bland-Altman Limits of Agreement	`bland-altman-analysis`
Blinding Procedures	`blinding-procedures`
Block Randomisation	`block-randomization`
Clinical Equivalence Trials	`equivalence-clinical`
Cluster-Randomised Trials	`cluster-rct`
Cohen's Kappa	`cohens-kappa`
Conditional Power	`conditional-power`
Crossover RCT Design	`rct-design-crossover`
Cutpoint Selection	`cutpoint-selection`
Diagnostic Test Accuracy	`diagnostic-accuracy`
Factorial Trials	`factorial-trial`

Method	Source slug
Interim Analyses and Group Sequential Designs	`interim-analysis-group-sequential`
Intraclass Correlation Coefficient (ICC)	`icc-continuous-agreement`
ITT vs Per-Protocol Analysis	`itt-vs-pp-analysis`
Likelihood Ratios	`likelihood-ratios`
Minimisation Algorithm	`minimization-algorithm`
Missing Data in RCTs	`missing-data-rct`
Multiple Imputation	`multiple-imputation`
Non-Inferiority Margin Selection	`non-inferiority-margin`
O'Brien-Fleming Boundary	`obrien-fleming-boundary`
Parallel-Group RCT Design	`rct-design-parallel`
Pocock Boundary	`pocock-boundary`
Predictive Values and Prevalence	`predictive-values-prevalence`
Randomisation Methods	`randomization-methods`
Reliability and Cronbach's Alpha	`reliability-cronbach-alpha`
ROC Analysis	`roc-analysis`
Sample Size Re-Estimation	`sample-size-reestimation`
Sensitivity Analyses in Clinical Trials	`sensitivity-analysis-clinical`
Stepped-Wedge Trial	`stepped-wedge-trial`
Stratified Randomisation	`stratified-randomization`
Subgroup Analyses	`subgroup-analysis`
Subgroup Forest Plots	`forest-plot-subgroup`
Weighted Kappa	`weighted-kappa`

14.2 Labs

Lab
Diagnostic testing: Se, Sp, PPV, NPV, LR
Kappa, ICC, Bland–Altman
Biomarker statistics (Youden, NRI, decision curves)
TRIPOD-AI, fairness auditing, reproducibility at scale

14.3 Adaptive Trial Designs

14.3.1 Introduction

Adaptive designs allow pre-specified modifications – sample size, randomisation ratios, treatment-arm selection – based on data accrued during the trial. They promise efficiency but require careful statistical control to preserve Type I error. The FDA and EMA both provide detailed guidance on acceptable adaptive modifications.

14.3.2 Prerequisites

Group-sequential trials; alpha spending; sample-size calculation.

14.3.3 Theory

Common adaptive features: - **Sample-size re-estimation** (blinded or unblinded). - **Early stopping** for efficacy or futility. - **Arm selection** (drop inferior arms in multi-arm trials). - **Response-adaptive randomisation** (shift allocation toward effective arms). - **Population enrichment** (restrict to a responsive subgroup).

Maintaining Type I error under pre-specified adaptations requires methods like group-sequential boundaries, combination tests, or conditional error functions.

14.3.4 Assumptions

Adaptation rules are fully pre-specified (protocol, SAP); unblinded information access is tightly controlled (Data Monitoring Committee); adjustments are statistically valid.

14.3.5 R Implementation

```
library(rpact)

# Group-sequential design with O'Brien-Fleming boundary, 3 analyses
design <- getDesignGroupSequential(
  sided = 2, alpha = 0.025, beta = 0.2,
  typeOfDesign = "OF",
  informationRates = c(0.5, 0.75, 1)
)
kable_summary <- summary(design)
print(design)

# Plan sample size for a two-arm trial with continuous outcome
ssr <- getSampleSizeMeans(design = design,
                          alternative = 0.3, stDev = 1)
print(ssr)
```

14.3.6 Output & Results

Group-sequential boundaries and associated sample sizes per stage; cumulative Type I error preserved at alpha.

14.3.7 Interpretation

"The adaptive design applied O'Brien-Fleming boundaries at 50 %, 75 %, and 100 % information; stage-1 interim inefficacy boundary would stop at $p > 0.0002$, preserving overall alpha = 0.025."

14.3.8 Practical Tips

- Pre-specify every adaptation (including the decision rule) in the protocol.
- FDA/EMA require a detailed justification of the adaptive feature and its operating characteristics.
- Independent data monitoring committee is essential for efficacy or futility stopping.
- Simulation is often used to verify operating characteristics; report them.
- Post-hoc adaptations ("seamless" trial extensions) are exploratory, not confirmatory.

14.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.3.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.4 Alpha-Spending Functions

14.4.1 Introduction

Alpha-spending functions (Lan & DeMets, 1983) allow interim analyses at unscheduled times while preserving overall Type I error. The spending function $f(t)$ specifies the cumulative alpha budget at information fraction $t \in [0, 1]$; nominal alpha at each analysis is the increment $f(t_k) - f(t_{k-1})$.

14.4.2 Prerequisites

Group-sequential designs; information fraction.

14.4.3 Theory

Common spending functions: - **OF-type**: $f(t) = 2 - 2\Phi(z_{\alpha/2}/\sqrt{t})$. Approximates the OF boundary. - **Pocock-type**: $f(t) = \alpha \log(1 + (e-1)t)$. Approximates Pocock. - **Power family**: $f(t) = \alpha t^{\rho}$ with $\rho > 0$. - **Custom**: any non-decreasing f with $f(0) = 0, f(1) = \alpha$.

Flexibility: interim analyses can occur at arbitrary information fractions, resolving the boundary each time.

14.4.4 Assumptions

Information fractions are known (approximately); stopping rule applied as specified; test statistic is normal.

14.4.5 R Implementation

```r
library(rpact)

# OF spending, flexible information fractions
design <- getDesignGroupSequential(
  sided = 1, alpha = 0.025, beta = 0.2,
  typeOfDesign = "asOF",          # alpha-spending OF-type
  informationRates = c(0.4, 0.75, 1)
)

print(design$stageLevels)
plot(design, type = 1)

# Compare spending functions
f_of  <- getDesignGroupSequential(sided = 1, alpha = 0.025,
                                  typeOfDesign = "asOF",
                                  informationRates = seq(0.1, 1, 0.1))
f_poc <- getDesignGroupSequential(sided = 1, alpha = 0.025,
                                  typeOfDesign = "asP",
                                  informationRates = seq(0.1, 1, 0.1))
cbind(OF = cumsum(f_of$stageLevels),
      Pocock = cumsum(f_poc$stageLevels))
```

14.4.6 Output & Results

Cumulative alpha at each information fraction for both spending families; OF delays alpha consumption, Pocock distributes it earlier.

14.4.7 Interpretation

"The alpha-spending design at information fractions 0.4, 0.75, 1.0 under OF-type spending allocated cumulative alpha of 0.001, 0.013, 0.025 respectively, enabling flexibility in interim timing without inflating Type I error."

14.4.8 Practical Tips

- Alpha spending is the **standard** for modern confirmatory group-sequential trials.
- Re-estimate information fractions at each interim if accrual deviates from plan.
- Information is usually subject-count for continuous outcomes, event-count for time-to-event.
- Never spend more alpha than the planned cumulative function at the current information fraction.
- For futility, use separate **beta-spending** functions (non-binding boundaries are standard).

14.4.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.4.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.5 Baseline Adjustment with ANCOVA

14.5.1 Introduction

Analysis of Covariance (ANCOVA) regresses the outcome on treatment and baseline value jointly. Compared to a naive change-from-baseline t-test, ANCOVA gains precision, handles regression to the mean correctly, and reduces SE roughly by a factor of $\sqrt{1-\rho^2}$ where ρ is baseline-outcome correlation.

14.5.2 Prerequisites

Linear regression; correlation; change scores.

14.5.3 Theory

Naive change-score analysis: compare $\Delta = Y_{\text{post}} - Y_{\text{baseline}}$ across arms. Unbiased under randomisation but less efficient than ANCOVA when baseline correlates with outcome.

ANCOVA: $Y_{\text{post}} = \alpha + \beta_{\text{trt}} \cdot T + \beta_{\text{base}} \cdot Y_{\text{baseline}} + \varepsilon$. Treatment effect β_{trt} has lower SE than the change-score test.

Regression to the mean: if groups have different baselines by chance, change scores bias toward the difference; ANCOVA corrects for this.

14.5.4 Assumptions

Linear relationship between baseline and outcome; no treatment-by-baseline interaction (common extension: stratify or add interaction term).

14.5.5 R Implementation

```r
set.seed(2026)
n <- 200
baseline <- rnorm(n, 10, 2)
arm      <- factor(rep(c("ctrl", "trt"), each = n/2))

# Outcome correlated with baseline; true trt effect = 1
outcome <- 0.7 * baseline + ifelse(arm == "trt", 1, 0) +
           rnorm(n, 0, 1)

# Naive change-score analysis
change <- outcome - baseline
t.test(change ~ arm)

# ANCOVA
fit <- lm(outcome ~ arm + baseline)
summary(fit)$coefficients

# SE comparison
sd(change[arm == "trt"] - mean(change[arm == "trt"]))
```

14.5.6 Output & Results

ANCOVA estimates the treatment effect with substantially smaller SE than the change-score t-test when baseline-outcome correlation is non-trivial.

14.5.7 Interpretation

"ANCOVA estimated the treatment effect as 1.02 (95 % CI 0.74-1.30, $p < 0.001$) with ~40 % lower SE than the change-score analysis, leveraging the 0.7 baseline-outcome correlation."

14.5.8 Practical Tips

- Use ANCOVA for any continuous outcome where a baseline measurement is available.
- Even with small baseline-outcome correlation (0.3), ANCOVA improves power.
- The "change from baseline" as an outcome is a special case of ANCOVA with $\beta_{\text{base}} = 1$ forced; ANCOVA with free β is preferred.
- Pre-specify baseline adjustment in the SAP; post-hoc addition risks bias.
- EMA guidance on baseline adjustment: stratification and ANCOVA are both acceptable; ANCOVA is more efficient when baseline is continuous.

14.5.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.5.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.6 Bland-Altman Limits of Agreement

14.6.1 Introduction

The Bland-Altman plot (1986) compares two measurement methods by plotting their difference against their mean. It exposes systematic bias, proportional bias, and the 95 % limits of agreement within which most differences fall. It is the standard graphical summary for method-comparison studies and has largely replaced correlation-based summaries.

14.6.2 Prerequisites

Paired measurements; method-comparison basics.

14.6.3 Theory

For paired measurements (A_i, B_i): - **Bias**: mean difference $\bar{d} = \overline{A_i - B_i}$. - **Limits of agreement**: $\bar{d} \pm 1.96 \cdot s_d$.

If 95 % of differences lie within the limits and the limits are clinically acceptable, the two methods can be used interchangeably. Proportional bias shows as a trend in the scatter.

14.6.4 Assumptions

Differences are approximately normal; differences do not systematically depend on the mean (check with regression); replicates are handled appropriately if present.

14.6.5 R Implementation

```r
library(ggplot2)

set.seed(2026)
n <- 100
truth <- rnorm(n, 10, 2)
A <- truth + rnorm(n, 0, 0.5)            # method A
B <- truth + 0.3 + rnorm(n, 0, 0.5)      # method B (small bias)

df <- data.frame(A, B,
                 mean = (A + B) / 2,
                 diff = A - B)

bias <- mean(df$diff)
sd_d <- sd(df$diff)
loa  <- c(bias - 1.96 * sd_d, bias + 1.96 * sd_d)

ggplot(df, aes(mean, diff)) +
  geom_point(colour = "#2A9D8F") +
  geom_hline(yintercept = bias, linetype = 1) +
  geom_hline(yintercept = loa, linetype = 2) +
  labs(x = "Mean of A and B", y = "A - B",
       title = "Bland-Altman plot",
       subtitle = sprintf("Bias %.2f; 95%% LoA [%.2f, %.2f]",
                          bias, loa[1], loa[2])) +
  theme_minimal()
```

14.6.6 Output & Results

A scatter of differences vs means with solid bias line and dashed LoA; the simulated systematic bias of -0.3 is recovered.

14.6.7 Interpretation

"Bland-Altman analysis revealed a bias of -0.3 units with 95 % limits of agreement (-1.7, 1.1). If the clinically acceptable limit is +-2 units, the methods are interchangeable for most practical purposes."

14.6.8 Practical Tips

- Always show both bias and LoA; correlation alone does not reveal systematic bias.
- Check for proportional bias by regressing differences on means; a non-zero slope indicates non-constant bias.
- For replicated measurements per subject, adjust the LoA calculation (Bland-Altman 1999 extension).
- Report the clinical acceptance criterion **before** calculating LoA; otherwise post-hoc thresholding biases conclusions.
- Paired with ICC, Bland-Altman gives both a quantitative and visual summary of agreement.

14.6.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.6.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.7 Blinding Procedures

14.7.1 Introduction

Blinding (masking) keeps trial participants and personnel unaware of treatment assignments to minimise performance, assessment, and analyst biases. Single, double, triple, and quadruple blinding refer to which stakeholder groups are masked; each layer blocks a specific bias pathway.

14.7.2 Prerequisites

RCT design; sources of bias in clinical research.

14.7.3 Theory

- **Single blinding**: participant unaware; investigator aware.
- **Double blinding**: participant and investigator both unaware. Standard for drug vs placebo.
- **Triple blinding**: adds blinded outcome assessors (PROBE designs invert this).
- **Quadruple blinding**: adds blinded statisticians / analysts.

Each additional layer addresses bias but also increases operational complexity.

14.7.4 Assumptions

Identical appearance, taste, and packaging of active and placebo; emergency unblinding procedures are in place.

14.7.5 R Implementation

Blinding is operational – not a statistical analysis per se – but the **success of blinding** should be audited.

```
# Simulate an end-of-study blinding-integrity questionnaire
set.seed(2026)
n <- 200
guess <- factor(sample(c("active", "placebo", "don't know"),
                       n, replace = TRUE,
                       prob = c(0.4, 0.35, 0.25)))
true  <- factor(rep(c("active", "placebo"), each = n/2))

# James blinding index (range 0-1; 0.5 = good blinding)
tab <- table(guess, true)
n <- sum(tab)
# Simpler chi-square test of correct-guess rate
correct <- sum(guess == true)
binom.test(correct, length(guess), p = 0.5)
```

14.7.6 Output & Results

Binomial test of whether the correct-guess rate exceeds chance; $p > 0.05$ consistent with effective blinding.

14.7.7 Interpretation

"End-of-study unblinding revealed 56 % correct guesses (95 % CI 49-62 %, p = 0.12 vs chance), consistent with successful blinding. The study reports the result per CONSORT recommendations."

14.7.8 Practical Tips

- Match active and placebo precisely (taste, colour, packaging, schedule); any difference leaks information.
- For difficult-to-blind interventions (surgery, behavioural), blind at least outcome assessment.
- Test blinding at study end (e.g., James/Bang blinding index); report the result.
- Pre-specify emergency unblinding procedures; document all unblinding events.
- A statistician blinded to allocation prevents analysis-choice bias even in open-label trials.

14.7.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.7.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.8 Block Randomisation

14.8.1 Introduction

Block randomisation allocates clinical-trial participants in random permutations within fixed-size blocks, guaranteeing equal numbers of subjects in each treatment arm at every block boundary. It is the standard alternative to simple randomisation in clinical trials because it prevents the substantial arm-size imbalance that simple randomisation can produce in small trials, in early enrolment phases, or at any moment when the trial is paused for an interim analysis. Block randomisation is now mandated or strongly recommended by the ICH-E9 statistical-principles guideline, by CONSORT for randomisation reporting, and by virtually every regulatory authority's clinical-trial guidance.

14.8.2 Prerequisites

A working understanding of randomisation as the foundation of causal inference in trials, allocation concealment as the procedural safeguard against selection

bias, and the difference between simple, blocked, and stratified randomisation.

14.8.3 Theory

With block size B and two equally-allocated arms, each block contains $B/2$ assignments of each arm in random order. At every block boundary the arm counts are exactly balanced; between boundaries the maximum imbalance is $B/2$. With fixed block size, an investigator who knows the block size and observes $B-1$ allocations within a block can predict the final allocation — a serious threat in open-label trials. **Variable block sizes** (mixing, e.g., blocks of 4 and 6) defeat this predictability while preserving the boundary-balance guarantee.

For multi-arm trials with k arms in equal allocation, blocks must be multiples of k; for unequal allocation (e.g., $2:1$), blocks are multiples of the sum of allocation ratios.

14.8.4 Assumptions

Allocation concealment is preserved (the randomisation list is prepared in advance, kept off-site, and never available to enrolling investigators), the block-size distribution is documented in the statistical analysis plan but withheld from those who could exploit it, and randomisation is implemented through an interactive web-response system (IWRS) or equivalent with audit trail.

14.8.5 R Implementation

```r
library(blockrand)

set.seed(2026)

fix <- blockrand(n = 40, num.levels = 2,
                 levels = c("A", "B"),
                 block.sizes = 2)
table(fix$treatment)
head(fix, 8)

var_b <- blockrand(n = 40, num.levels = 2,
                   levels = c("A", "B"),
                   block.sizes = c(2, 3))
head(var_b, 10)
```

14.8.6 Output & Results

`blockrand()` returns an allocation schedule with exactly equal arm counts across the requested n participants. With variable block sizes, the schedule blends

blocks of different lengths in random order, preventing investigators from predicting the next allocation late in any single block. The schedule is typically exported to an IWRS and made available only to the trial pharmacist or unblinded statistician.

14.8.7 Interpretation

A reporting sentence: "Treatment allocation used block randomisation with variable block sizes of 4 and 6, generated by `blockrand` and managed via the trial's interactive web-response system. Allocation was stratified by site and disease severity, with blocks nested within strata. The block-size distribution was documented in the SAP and concealed from enrolling investigators throughout the trial. Final arm counts were balanced (200 in each arm of the 400-patient trial)." Always report block-size distribution and concealment procedure.

14.8.8 Practical Tips

- Avoid fixed block size alone in open-label trials; variable block sizes are now the de facto standard for randomised clinical trials and are explicitly recommended by ICH-E9 because they prevent end-of-block predictability without sacrificing balance.
- Document the block-size distribution in the SAP but withhold the actual block sequence from enrolling investigators; sharing the block sequence (even informally) compromises allocation concealment and is a recurring cause of CONSORT-cited methodological flaws.
- For stratified designs (by site, disease severity, age category), nest blocks within strata so that each stratum maintains independent arm balance; this is standard practice in multicentre trials and prevents centre-by-treatment confounding.
- Very large blocks reduce guessability further but also relax the boundary-balance guarantee at any given moment in enrolment; small to moderate blocks (2 to 6 in two-arm trials) are the standard compromise and adequately balance most trials.
- Commercial randomisation services (IWRS / IRT) manage the list, preserve concealment, and provide a tamper-proof audit trail; for sponsor-led trials the cost is justified by the regulatory protection.
- For trials with more than two arms, use stratified blocked randomisation with appropriately sized blocks; permuted-block randomisation extends naturally to any number of arms with equal or unequal allocation ratios.

14.8.9 R Packages Used

`blockrand` for canonical fixed and variable-block randomisation with built-in stratification support; `randomizr` for tidyverse-friendly randomisation including blocked and stratified designs; `bcrm` for biased-coin and minimisation alternatives; `ldhmm` and `psborrow` for adaptive randomisation in more complex designs;

`Mediana` for trial-design simulation including randomisation strategies.

14.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.8.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.9 Cluster-Randomised Trials

14.9.1 Introduction

Cluster-randomised trials (CRTs) randomise groups – clinics, schools, villages – rather than individuals. Used when an intervention must be delivered at cluster level (implementation, educational campaign) or when contamination between individuals would bias a standard RCT. Clustering inflates variance and must be accounted for in sample-size and analysis.

14.9.2 Prerequisites

Intraclass correlation (ICC); mixed-effects models.

14.9.3 Theory

Design effect: $DE = 1 + (m-1)\rho$, where m is average cluster size and ρ is the ICC. Effective sample size = actual N / DE. Sample-size calculations inflate by DE relative to individually-randomised trials.

Analysis accounts for clustering via mixed-effects models (random cluster intercept) or GEE with cluster-robust SE.

14.9.4 Assumptions

Clusters are exchangeable; intervention is applied uniformly within cluster; ICC estimate from pilot / literature is approximately correct.

14.9.5 R Implementation

```r
library(lme4); library(lmerTest)

set.seed(2026)
# 20 clusters, avg 15 patients per cluster
n_clusters <- 20
m_per <- 15
cluster <- factor(rep(1:n_clusters, each = m_per))
arm     <- factor(rep(c("ctrl", "trt"), each = (n_clusters/2) * m_per))
clust_re <- rep(rnorm(n_clusters, 0, 0.8), each = m_per)

y <- clust_re + ifelse(arm == "trt", 0.5, 0) +
     rnorm(n_clusters * m_per, 0, 1)

df <- data.frame(cluster, arm, y)
fit <- lmer(y ~ arm + (1 | cluster), data = df)
summary(fit)$coefficients

# Empirical ICC
vc <- as.data.frame(VarCorr(fit))
icc <- vc$vcov[1] / sum(vc$vcov)
cat("Estimated ICC:", round(icc, 3), "\n")
```

14.9.6 Output & Results

Cluster-random-effect-adjusted treatment effect (~0.5) with SE accounting for clustering; ICC estimate ~0.4.

14.9.7 Interpretation

“Cluster-randomised analysis estimated a 0.49 SD improvement (95 % CI 0.21-0.77, p = 0.001), accounting for the intra-cluster correlation of 0.42 via a random-cluster intercept.”

14.9.8 Practical Tips

- Even a small ICC (0.01) inflates required sample size substantially; budget accordingly.
- Pilot data or literature usually provides ICC; report both the planning value and the observed value.
- For few clusters (< 30), mixed-effects SE underestimates; use Kenward-Roger or Satterthwaite DF.
- Report per CONSORT extension for cluster trials; include number of clusters, cluster sizes, ICC.

- Stratified or matched-pair cluster designs improve balance when cluster count is small.

14.9.9 Reporting

A defensible cluster-trial report names the unit of randomisation, the unit of analysis, the planning ICC, and the achieved ICC, and explains how the analytical model handles their potential mismatch. Where the number of clusters is below thirty, state which small-sample correction was used for standard errors and degrees of freedom, since naive likelihood-based intervals are anti-conservative in this regime. If clusters varied substantially in size, mention whether weights were applied and how missing clusters or partial cluster dropout were handled, because differential cluster attrition can bias the estimated treatment effect even when individual-level missingness is modest.

14.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.9.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.10 Cohen's Kappa

14.10.1 Introduction

Cohen's kappa, introduced by Jacob Cohen in 1960, measures agreement between two raters on a categorical scale, with a correction for the agreement that would be expected by chance alone. Raw percent agreement can look impressive even when most of it reflects coincidence — two raters who both diagnose 90 % of patients as healthy will agree on at least 81 % of cases purely by chance. Kappa subtracts this chance baseline, leaving a more honest measure of the genuine signal in inter-rater agreement. It is now the de facto standard for inter-rater reliability on nominal categorical outcomes, widely used in imaging-rater studies, pathology grading, diagnostic-criteria validation, and any reliability assessment with two raters and a categorical scale.

14.10.2 Prerequisites

A working understanding of categorical data, contingency-table summaries, observed agreement as a percentage, and the concept of chance-expected agreement under independent raters.

14.10.3 Theory

Cohen's kappa is

$$\kappa = \frac{p_o - p_e}{1 - p_e},$$

where p_o is the observed proportion of cases on which the two raters agreed and p_e is the proportion expected by chance, computed as $p_e = \sum_k p_{1k} p_{2k}$ from the marginal proportions. Kappa ranges from -1 to $+1$: $\kappa = 1$ is perfect agreement, $\kappa = 0$ is exactly chance-level, and negative values indicate systematic disagreement worse than chance.

Landis and Koch (1977) proposed widely used (and equally widely criticised) benchmarks: 0.01–0.20 slight, 0.21–0.40 fair, 0.41–0.60 moderate, 0.61–0.80 substantial, and 0.81–1.00 almost perfect. These thresholds are descriptive heuristics, not strict cut-offs.

14.10.4 Assumptions

There are exactly two raters, the categorical scale has mutually exclusive and exhaustive categories, ratings between the two raters are independent (one did not see or influence the other), and both raters classify the same set of subjects.

14.10.5 R Implementation

```
library(psych)

set.seed(2026)
n <- 100
rater1 <- factor(sample(c("pos", "neg"), n, replace = TRUE))
agree <- rbinom(n, 1, 0.8)
rater2 <- ifelse(agree == 1, as.character(rater1),
                 ifelse(rater1 == "pos", "neg", "pos"))
rater2 <- factor(rater2, levels = levels(rater1))

tab <- table(rater1, rater2)
tab

cohen.kappa(cbind(rater1, rater2))
```

14.10.6 Output & Results

`cohen.kappa()` returns the unweighted (and weighted, where applicable) kappa statistic, its standard error, and a confidence interval. Reporting the contingency table alongside the kappa value gives readers the raw evidence; a small or imbalanced contingency table can produce surprising kappa behaviour and the table is the only diagnostic that reveals it.

14.10.7 Interpretation

A reporting sentence: "Inter-rater agreement on the binary classification was substantial (Cohen's $\kappa = 0.58$, 95 % CI 0.41 to 0.75) per the Landis-Koch benchmarks, with observed agreement 80 % and chance-expected agreement 52 %. The contingency table showed approximately balanced marginals (rater 1: 51 % positive; rater 2: 53 % positive), so the kappa-paradox concern that affects skewed-marginal samples does not apply here." Always report observed agreement, marginals, and the kappa value together.

14.10.8 Practical Tips

- Kappa depends on the prevalence and balance of the categories — the well-known "kappa paradox": very low-prevalence categories can produce small kappa values even when observed agreement is high, because the chance-expected agreement is also high. Always report kappa alongside the observed agreement and the marginals so readers can diagnose this.
- For nominal scales with more than two categories, the unweighted kappa treats every disagreement equally; for ordinal scales (mild / moderate / severe) use the weighted kappa to credit partial agreement, with quadratic weights as the conventional default.
- For more than two raters, use Fleiss's kappa (a generalisation of Cohen's kappa) or, when the rating is on an interval-like scale, the intraclass correlation coefficient (ICC); these handle multi-rater designs that Cohen's kappa cannot.
- Confidence intervals on kappa via the delta method are routinely reported by `psych::cohen.kappa()`; bootstrap CIs are preferable for small samples or when the marginal distributions are very imbalanced.
- Distinguish inter-rater reliability (different raters on the same subjects) from intra-rater reliability (the same rater on different occasions); both can be assessed by kappa, but the design and inferential implications differ.
- For continuous outcomes use a Bland-Altman analysis or the ICC; kappa is appropriate only for categorical scales and is misleading when applied to continuous data after dichotomisation.

14.10.9 R Packages Used

`psych::cohen.kappa()` for the canonical Cohen's and weighted kappa with confidence intervals; `irr::kappa2()` and `irr::kappam.fleiss()` for an alternative

interface and multi-rater extensions; `vcd::Kappa()` for kappa within the `vcd` contingency-table ecosystem; `epibasix` for kappa with epidemiological reporting; `DescTools::CohenKappa()` for fast computation alongside related descriptive statistics.

14.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.10.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.11 Conditional Power

14.11.1 Introduction

Conditional power is the probability of rejecting the null at the final analysis given the observed interim data and an assumed treatment effect for the remainder. It is the standard tool for futility stopping: a low conditional power indicates the trial is unlikely to succeed even with favourable future data.

14.11.2 Prerequisites

Group-sequential designs; interim analyses; power calculations.

14.11.3 Theory

For a two-sided Z-test at fraction t of information, conditional power under assumption θ is

$$\mathrm{CP}(\theta) = 1 - \Phi\left(\frac{z_{1-\alpha}\sqrt{1} - \sqrt{t}\,Z_t - (1-t)\theta/\sqrt{V}}{\sqrt{1-t}}\right)$$

where Z_t is the observed interim statistic and V is the information.

Typical futility trigger: stop if CP(assumed effect) < 20 %.

14.11.4 Assumptions

Assumed effect for the remainder of the trial is appropriate (observed, target, or conservative); test is Z-like.

14.11.5 R Implementation

```r
library(rpact)

design <- getDesignGroupSequential(
  sided = 1, alpha = 0.025, beta = 0.2,
  typeOfDesign = "OF",
  informationRates = c(0.5, 1)
)

# Interim analysis results: observed Z = 0.4 (weak signal)
results <- getDataSet(n1 = c(50, 50), means1 = c(0.1, NA),
                      stDevs1 = c(1, NA), n2 = c(50, 50),
                      means2 = c(0, NA), stDevs2 = c(1, NA))

ana <- getAnalysisResults(design = design, dataInput = results,
                          thetaH0 = 0, stage = 1)

# Conditional power at planned effect (delta = 0.3) and observed trend
cond <- getConditionalPower(ana,
                            nPlanned = c(50, 50),
                            thetaH1 = c(0.1, 0.3))
cond
```

14.11.6 Output & Results

Conditional power at two assumed future effects; if CP at the planned effect is low (say < 20 %), a DMC might recommend futility stopping.

14.11.7 Interpretation

"Interim conditional power under the originally planned effect was 0.18; under the observed interim trend, 0.11. The DMC recommended futility stopping at the pre-specified < 20 % threshold."

14.11.8 Practical Tips

- Always pre-specify the futility threshold and assumed effect in the SAP.
- CP under the **observed effect** is the "optimistic" view; CP under zero effect is the conservative view.

- **Predictive power** (Bayesian analogue) averages CP over a posterior for the effect – often preferred in modern trials.
- Futility boundaries are typically **non-binding** (can be overridden by DMC) to preserve alpha.
- CP-based futility can save substantial cost in otherwise failing trials.

14.11.9 Reporting

A clear conditional-power report distinguishes the assumed effect from the observed interim effect, and presents both anchors so reviewers can judge the futility decision against the original design intent and against the trial's actual interim trajectory. Quote the threshold prospectively recorded in the statistical analysis plan and state whether crossing it triggered a binding stop or only a recommendation that the data monitoring committee could override. Where Bayesian predictive power was computed, report the prior used for the effect and explain why that prior was deemed plausible at the design stage, since the futility decision is only as defensible as the assumptions feeding it.

14.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.11.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.12 Cutpoint Selection

14.12.1 Introduction

Once a continuous diagnostic test's discrimination is established, a clinical cutpoint must be chosen to turn continuous scores into binary decisions. The optimal cutpoint depends on the trade-off between sensitivity and specificity and on relative costs of false positives vs false negatives.

14.12.2 Prerequisites

Sensitivity and specificity; ROC curves.

14.12.3 Theory

Common criteria: - **Youden's J**: maximise Sens + Spec − 1. Implicitly assumes equal cost of FN and FP. - **Closest to (0, 1)**: minimise $\sqrt{(1-\text{Sens})^2 + (1-\text{Spec})^2}$. - **Cost-weighted**: minimise $c_{FN}(1-\text{Sens}) \cdot p_D + c_{FP}(1-\text{Spec}) \cdot (1-p_D)$ where p_D is prevalence. - **Target specificity (or sensitivity)**: fix one and maximise the other.

14.12.4 Assumptions

Target population has a known prevalence; costs of misclassification are elicited or set by convention; cutpoint will generalise to a new sample.

14.12.5 R Implementation

```
library(pROC); library(cutpointr)

set.seed(2026)
n <- 300
disease <- factor(sample(c(0, 1), n, replace = TRUE, prob = c(0.6, 0.4)))
score   <- rnorm(n, mean = ifelse(disease == 1, 1.2, 0))

roc_obj <- roc(response = disease, predictor = score,
               levels = c("0", "1"), direction = "<")
youden_thr <- coords(roc_obj, "best", best.method = "youden",
                     transpose = FALSE)
youden_thr

# cutpointr package: multiple criteria in one call
cp <- cutpointr(data.frame(score, disease),
                x = score, class = disease,
                method = maximize_metric, metric = youden)
summary(cp)
```

14.12.6 Output & Results

Cutpoint at Youden's J and associated sensitivity/specificity. Common cutpoint in this simulation: ~0.6 with sensitivity ~0.7 and specificity ~0.7.

14.12.7 Interpretation

"Maximum Youden's J was achieved at a cutpoint of 0.63 (sensitivity 0.72, specificity 0.71). In a population with 40 % prevalence, this yields PPV 0.62 and NPV 0.80."

14.12.8 Practical Tips

- Internal cutpoints overfit the training data; cross-validate or use a separate holdout set.
- Clinical cutpoints should be stable, rounded to a meaningful precision, and validated prospectively.
- For screening tests, favour sensitivity; for confirmatory tests, favour specificity.
- Report both the cutpoint and its downstream metrics (Sens, Spec, PPV, NPV, LR+, LR-).
- Decision-curve analysis (Vickers-Elkin) incorporates clinical utility across a range of threshold probabilities.

14.12.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.12.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.13 Diagnostic Test Accuracy

14.13.1 Introduction

When a new diagnostic test is proposed, the first question is: how accurately does it classify disease status against a reference standard? The answer comes in several complementary numbers – sensitivity, specificity, positive and negative predictive values, likelihood ratios, and the area under the ROC curve. Each answers a different question, and reporting any one alone is inadequate.

14.13.2 Prerequisites

The reader should understand the distinction between a test result (positive/negative) and disease status (truly positive/negative), and should be comfortable with 2x2 tables and proportions.

14.13.3 Theory

Given a binary test and a binary gold-standard diagnosis, every study produces a 2x2 table:

	Disease +	Disease -
Test +	TP	FP
Test -	FN	TN

From this table:

- **Sensitivity** = TP / (TP + FN). The probability that the test is positive in a diseased person.
- **Specificity** = TN / (TN + FP). The probability that the test is negative in a healthy person.
- **Positive predictive value (PPV)** = TP / (TP + FP). The probability of disease given a positive test.
- **Negative predictive value (NPV)** = TN / (TN + FN). The probability of no disease given a negative test.
- **Positive likelihood ratio (LR+)** = sensitivity / (1 - specificity). How many times more likely a positive test is in diseased versus healthy people.
- **Negative likelihood ratio (LR-)** = (1 - sensitivity) / specificity.

Sensitivity and specificity are properties of the test that are (approximately) invariant to prevalence. Predictive values depend strongly on prevalence: a test with 99% sensitivity and 99% specificity still has PPV below 50% when disease prevalence is 1%. Likelihood ratios, via Bayes' theorem, convert a pre-test probability into a post-test probability and thus tie the test-level quantities to the clinical reasoning a doctor actually does.

For a continuous marker, each possible threshold produces a pair (sensitivity, 1 - specificity). Plotting these across all thresholds gives the **receiver operating characteristic (ROC) curve**. The **area under the ROC curve (AUC)** is the probability that a randomly chosen diseased person has a higher marker value than a randomly chosen healthy person – an interpretable summary of discrimination independent of any threshold.

14.13.4 Assumptions

- The reference standard is a true gold standard (otherwise sensitivity and specificity are biased).
- The test is evaluated on a representative spectrum of diseased and healthy individuals (spectrum bias can inflate apparent accuracy).
- Test results are read blinded to the reference standard (verification and review bias are the two most common threats in reporting).

14.13.5 R Implementation

```r
library(pROC)
library(cutpointr)

set.seed(2026)
n <- 200
disease <- rbinom(n, 1, 0.3)
marker <- ifelse(disease == 1,
                 rnorm(n, mean = 60, sd = 12),
                 rnorm(n, mean = 45, sd = 10))

df <- data.frame(disease = factor(disease, levels = c(0, 1),
                                  labels = c("healthy", "diseased")),
                 marker = marker)

roc_obj <- roc(df$disease, df$marker,
               levels = c("healthy", "diseased"), direction = "<")

auc(roc_obj)
ci.auc(roc_obj)

plot(roc_obj, print.auc = TRUE, ci = TRUE)

cp <- cutpointr(df, marker, disease,
                pos_class = "diseased",
                method = maximize_metric,
                metric = youden)
summary(cp)
plot(cp)
```

`pROC::roc()` constructs the ROC object from marker values and disease labels. `auc()` and `ci.auc()` give the point estimate and 95% CI. The `cutpointr` package finds the threshold that maximises Youden's index (sensitivity + specificity - 1) and reports the operating characteristics at that threshold.

14.13.6 Output & Results

The simulated example gives an AUC of approximately 0.85 (95% CI 0.79 to 0.90). The Youden-optimal cutoff is around 52, with sensitivity ~0.75 and specificity ~0.80 at that threshold.

14.13.7 Interpretation

A manuscript table should report sensitivity, specificity, PPV, NPV, LR+, LR-, and the AUC, each with 95% confidence intervals. For a binary test:

"Sensitivity was 75% (95% CI 66-83%), specificity 80% (74-86%), PPV 62% (52-72%), NPV 88% (82-93%), LR+ 3.8 (2.7-5.2), LR- 0.31 (0.22-0.44), AUC 0.85 (0.79-0.90)."

Crucially, the PPV depends on the disease prevalence in the study population. If the intended clinical use is in a lower-prevalence setting, report the projected PPV at that prevalence using Bayes' theorem.

14.13.8 Practical Tips

- Always report the reference standard explicitly and justify it as a gold standard.
- Report sensitivity and specificity *with* predictive values, not instead of them. Predictive values are what the clinician uses; sensitivity and specificity are what the test provides.
- Use 95% Wilson or Clopper-Pearson intervals for proportions, not the Wald interval, which can extend outside $[0, 1]$ or have poor coverage near 0 and 1.
- Avoid choosing a threshold from the same data that will report its performance; hold out a validation set or use cross-validation.
- Follow STARD reporting guidelines: flow diagram, blinding, reference-standard description, thresholds, and indeterminate results.

14.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.13.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.14 Clinical Equivalence Trials

14.14.1 Introduction

Clinical equivalence trials test whether two treatments are clinically interchangeable within a pre-specified equivalence margin — narrow enough that any difference inside the margin is regarded as clinically meaningless. Bioequivalence studies, used to support generic-drug approval, are the archetypal application: reg-

ulators require that the ratio of pharmacokinetic parameters between a generic and the reference product fall within 80 % to 125 % of unity, demonstrating that the generic delivers essentially the same exposure as the originator. Equivalence frameworks also apply in clinical contexts where two interventions are competing on safety or convenience and an investigator wishes to show "no clinically meaningful difference" rather than the more usual "test product is better".

14.14.2 Prerequisites

A working understanding of non-inferiority trial design, confidence-interval logic, and the distinction between absence of evidence (high p-value) and evidence of absence (CI within an equivalence margin).

14.14.3 Theory

The two one-sided tests (TOST) procedure rejects the null of non-equivalence if and only if the two-sided 90 % confidence interval of the treatment effect lies entirely within the equivalence margin $[-\Delta, +\Delta]$. This is equivalent to two one-sided tests, each at $\alpha = 0.05$, against the lower and upper non-equivalence boundaries; the overall type-I error is preserved at 0.05 because at most one of the two boundaries can be violated under any single state of the world.

For bioequivalence on log-transformed pharmacokinetic parameters such as AUC and $C_{\max}$, the ratio μ_T/μ_R must lie within $(0.80, 1.25)$, corresponding to $\pm\log(1.25) \approx \pm 0.223$ on the natural-log scale. Log-transformation is mandated by regulators because it makes the test-to-reference ratio symmetric around unity and renders the inference Normal-theory tractable.

14.14.4 Assumptions

The outcome (typically a log-transformed pharmacokinetic parameter) is approximately Normal, the design is a crossover with adequate washout to eliminate carryover, the within-subject variance is reasonably estimated, and observations are independent across subjects.

14.14.5 R Implementation

```
library(PowerTOST)

n <- sampleN.TOST(alpha = 0.05, targetpower = 0.80,
                  theta0 = 0.95,
                  theta1 = 0.80, theta2 = 1.25,
                  CV = 0.20,
                  design = "2x2")
n
```

```r
set.seed(2026)
n_sub <- 30
subject_re <- rnorm(n_sub, 0, 0.15)
period1 <- exp(subject_re + rnorm(n_sub, 0, 0.1))
period2 <- exp(subject_re + log(0.95) + rnorm(n_sub, 0, 0.1))

log_diff <- log(period1) - log(period2)
m  <- mean(log_diff); sd_d <- sd(log_diff)
ci <- m + c(-1, 1) * qt(0.95, df = n_sub - 1) * sd_d / sqrt(n_sub)
exp(ci)
```

14.14.6 Output & Results

`sampleN.TOST()` returns the sample size required to achieve target power for the bioequivalence hypothesis under specified true ratio and within-subject coefficient of variation. The simulation block then computes a 90 % confidence interval on the test-to-reference ratio, which is the regulatory-relevant inference. Reporting both the point ratio and the CI is the standard expected by FDA and EMA.

14.14.7 Interpretation

A reporting sentence: "The 90 % confidence interval of the test-to-reference ratio was 0.92 to 1.08, fully within the regulatory bioequivalence window of 0.80 to 1.25 for both AUC and $C_{\max}$. The geometric mean ratio was 1.00, with within-subject coefficient of variation 18 %. Bioequivalence was therefore demonstrated under standard FDA and EMA criteria; the formal TOST procedure rejected non-equivalence on both boundaries at $\alpha = 0.05$." Always report the 90 % CI on the back-transformed scale.

14.14.8 Practical Tips

- Always analyse log-transformed PK parameters rather than raw values; log-ratios are symmetric around unity, the regulatory equivalence window translates to a symmetric range on the log scale, and Normal-theory inference is tractable on the log scale.
- Use the 90 % confidence interval, not the 95 %; the TOST procedure at $\alpha = 0.05$ corresponds exactly to a two-sided 90 % CI lying entirely within the equivalence margin, and this is the regulatory standard.
- The bioequivalence margin (0.80 to 1.25) is fixed by FDA and EMA regulation; clinical equivalence margins for non-PK outcomes must be prespecified and justified clinically, because the equivalence claim hinges entirely on the margin width.
- Reference-scaled bioequivalence (RSABE) is used for highly variable drugs with within-subject CV above 30 %; the equivalence margin is then

widened proportionally to the reference-product variability, preserving statistical feasibility for inherently variable products.
- Replicate crossover designs (each subject receives test and reference twice) reduce within-subject variance and improve efficiency; they are the standard for highly variable drugs and increasingly the default in modern bioequivalence trials.
- Pre-specify the equivalence margin, washout period, and analysis model in the protocol; FDA and EMA scrutinise these choices closely, and post-hoc margin selection is grounds for rejection.

14.14.9 R Packages Used

`PowerTOST` for canonical TOST sample-size calculation, power analysis, and simulation across crossover and parallel-group equivalence designs; `bear` for end-to-end FDA-compliant bioequivalence analysis with all standard reporting; `bioequivalence` and `bioequivalenceR` for alternative interfaces; `replicateBE` for replicate-design bioequivalence with reference-scaled procedures; `BE` for Bayesian bioequivalence approaches.

14.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.14.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.15 Factorial Trials

14.15.1 Introduction

Factorial trials test two or more interventions in a single experiment: participants are randomised to each factor independently, producing 2x2 (or higher) cells. The design is efficient if the interventions act independently (no interaction); otherwise the interaction itself becomes the primary finding.

14.15.2 Prerequisites

ANOVA; interactions.

14.15.3 Theory

In a 2x2 design, four cells: control, A only, B only, A + B. Main effects are estimated by averaging over the other factor; the interaction compares observed A + B effect to the sum of individual effects.

If interaction is negligible, factorial is efficient: same power as two separate trials with half the total sample size.

14.15.4 Assumptions

Treatments do not interact (or the interaction is the inferential target); randomisation is to each factor independently; outcome is measured under identical conditions across cells.

14.15.5 R Implementation

```r
set.seed(2026)
n_per_cell <- 40
A <- factor(rep(c("no", "yes"), each = 2 * n_per_cell))
B <- factor(rep(rep(c("no", "yes"), each = n_per_cell), 2))

# Simulate additive effects, mild positive interaction
y <- rnorm(4 * n_per_cell) +
     ifelse(A == "yes", 0.5, 0) +
     ifelse(B == "yes", 0.3, 0) +
     ifelse(A == "yes" & B == "yes", 0.2, 0)

fit <- lm(y ~ A * B)
summary(fit)$coefficients
anova(fit)
```

14.15.6 Output & Results

Main-effect estimates for A and B plus the interaction term. The interaction is small, consistent with the simulated +0.2.

14.15.7 Interpretation

"The factorial analysis estimated a main effect of A = 0.48, B = 0.31, with a small positive interaction (0.22, p = 0.14). Main-effect analyses are interpretable in the absence of significant interaction."

14.15.8 Practical Tips

- Pre-specify the interaction test and its interpretation; a non-significant test does not guarantee additivity.

- Factorial trials are under-powered for interactions unless specifically designed for them.
- Report both main effects (averaged over the other factor) and cell means.
- Partial factorial ('unbalanced') designs drop problematic cells – useful when certain combinations are unethical or impractical.
- For > 3 factors, fractional factorial designs (Taguchi) reduce cell count at the cost of confounding higher-order interactions.

14.15.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.15.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.16 Subgroup Forest Plots

14.16.1 Introduction

Subgroup forest plots display, on a single figure, the treatment effect estimate and confidence interval for each pre-specified subgroup of a clinical trial alongside the overall effect. They are the standard tool for visually communicating effect modification — whether the intervention works differently in men and women, in older and younger patients, in mild and severe disease — and they are now a near-universal element of CONSORT-compliant trial reporting. The compact horizontal-error-bar layout makes the magnitude and precision of each subgroup-specific effect immediately legible, while a vertical reference line at the null anchors interpretation. The plot's strength is also its risk: readers eye-ball heterogeneity at a glance and may infer effect modification where the formal interaction test does not support it.

14.16.2 Prerequisites

A working understanding of pre-specified subgroup analysis, the difference between within-subgroup tests and the test for interaction, and confidence-interval visualisation.

14.16.3 Theory

Each row of a forest plot shows the subgroup name, sample sizes per arm, the point estimate of the treatment effect (or the within-subgroup analogue), and its 95 % confidence interval as a horizontal whisker. An overall effect — the marginal estimate across the full trial population — appears at the top or bottom of the plot for reference. A vertical line at the null value (0 for differences, 1 for ratios) anchors interpretation, and the formal test for interaction (whether the effect varies across subgroups beyond chance) is reported either in the figure or alongside it.

14.16.4 Assumptions

Subgroups are pre-specified in the trial protocol or statistical analysis plan rather than chosen post hoc, the effect estimates and confidence intervals are correctly computed for each subgroup, and the formal interaction test (rather than within-subgroup p-value comparisons) is the basis for any claim of effect modification.

14.16.5 R Implementation

```
library(ggplot2); library(dplyr)

set.seed(2026)
subgroups <- data.frame(
  group   = c("Overall", "Male", "Female",
              "Age < 65", "Age >= 65",
              "Mild", "Moderate", "Severe"),
  n1      = c(200, 105, 95, 90, 110, 70, 80, 50),
  n2      = c(200, 98, 102, 92, 108, 72, 79, 49),
  effect  = c(0.30, 0.18, 0.43, 0.22, 0.38, 0.12, 0.32, 0.55),
  lower   = c(0.15, -0.02, 0.21, 0.02, 0.18, -0.14, 0.10, 0.28),
  upper   = c(0.45, 0.38, 0.65, 0.42, 0.58, 0.38, 0.54, 0.82)
)

ggplot(subgroups, aes(y = factor(group, levels = rev(group)),
                      x = effect)) +
  geom_vline(xintercept = 0, linetype = 2, colour = "grey50") +
  geom_errorbarh(aes(xmin = lower, xmax = upper), height = 0.15) +
  geom_point(size = 3, colour = "#2A9D8F") +
  labs(x = "Treatment effect (with 95% CI)", y = NULL,
       title = "Subgroup forest plot") +
  theme_minimal() +
  theme(panel.grid.minor = element_blank())
```

14.16.6 Output & Results

The resulting plot is a vertical list of subgroups with horizontal confidence intervals referenced to the null line. Combining this with a column of sample sizes and an interaction-test p-value column produces a publication-ready forest plot. Most clinical-trial reports also include the formal interaction p-value at the right of the figure or in the figure caption.

14.16.7 Interpretation

A reporting sentence: "The forest plot showed the treatment effect was present across all eight pre-specified subgroups; directional heterogeneity was observed by sex (point estimate 0.43 in women vs 0.18 in men) and disease severity (0.55 in severe vs 0.12 in mild patients). Formal tests for interaction were not significant ($p_{\text{sex}} = 0.11$, $p_{\text{severity}} = 0.07$), suggesting the observed within-subgroup differences may reflect sampling rather than genuine effect modification. Subgroups were pre-specified in the SAP." Always report formal interaction tests, not within-subgroup p-values.

14.16.8 Practical Tips

- Always display subgroup sample sizes alongside each row; a small subgroup with a wide CI can look extreme on the plot but carry very little weight in the overall conclusion, and readers benefit from seeing the precision context directly.
- Order subgroups by category (sex, age, disease severity, geographic region) — never by point estimate. Post-hoc ordering is a recurring source of biased visual interpretation and is increasingly flagged by trial reviewers.
- Show the overall trial effect prominently, at the top or bottom of the plot, as the reference against which subgroup deviations are read; subgroup forest plots without an overall reference line are difficult to interpret.
- For odds ratios, hazard ratios, or risk ratios, use a logarithmic scale on the horizontal axis; on the linear scale, ratios appear asymmetric and visual judgements of "large" vs "small" effects become misleading.
- Include the formal interaction p-value in the figure or directly beside it; this discourages the well-known fallacy of comparing within-subgroup p-values, which are always under-powered and routinely yield "significant in one subgroup, not the other" patterns by chance alone.
- Keep the number of subgroups manageable (typically 5–10); too many subgroups crowd the plot and create false-positive risk through multiple comparisons even with correct interaction-test reporting.

14.16.9 R Packages Used

`ggplot2` for custom forest plots with full layout control; `forestplot` for highly customised clinical-trial forest plots with multiple columns and risk-of-bias annotation; `forester` for tidyverse-friendly forest-plot construction with built-in

subgroup-table integration; `metafor::forest()` when the underlying analysis is meta-analytic; `survminer` for survival-specific forest plots when subgroup effects are hazard ratios.

14.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.16.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.17 Intraclass Correlation Coefficient (ICC)

14.17.1 Introduction

The Intraclass Correlation Coefficient (ICC) quantifies inter-rater reliability on continuous measurements. Unlike Pearson's correlation, it penalises systematic rater bias (two raters who differ by a constant still have Pearson = 1 but ICC < 1). Several ICC forms reflect different study designs and questions.

14.17.2 Prerequisites

Variance components; random vs fixed effects.

14.17.3 Theory

Shrout-Fleiss (1979) forms: - **ICC(1, 1)**: one-way random effects; each subject rated by a different random rater. - **ICC(2, 1)**: two-way random effects; **absolute agreement** between raters. - **ICC(3, 1)**: two-way mixed effects; **consistency** (ignores systematic rater bias).

Single rater vs average of k raters: ICC(x, 1) vs ICC(x, k).

14.17.4 Assumptions

Subjects and raters are drawn from appropriate populations; ratings are independent given subject; ICC form matches the intended use.

14.17.5 R Implementation

```r
library(psych)

set.seed(2026)
n <- 30
subject_re <- rnorm(n, 0, 1)

# 3 raters, each with own systematic bias
r1 <- subject_re + rnorm(n, 0, 0.4)
r2 <- subject_re + 0.3 + rnorm(n, 0, 0.4)    # rater 2 higher by 0.3
r3 <- subject_re - 0.2 + rnorm(n, 0, 0.4)

df <- cbind(r1, r2, r3)

icc_res <- ICC(df, lmer = FALSE)
icc_res$results[, c("type", "ICC", "lower bound", "upper bound")]
```

14.17.6 Output & Results

Six ICC forms with 95 % CIs. Agreement ICC (2, 1) is lower than consistency ICC (3, 1) when raters have systematic biases.

14.17.7 Interpretation

"Single-rater absolute-agreement ICC(2, 1) = 0.78 (95 % CI 0.63-0.88); consistency ICC(3, 1) = 0.83. Systematic rater differences moderately lowered absolute agreement."

14.17.8 Practical Tips

- Choose ICC form by study question:
  - **ICC(2, 1)** if you want to quantify agreement including systematic rater differences.
  - **ICC(3, 1)** if you will remove systematic rater effects in practice (e.g., calibrate each rater).
- For clinical usability, average-of-k-raters ICCs are the most interpretable.
- Thresholds (Koo-Li): < 0.5 poor, 0.5-0.75 moderate, 0.75-0.9 good, > 0.9 excellent.
- ICC < 0.7 is usually insufficient for individual decision-making.
- Pair ICC with Bland-Altman for graphical assessment of rater-specific bias.

14.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.17.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.18 Interim Analyses and Group Sequential Designs

14.18.1 Introduction

Group-sequential designs schedule a sequence of interim analyses at pre-specified information fractions, each with efficacy and/or futility boundaries. Trials can stop early for benefit, harm, or futility while preserving overall Type I error. They are the standard for confirmatory trials with ethical imperatives for early stopping.

14.18.2 Prerequisites

Type I error; sample-size calculation; information fraction.

14.18.3 Theory

With K analyses and overall two-sided alpha α, boundaries are chosen so the union of rejection events has probability α under the null.

Common families: - **O'Brien-Fleming**: conservative early, near-nominal late – preserves nominal alpha at final analysis. - **Pocock**: constant boundary across looks – more early stopping but harder to reach final. - **Alpha-spending** (Lan-DeMets): flexible timing; spending function $f(t)$ at information fraction t.

14.18.4 Assumptions

Analyses occur at pre-specified information fractions; test statistic is approximately normal at each look.

14.18.5 R Implementation

```r
library(rpact)

# Group-sequential design: 3 analyses, O'Brien-Fleming, alpha=0.025
 ↪  one-sided
design <- getDesignGroupSequential(
  sided = 1, alpha = 0.025, beta = 0.2,
  informationRates = c(0.33, 0.67, 1),
  typeOfDesign = "OF"
)
print(design)

# Corresponding sample sizes for two-mean comparison
ss <- getSampleSizeMeans(design = design,
                         alternative = 0.3, stDev = 1)
print(ss)
```

14.18.6 Output & Results

Three-stage design with cumulative alpha budgets per stage summing to 0.025; stage sample sizes scale with the chosen information fractions.

14.18.7 Interpretation

"The group-sequential design with O'Brien-Fleming boundary allocated very little alpha to the first two interims (< 0.001 each), preserving nearly the full alpha for the final analysis. Early stopping is extremely unlikely unless the effect is large."

14.18.8 Practical Tips

- OF boundaries are standard for confirmatory trials; Pocock may suit exploratory ones or single-arm phase II.
- Alpha-spending (Lan-DeMets) is more flexible – timing does not need to be exact, only pre-specified.
- Independent DMC must oversee interim analyses; investigators remain blinded.
- Adjust point estimates and CIs for stopping (median-unbiased, repeated CI).
- Futility boundaries (betagamma spending) complement efficacy and can be non-binding.

14.18.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.18.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.19 ITT vs Per-Protocol Analysis

14.19.1 Introduction

The **intent-to-treat (ITT)** principle analyses participants according to their randomised assignment, irrespective of actual treatment received or adherence. It estimates the effect of *prescribing* the intervention under real-world conditions. **Per-protocol (PP)** restricts analysis to adherent participants and estimates the effect of *receiving* the intervention – a different estimand.

14.19.2 Prerequisites

Randomisation; estimands; missing data.

14.19.3 Theory

ITT preserves randomisation and tends to be conservative in superiority trials (dilution by non-adherers). PP can exaggerate efficacy or introduce selection bias because adherence is post-randomisation.

Modified ITT (mITT) excludes participants who never started or have no post-baseline data; common but can reintroduce bias if exclusion correlates with arm.

Non-inferiority trials: PP and ITT are both reported; the more conservative result (less close to margin) drives inference.

14.19.4 Assumptions

Randomisation is properly concealed; adherence classification is unaffected by knowledge of outcome; missingness mechanism is characterised.

14.19.5 R Implementation

```r
set.seed(2026)
n_per <- 100
arm <- rep(c("trt", "ctrl"), each = n_per)

# 20 % non-adherence in trt arm; 5 % in ctrl
adhered <- ifelse(arm == "trt", rbinom(n_per * 2, 1, 0.8),
                  rbinom(n_per * 2, 1, 0.95))
adhered[1:n_per] <- rbinom(n_per, 1, 0.8)
adhered[(n_per+1):(2 * n_per)] <- rbinom(n_per, 1, 0.95)

# True effect when actually received
true_effect <- ifelse(adhered == 1 & arm == "trt", 0.7, 0)
y <- rnorm(2 * n_per) + true_effect

# ITT analysis (analyse as randomised)
itt <- t.test(y ~ arm)
# Per-protocol analysis (adherers only)
pp  <- t.test(y[adhered == 1] ~ arm[adhered == 1])

rbind(ITT = c(est = diff(itt$estimate), p = itt$p.value),
      PP  = c(est = diff(pp $estimate), p = pp $p.value))
```

14.19.6 Output & Results

ITT effect is diluted by non-adherence; PP effect recovers the on-treatment effect but is subject to selection bias.

14.19.7 Interpretation

"The primary ITT analysis estimated a 0.56 point advantage for the intervention (95 % CI 0.28-0.84, $p < 0.001$); PP analysis gave 0.71 (CI 0.38-1.04). ITT is the primary inference; PP is supportive."

14.19.8 Practical Tips

- Pre-specify ITT as primary in the SAP; never switch post-hoc.
- Report a flow diagram (CONSORT) showing how participants were classified.
- Handle missing outcome data with multiple imputation, not naive exclusion.
- **Complier-average causal effect (CACE)** via instrumental variables estimates the effect among adherers without PP's selection bias.
- Non-inferiority trials commonly report both ITT and PP; both must meet the margin.

14.19.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.19.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.20 Likelihood Ratios

14.20.1 Introduction

Likelihood ratios (LR+ and LR-) summarise the information content of a diagnostic test in a way that is independent of disease prevalence. Sensitivity and specificity describe how the test performs in known-disease and known-no-disease populations, but they do not directly tell a clinician what to believe after observing a positive or negative test result; predictive values do, but they depend on prevalence and are therefore not portable across populations. Likelihood ratios bridge this gap: they combine multiplicatively with the pre-test odds to yield the post-test odds via Bayes' theorem, making them the natural building blocks of clinical Bayesian reasoning. Modern evidence-based-medicine guidance and core teaching texts present LRs as the preferred diagnostic-performance summary precisely because of this prevalence-independence.

14.20.2 Prerequisites

A working understanding of sensitivity and specificity, the relationship between probability and odds, and Bayes' theorem in odds form.

14.20.3 Theory

The two basic likelihood ratios are

$$\mathrm{LR}^{+} = \frac{\mathrm{Sens}}{1 - \mathrm{Spec}}, \qquad \mathrm{LR}^{-} = \frac{1 - \mathrm{Sens}}{\mathrm{Spec}}.$$

The Bayes-theorem update is

$$\text{Odds}_{\text{post}} = \text{Odds}_{\text{pre}} \times \text{LR}.$$

Conventional clinical interpretation: $\text{LR}^{+} > 10$ or $\text{LR}^{-} < 0.1$ is often decisive; 5–10 or 0.1–0.2 is moderate evidence; 2–5 or 0.2–0.5 is weak; near 1 is uninformative. Fagan's nomogram graphically converts pre-test probability and LR directly to post-test probability and is a useful bedside tool.

14.20.4 Assumptions

The same assumptions as for sensitivity and specificity: a reliable gold-standard reference for disease status, accurate test classification, and that the test characteristics estimated in one population generalise to the patients to whom the LR is being applied. Verification bias and spectrum bias both threaten this generalisation.

14.20.5 R Implementation

```r
library(epiR)

tab <- as.table(matrix(c(80, 20,
                         20, 880),
                       nrow = 2, byrow = FALSE,
                       dimnames = list(Test = c("+", "-"),
                                       Disease = c("yes", "no"))))
epi.tests(tab)$detail[c("lrpos", "lrneg"), ]

sens <- 0.8; spec <- 0.978
lr_pos <- sens / (1 - spec); lr_neg <- (1 - sens) / spec

prior_odds <- 0.1 / 0.9
post_odds  <- prior_odds * lr_pos
post_prob  <- post_odds / (1 + post_odds)
c(lr_pos = lr_pos, lr_neg = lr_neg, post_prob_after_pos = post_prob)
```

14.20.6 Output & Results

`epi.tests()` reports LR+ and LR- with their confidence intervals from the input contingency table. The Bayesian update example shows how a 10 % pre-test probability rises to roughly 80 % post-test after a positive result with LR+ = 36, illustrating the multiplicative-on-odds nature of the update.

14.20.7 Interpretation

A reporting sentence: "The diagnostic test had sensitivity 80 % (95 % CI 71 to 87 %) and specificity 97.8 % (96.4 to 98.7), corresponding to LR+ = 36 (95 %

CI 16 to 81) and LR- = 0.20 (95 % CI 0.13 to 0.31). A positive test raised the pre-test probability of 10 % to a post-test probability of 80 %, while a negative test reduced it to roughly 2 %. The test is therefore strongly informative in both directions for the typical pre-test probability range of this clinical setting." Always report LRs with CIs.

14.20.8 Practical Tips

- Report likelihood ratios with their 95 % confidence intervals; wide CIs indicate fragile diagnostic estimates and should temper interpretation, especially when small sample sizes or rare disease drive uncertainty in sensitivity or specificity.
- For multi-category or ordinal tests (rating scales, semi-quantitative biomarker results), compute stratum-specific LRs for each score level rather than collapsing to a single binary LR; this preserves the information content of the gradation.
- Likelihood ratios generalise across populations as long as the test characteristics (sensitivity, specificity) hold in the new population; this is their primary advantage over predictive values, which depend on local prevalence and do not transfer.
- Clinical decision thresholds are often pre-specified in terms of required LR (e.g., LR+ $\geq$ 10 to justify initiating treatment, LR- $\leq$ 0.1 to confidently rule out disease); building these thresholds into the diagnostic pathway is the operational analogue of Bayesian reasoning at the bedside.
- Chain multiple test results by multiplying their LRs only if the tests are conditionally independent given disease status; in practice this assumption is often violated (a second test of the same type is correlated with the first), and joint LRs from a multivariable predictor are often more honest.
- For complex multi-variable diagnostic tools (clinical prediction rules), the LR concept generalises naturally — the rule's score corresponds to a stratum-specific LR — and is a useful way to communicate the rule's discrimination at decision-relevant cut-points.

14.20.9 R Packages Used

`epiR::epi.tests()` for canonical sensitivity, specificity, predictive values, and likelihood ratios with confidence intervals from a 2 $\times$ 2 table; `epibasix` for an alternative interface; `caret::confusionMatrix()` for ML-style classification metrics including LRs; `pROC` for LRs across the full ROC operating range; `bayesmeta` and related packages for Bayesian meta-analytic pooling of likelihood ratios across diagnostic-accuracy studies.

14.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.20.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.21 Minimisation Algorithm

14.21.1 Introduction

Minimisation, introduced by Taves (1974) and refined by Pocock and Simon (1975), is a covariate-adaptive allocation procedure that assigns each new participant to the treatment arm that prospectively makes the distribution of prespecified prognostic covariates most balanced across arms. Unlike stratified block randomisation, which requires a separate randomisation list for every stratum and quickly becomes impractical when more than two or three covariates need balancing, minimisation handles many prognostic factors simultaneously in a single allocation framework. It is now widely used in trials with several important baseline covariates and small-to-moderate sample sizes, where stratified randomisation would create too many empty strata to be useful.

14.21.2 Prerequisites

A working understanding of simple, block, and stratified randomisation; balance metrics for cross-tabulated baseline covariates; and the regulatory framework around covariate-adaptive allocation.

14.21.3 Theory

For each candidate arm assignment, the algorithm computes a balance score — typically the sum across covariates of marginal imbalances that would result from that assignment. The new participant is allocated to the balance-minimising arm with high probability p (commonly 0.8 or 0.9), and to the other arm with probability $1 - p$ to preserve a degree of allocation unpredictability. The probabilistic element matters: a deterministic minimisation that always chooses the balance-minimising arm becomes predictable to investigators who know the algorithm and the previous assignments, compromising allocation concealment.

14.21.4 Assumptions

The covariates to be balanced are pre-specified in the protocol, allocation is performed centrally (typically through an interactive web-response system) rather than manually, and the trial's analysis model includes all minimisation covariates as fixed effects to preserve valid inference under randomisation theory.

14.21.5 R Implementation

```
library(Minirand)

set.seed(2026)
n <- 60
covmat <- data.frame(
  centre = sample(c("A", "B", "C"), n, replace = TRUE),
  sex    = sample(c("M", "F"), n, replace = TRUE),
  age_g  = sample(c("young", "old"), n, replace = TRUE)
)

res <- character(n)
for (j in 1:n) {
  res[j] <- Minirand(covmat = covmat, covwt = rep(1, 3),
                     ntrt = 2, trtseq = c("A", "B"),
                     ratio = c(1, 1),
                     p = 0.9, j = j, result = res)
}

table(trt = res, centre = covmat$centre)
table(trt = res, sex = covmat$sex)
```

14.21.6 Output & Results

`Minirand()` allocates each subject sequentially based on the prior allocation history and the new subject's covariate profile. Cross-tabulations of treatment by each covariate show approximately equal arm counts within every covariate level — the design's primary objective — even when no single covariate combination has many subjects.

14.21.7 Interpretation

A reporting sentence: "Treatment allocation used Pocock-Simon minimisation balancing on three pre-specified covariates (centre, sex, age category), each with equal weight; the random component used probability 0.9 of allocation to the balance-minimising arm. The trial's analysis model retained centre, sex, and age category as fixed-effect covariates to preserve valid inference under minimisation; in the final 60-subject sample, the maximum marginal arm imbalance on any

covariate was 1 subject." Always state the random-component probability and the analysis-model covariates.

14.21.8 Practical Tips

- Always analyse the trial with the minimisation covariates as fixed-effect adjustments in the primary analysis model; omitting them violates the randomisation-inference framework that minimisation relies on, and the resulting standard errors are typically too small.
- Pre-specify the covariates, their weights, and the random-component probability p in the protocol and SAP; adding covariates post hoc defeats minimisation's protective role and is disallowed by most regulators.
- Commercial IWRS systems are required for robust minimisation in any non-trivial trial; manual implementation is error-prone, especially as the trial grows, and a single manual error can compromise allocation concealment for the entire study.
- FDA and EMA accept minimisation when the method, covariates, and analysis model are pre-specified; the regulatory concern about covariate-adaptive allocation is largely addressed by transparent documentation and analysis-model adjustment.
- Minimisation is less transparent than stratified block randomisation — investigators cannot reproduce the allocation list from a simple description — so the protocol should describe the algorithm carefully, including the balance metric, weights, and probability parameter.
- For trials with very few prognostic covariates and moderate sample size, stratified block randomisation is often preferable because of its operational simplicity; minimisation's advantage grows with the number of covariates and with smaller per-stratum sample sizes.

14.21.9 R Packages Used

`Minirand` for canonical Pocock-Simon minimisation with arbitrary weights and ratio support; `randomizeR` for an alternative interface integrated with broader randomisation simulation; `bcrm` for biased-coin and minimisation alternatives; `RandomizationLogic` for full IWRS-style simulation including audit trail; `Mediana` for trial-design simulation with minimisation-allocation strategies.

14.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.21.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.22 Missing Data in RCTs

14.22.1 Introduction

Missing data are ubiquitous in RCTs and can invalidate inference if not handled carefully. Three missingness mechanisms characterise the problem: **MCAR** (missingness is random), **MAR** (missingness depends on observed variables), and **MNAR** (missingness depends on the unobserved value). Valid analyses require assumptions on the mechanism.

14.22.2 Prerequisites

Basic probability; estimands framework (ICH E9 R1).

14.22.3 Theory

- **MCAR**: $P(\text{missing}) = P(\text{missing} \mid X, Y)$. Complete-case analysis is unbiased but inefficient.
- **MAR**: $P(\text{missing} \mid X, Y) = P(\text{missing} \mid X)$. Multiple imputation, ML, or weighting is unbiased.
- **MNAR**: $P(\text{missing})$ depends on unobserved Y. Sensitivity analyses with assumed MNAR mechanisms are needed.

Missingness is rarely MCAR in practice; MAR is the default operating assumption, with MNAR as sensitivity.

14.22.4 Assumptions

Missingness pattern is characterised by the analyst; auxiliary variables are included in the imputation model; the mechanism assumption matches the method.

14.22.5 R Implementation

```
library(mice); library(naniar)

set.seed(2026)
n <- 300
baseline <- rnorm(n, 5, 1)
```

```
arm <- rep(c("ctrl", "trt"), each = n/2)
outcome <- baseline + ifelse(arm == "trt", 1, 0) + rnorm(n, 0, 1)

# MAR: missingness depends on baseline
prob_missing <- plogis(-2 + 0.3 * baseline)
outcome[rbinom(n, 1, prob_missing) == 1] <- NA

df <- data.frame(arm = factor(arm), baseline, outcome)

# Missing-data summary
miss_var_summary(df)

# Multiple imputation
imp <- mice(df, m = 10, method = "pmm", printFlag = FALSE)
pool(with(imp, lm(outcome ~ arm + baseline))) %>% summary()
```

14.22.6 Output & Results

Missing-data summary (outcome has ~20 % missingness); pooled MI estimate for treatment effect after adjusting for baseline.

14.22.7 Interpretation

"Under MAR, multiple imputation with 10 imputations gave a treatment effect of 0.94 (95 % CI 0.62-1.26, $p < 0.001$); complete-case analysis gave a similar estimate, consistent with MAR assumption."

14.22.8 Practical Tips

- Prevent missing data by design (pre-specified follow-up, low attrition) before analysis tricks.
- Distinguish missing at random from missing-completely-at-random; CCA needs the stronger MCAR.
- Always report the missingness rate and pattern by arm; differential missingness is a red flag.
- Pre-specify primary analysis under MAR; sensitivity analyses under MNAR (tipping-point).
- ICH E9 R1 estimands framework formalises how missing data interacts with the estimand; align analysis to the estimand, not vice versa.

14.22.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.22.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.23 Multiple Imputation

14.23.1 Introduction

Multiple imputation (MI; Rubin 1987) replaces each missing value with $m > 1$ plausible values drawn from the posterior predictive distribution of the missing data given the observed. The analysis is run on each of the m completed datasets; Rubin's rules combine the results into a single inference reflecting both within-imputation and between-imputation uncertainty.

14.23.2 Prerequisites

Missing-data mechanisms (MAR); Bayesian predictive distributions.

14.23.3 Theory

MI procedure: 1. Impute: create m completed datasets via a predictive model. 2. Analyse: apply the intended model to each dataset. 3. Pool: combine $\bar{\hat{\beta}} = (1/m)\sum \hat{\beta}_k$; total variance $T = \bar{W} + (1 + 1/m)B$, where $\bar{W}$ is mean within-imputation variance and B is between-imputation variance.

`mice` uses chained equations: iteratively impute each variable using the others as predictors.

14.23.4 Assumptions

Missing at random (MAR); imputation model is correctly specified (correct functional form, includes all predictors that might correlate with missingness).

14.23.5 R Implementation

```
library(mice)

set.seed(2026)
n <- 300
```

```
df <- data.frame(
  x1 = rnorm(n), x2 = rnorm(n),
  x3 = sample(c("a", "b", "c"), n, replace = TRUE),
  y  = rnorm(n)
)
df$y[sample(n, 60)] <- NA       # 20% missing
df$x2[sample(n, 30)] <- NA      # 10% missing

# Chained-equations imputation, 20 imputations
imp <- mice(df, m = 20, method = c("pmm", "pmm", "polyreg", "pmm"),
            printFlag = FALSE)

# Fit the model on each imputation and pool
fit <- with(imp, lm(y ~ x1 + x2 + x3))
pooled <- pool(fit)
summary(pooled)
```

14.23.6 Output & Results

Pooled regression coefficients with SEs that correctly reflect imputation uncertainty; `fmi` (fraction of missing information) indicates how much of the variance comes from imputation.

14.23.7 Interpretation

"Multiple imputation (m = 20) under MAR gave a pooled coefficient of 0.94 (SE 0.10, 95 % CI 0.74-1.14); fraction of missing information 0.18 suggests reasonable efficiency."

14.23.8 Practical Tips

- Include **all** variables used in the substantive model, plus auxiliary variables correlated with missingness, in the imputation model.
- m should be ≥ 100 when a large fraction is missing; 20 is a minimum for exploratory work.
- For regression on interactions or non-linearity, include those terms in the imputation model too (impute so the model matches).
- Use predictive mean matching (pmm) for continuous variables to avoid unrealistic extrapolations.
- Always check convergence of chained equations via trace plots.

14.23.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.23.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.24 Non-Inferiority Margin Selection

14.24.1 Introduction

Non-inferiority (NI) trials test whether a new treatment is not worse than an active comparator by more than a pre-specified margin Δ. Margin selection is the most consequential and scrutinised part of NI trial design: too loose and a genuinely inferior treatment is approved; too tight and the study is infeasible.

14.24.2 Prerequisites

Hypothesis testing; active-controlled trials; historical-control data.

14.24.3 Theory

Two common approaches: - **Fixed-margin (synthesis) method**: margin Δ is chosen based on the historical effect of the active comparator vs placebo, typically preserving 50-75 % of that effect. Example: if active reduces mortality by 10 % vs placebo, Δ might be set at 5 %. - **Clinical margin**: a clinically judged threshold of practical importance, independent of historical data.

Both require regulatory justification; FDA and EMA typically require the synthesis approach with supporting clinical judgement.

14.24.4 Assumptions

Historical active-vs-placebo effect is consistent and generalisable to the current trial population; assay sensitivity (ability to detect a difference if truly present) is preserved.

14.24.5 R Implementation

```
# Synthesis-method margin calculation
# Historical effect: placebo-controlled active gives risk reduction 10%
 ↪  (95% CI 7%-13%)
```

```r
# Conservative estimate: lower bound 7%
preservation <- 0.5                    # preserve at least 50% of effect
margin_synthesis <- 0.07 * (1 - preservation)
cat("Synthesis-based NI margin:", margin_synthesis, "\n")

# Sample size for an NI trial with continuous outcome
# Expected true difference 0; margin 0.25 SD; alpha=0.025; power=0.80
library(pwr)
pwr.t.test(d = 0.25, sig.level = 0.025, power = 0.80,
           type = "two.sample", alternative = "greater")
```

14.24.6 Output & Results

Synthesis-based margin (0.035 = 3.5 % preserving 50 % of historical effect); sample-size calculation gives ~252/arm for a 0.25 SD margin.

14.24.7 Interpretation

"The non-inferiority margin was prospectively set at 3.5 percentage points, preserving at least 50 % of the historical benefit of the active comparator (7 % lower bound of historical effect). Sample size was 504 based on 80 % power to rule out a margin with one-sided alpha 0.025."

14.24.8 Practical Tips

- Margin selection must be pre-specified, regulator-reviewed, and clinically justified.
- Run both ITT and per-protocol analyses; NI requires consistency.
- Beware "biocreep": repeated non-inferiority approvals without placebo anchoring drift from effective therapy.
- Assay sensitivity is hard to demonstrate without a placebo arm; three-arm trials (NI + placebo) are ideal but often unethical.
- Non-inferiority claims should report the effect estimate and CI, not just "p $<$ 0.025".

14.24.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.24.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.25 O'Brien-Fleming Boundary

14.25.1 Introduction

The O'Brien-Fleming (OF) boundary, introduced by Peter O'Brien and Thomas Fleming in 1979, is the most widely used efficacy stopping boundary for group-sequential clinical trials and the de facto regulatory default for confirmatory phase-3 trials. It is conservative early and liberal late: very little type-I error budget is spent at early interim analyses, so stopping early for efficacy requires an unusually large effect, while the final analysis uses nearly the full nominal alpha. The practical consequence is that interim stops are rare and convincing — they happen only when the treatment effect is much larger than the powered alternative — while the final analysis suffers only a tiny multiplicity penalty if no early stop occurs.

14.25.2 Prerequisites

A working understanding of group-sequential trial design, the alpha-spending framework, and the trade-off between early-stopping ease and final-analysis stringency in repeated-look hypothesis testing.

14.25.3 Theory

The O'Brien-Fleming boundary on the standardised Wald-statistic scale takes the form $c_k = c/\sqrt{t_k}$ at information fraction t_k, so the nominal p-value threshold required at interim k shrinks with $1/\sqrt{t_k}$. In practice, a five-look equally-spaced OF design has nominal α values approximately 5×10^{-6}, 0.0013, 0.008, 0.018, 0.041 at the five sequential analyses (for one-sided $\alpha = 0.025$). The final analysis therefore uses 0.041 instead of 0.025 — a mild penalty for the option to stop early — while early looks are protected against premature rejection.

14.25.4 Assumptions

Information accrues as planned (regular interim spacing or alpha-spending implementation that handles irregular timing), the test statistic is approximately Normal at each look, and the multiplicity correction is pre-specified before any data are unblinded.

14.25.5 R Implementation

```r
library(rpact)

design <- getDesignGroupSequential(
  sided = 1, alpha = 0.025, beta = 0.2,
  typeOfDesign = "OF",
  informationRates = seq(0.2, 1, by = 0.2)
)

print(design$stageLevels)

print(design$criticalValues)

plot(design, type = 1)
```

14.25.6 Output & Results

`rpact` returns the stage-specific nominal alphas (very conservative at early looks, near-nominal at the final analysis) and the corresponding critical values on the standardised test-statistic scale. The boundary plot makes the asymmetric "high early, low late" shape visually obvious and is a standard supplementary figure in trial design documents.

14.25.7 Interpretation

A reporting sentence: "The five-stage group-sequential design with O'Brien-Fleming efficacy boundaries required nominal $p < 5 \times 10^{-6}$ at the 20 % information interim, relaxing to $p < 0.041$ at the final analysis to maintain overall one-sided $\alpha = 0.025$. Early stopping was therefore triggered only by treatment effects substantially larger than the powered alternative; the final analysis suffered only a 0.009 multiplicity penalty (0.041 vs unadjusted 0.025) if no earlier stop occurred. The maximum sample size was 6 % larger than a fixed-design equivalent." Always justify boundary choice.

14.25.8 Practical Tips

- O'Brien-Fleming is the default efficacy boundary in confirmatory phase-3 trials and is virtually always the regulatory expectation; deviations should be justified in the protocol with explicit reasoning about why a different boundary (Pocock, Hwang-Shih-DeCani, custom) is preferred.
- Pair O'Brien-Fleming efficacy boundaries with a non-binding futility boundary (gamma-spending or beta-spending) to detect hopeless trials early; this combination preserves the type-I error of the efficacy analysis while allowing the trial to stop for futility when the conditional power is low.

- Alpha-spending implementations of O'Brien-Fleming (Lan-DeMets with the OF-shape spending function) preserve the OF behaviour under irregular interim timing and are the modern default for handling unscheduled looks.
- Post-stopping treatment-effect point estimates are upwardly biased — the trial stopped early precisely because the random fluctuation of the effect was large. Report repeated confidence intervals (Jennison-Turnbull) or median-unbiased estimates rather than the naive maximum-likelihood estimate when stopping for efficacy.
- Compared with the Pocock boundary, OF makes early stopping substantially harder but preserves near-nominal final-analysis alpha and a smaller maximum sample size; Pocock makes early stopping easier but at higher final-analysis cost. Choice should reflect the trial's ethical and practical priorities.
- Stage-specific information fractions need not be equally spaced; alpha-spending OF accommodates whatever schedule the data-monitoring committee prefers, but the schedule should be pre-specified or generated from a pre-specified spending function.

14.25.9 R Packages Used

`rpact` for canonical group-sequential design with O'Brien-Fleming, Pocock, Hwang-Shih-DeCani, and custom alpha-spending boundaries; `gsDesign` for an alternative comprehensive group-sequential framework; `ldbounds` for Lan-DeMets alpha-spending; `gsbDesign` for Bayesian group-sequential alternatives; `Mediana` for trial-design simulation including OF and other boundary comparisons.

14.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.25.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.26 Pocock Boundary

14.26.1 Introduction

The Pocock boundary, introduced by Stuart Pocock in 1977, was the first systematic group-sequential efficacy stopping rule for clinical trials. It uses the same nominal type-I error level at every interim and final analysis, distributing the type-I error budget approximately uniformly across looks. The result is a design that makes early stopping for efficacy comparatively easy — a constant relatively-low nominal threshold across all looks — at the cost of requiring an unusually stringent nominal p-value at the final analysis if no earlier look has stopped the trial. Pocock boundaries are conceptually simple, ethically attractive when early stopping is a priority, but somewhat costly in terms of maximum sample size compared with the more conservative O'Brien-Fleming alternative.

14.26.2 Prerequisites

A working understanding of group-sequential trial designs, the alpha-spending framework, and the trade-off between early-stopping ease and final-analysis stringency in repeated-look hypothesis testing.

14.26.3 Theory

With K planned analyses and overall two-sided type-I error α, the Pocock boundary uses a constant nominal alpha α^* at every look such that the probability of any rejection event under the null exactly equals α. For $K = 5$ and $\alpha = 0.05$ (two-sided), $\alpha^* \approx 0.0158$ at every look — substantially below the unadjusted 0.05 because of the multiplicity correction. Compared with O'Brien-Fleming, Pocock rejects more easily at early interim analyses (where O'Brien-Fleming demands extreme test statistics) but less easily at the final analysis (where O'Brien-Fleming approaches the unadjusted threshold).

14.26.4 Assumptions

Information accrues as planned (regular interim spacing or alpha-spending implementation that handles irregular timing), the test statistic is approximately Normal at each look, and the multiplicity correction is pre-specified before any data are unblinded.

14.26.5 R Implementation

```
library(rpact)

design <- getDesignGroupSequential(
  sided = 1, alpha = 0.025, beta = 0.2,
  typeOfDesign = "P",
```

```
  informationRates = seq(0.2, 1, by = 0.2)
)

print(design$stageLevels)
print(design$criticalValues)

plot(design, type = 1)

ss_pocock <- getSampleSizeMeans(design = design,
                                alternative = 0.3, stDev = 1)

design_of <- getDesignGroupSequential(
  sided = 1, alpha = 0.025, beta = 0.2,
  typeOfDesign = "OF",
  informationRates = seq(0.2, 1, by = 0.2)
)
ss_of <- getSampleSizeMeans(design = design_of,
                            alternative = 0.3, stDev = 1)

c(pocock_max_n = ss_pocock$maxNumberOfSubjects,
  of_max_n    = ss_of   $maxNumberOfSubjects)
```

14.26.6 Output & Results

`rpact` returns the constant nominal alpha at each stage, the corresponding critical values on the test-statistic scale, and the maximum sample size needed to maintain the targeted overall power. Comparing Pocock and O'Brien-Fleming sample-size calculations side by side quantifies the cost of the more aggressive early-stopping property.

14.26.7 Interpretation

A reporting sentence: "The five-stage group-sequential design with Pocock boundaries required nominal $p < 0.0158$ at every interim and final analysis to maintain overall one-sided $\alpha = 0.025$. This design enables earlier efficacy stopping than the equivalent O'Brien-Fleming design, but requires approximately 15 % more maximum sample size if no early stop is triggered. The choice was justified by the ethical imperative to halt enrolment as soon as a clinically meaningful benefit is established, given the trial's seriously-ill population." Always justify the boundary choice ethically.

14.26.8 Practical Tips

- Pocock boundaries favour early stopping for efficacy; O'Brien-Fleming favours late stopping with high final-analysis power. The choice should

reflect whether the trial prioritises ethical termination of clearly beneficial interventions (Pocock) or efficient final-analysis confirmation (O'Brien-Fleming).
- The maximum sample size is larger under Pocock than under O'Brien-Fleming for the same overall power; O'Brien-Fleming is usually preferred when a meaningful effect is expected only at the final analysis and ethics permit waiting.
- Alpha-spending implementations (Lan-DeMets with the Pocock-shape spending function αt or related linear forms) approximate Pocock behaviour under irregular interim timing and are the modern default in regulatory submissions.
- Pocock boundaries are now rarely the primary choice in confirmatory phase-3 trials; O'Brien-Fleming has become the regulatory default. Pocock remains useful in phase-2 trials and futility-stopping contexts where early termination is a stronger priority.
- A hybrid Hwang-Shih-DeCani family interpolates between Pocock and O'Brien-Fleming behaviour via a single shape parameter γ, allowing trial designers to dial the trade-off explicitly.
- Pre-specify the boundary type, the spending function, and any contingency for unscheduled interims in the protocol; ad hoc adjustments after data inspection inflate type-I error and are increasingly flagged by regulators.

14.26.9 R Packages Used

`rpact` for canonical group-sequential design with Pocock, O'Brien-Fleming, Hwang-Shih-DeCani, and custom alpha-spending boundaries; `gsDesign` for an alternative comprehensive group-sequential framework; `ldbounds` for Lan-DeMets alpha-spending implementation; `gsbDesign` for Bayesian group-sequential alternatives; `Mediana` for trial-design simulation including Pocock and O'Brien-Fleming boundary comparisons.

14.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.26.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.27 Predictive Values and Prevalence

14.27.1 Introduction

Sensitivity and specificity are properties of a diagnostic test, not its clinical utility. **Positive predictive value (PPV)** – the probability that a positive test indicates disease – and **negative predictive value (NPV)** depend on disease prevalence. The same test can have very high PPV in a high-prevalence setting and very low PPV in a screening setting.

14.27.2 Prerequisites

Sensitivity and specificity; Bayes' theorem.

14.27.3 Theory

$$\mathrm{PPV} = \frac{\mathrm{Sens} \cdot \mathrm{Prev}}{\mathrm{Sens} \cdot \mathrm{Prev} + (1 - \mathrm{Spec}) \cdot (1 - \mathrm{Prev})}.$$

$$\mathrm{NPV} = \frac{\mathrm{Spec} \cdot (1 - \mathrm{Prev})}{(1 - \mathrm{Sens}) \cdot \mathrm{Prev} + \mathrm{Spec} \cdot (1 - \mathrm{Prev})}.$$

For a fixed test, PPV rises with prevalence and NPV falls. A high-specificity test is essential for screening in low-prevalence populations.

14.27.4 Assumptions

Test characteristics (Sens, Spec) generalise to the target population; prevalence is correctly estimated.

14.27.5 R Implementation

```r
library(epiR)

# 2x2 table from a diagnostic study
tab <- as.table(matrix(c(90, 10,     # TP, FP
                         20, 880),   # FN, TN
                       nrow = 2, byrow = FALSE,
                       dimnames = list(Test = c("+", "-"),
                                       Disease = c("yes", "no"))))

epi.tests(tab, conf.level = 0.95)

# What happens when prevalence is only 1%?
sens <- 0.82; spec <- 0.99
for (p in c(0.5, 0.2, 0.05, 0.01)) {
  ppv <- sens * p / (sens * p + (1 - spec) * (1 - p))
```

```
  npv <- spec * (1 - p) / ((1 - sens) * p + spec * (1 - p))
  cat(sprintf("Prev=%.2f  PPV=%.3f  NPV=%.3f\n", p, ppv, npv))
}
```

14.27.6 Output & Results

Test statistics including PPV/NPV in the sample; manual computation shows PPV dropping sharply as prevalence falls, from 0.98 at 50 % prevalence to 0.45 at 1 %.

14.27.7 Interpretation

"In a population with 1 % prevalence, a positive test (sens 82 %, spec 99 %) has a PPV of only 0.45; most positives are false. For screening, test specificity drives PPV far more than sensitivity does."

14.27.8 Practical Tips

- Always report PPV and NPV at the **target** population's prevalence, not the study sample's.
- For low-prevalence settings, confirmatory testing after a positive screen is usually essential.
- Likelihood ratios avoid dependence on prevalence and combine multiplicatively with prior odds.
- Bayes' post-test probability: $P(D \mid +) = \mathrm{LR}(+) \cdot \mathrm{Prev} / (1 + \mathrm{LR}(+) \cdot \mathrm{Prev})$ using prior odds.
- Decision curves (Vickers-Elkin) integrate PPV / NPV at different thresholds into a utility measure.

14.27.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.27.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.28 Randomisation Methods

14.28.1 Introduction

Randomisation is the defining feature of an RCT: participants are allocated by chance rather than choice. Proper randomisation prevents selection bias (investigators cannot predict allocation) and ensures balance of known and unknown confounders in expectation. Several schemes trade simplicity, balance, and unpredictability.

14.28.2 Prerequisites

Probability; allocation concealment.

14.28.3 Theory

Simple randomisation: each participant flips a fair coin. Easy but can produce imbalance in small trials.

Block randomisation: within blocks of size B, allocate equal counts to each arm. Guarantees balance at block boundaries.

Stratified randomisation: block within strata defined by baseline covariates (sex, age, centre). Balances the stratification variables without post-hoc adjustment.

Minimisation (covariate-adaptive): allocate each new participant to minimise covariate imbalance; quasi-random, less transparent.

14.28.4 Assumptions

Allocation is concealed until the participant is enrolled; blocks / strata definitions are pre-specified.

14.28.5 R Implementation

```r
library(blockrand)

set.seed(2026)
# Block randomisation with variable block sizes (4, 6)
alloc <- blockrand(n = 60, num.levels = 2,
                   levels = c("ctrl", "trt"),
                   block.sizes = c(2, 3))
head(alloc, 10)
table(alloc$treatment)

# Stratified randomisation: stratify by sex
library(randomizeR)
```

```
pbr <- pbrPar(rb = c(4, 6), K = 2, ratio = c(1, 1))
rand_m <- genSeq(pbr, r = 1, seed = 2026)
rand_f <- genSeq(pbr, r = 1, seed = 2027)
```

14.28.6 Output & Results

Allocation sequence with approximately equal arm counts; variable block sizes prevent predictability at block boundaries.

14.28.7 Interpretation

"Block-randomisation with variable block sizes (4, 6) was used to allocate 60 participants, guaranteeing equal arm counts at every 20-participant batch. Stratification by centre ensured balance across sites."

14.28.8 Practical Tips

- Use variable block sizes (e.g., 4 and 6) to prevent predictability; investigators guessing the next allocation defeats concealment.
- Stratify on a small number of strong prognostic variables (typically 2-3); over-stratification creates empty cells.
- Centralise the allocation schedule; local administration risks unblinding.
- Document randomisation method and concealment in the published paper and trial registry.
- For small trials, stratified block randomisation is usually the best choice.

14.28.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.28.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.29 Crossover RCT Design

14.29.1 Introduction

A crossover randomised controlled trial assigns each participant to receive two or more treatments in sequence, separated by a washout period, so that every subject acts as their own control across the treatment comparison. The within-subject comparison removes between-subject variability — typically the dominant source of variance in clinical-pharmacology and chronic-disease studies — from the treatment estimate, giving the crossover design substantially more power than a parallel-group RCT of equal total sample size. Crossover designs are particularly common in early-phase clinical pharmacology, bioequivalence studies, sleep and migraine research, and other contexts in which the underlying condition is stable, treatment effects are reversible, and an adequate washout can eliminate pharmacological carryover.

14.29.2 Prerequisites

A working understanding of parallel-group RCT design, within-subject paired comparisons, mixed-effects models with subject as a random effect, and the concepts of period, sequence, and carryover effects.

14.29.3 Theory

The standard 2×2 crossover randomises participants to sequence AB (treatment A in period 1, B in period 2) or BA. The within-subject treatment contrast is the primary inference; analysis is by Grizzle's classical t-test on within-subject differences, or — preferably — a mixed-effects model with subject random intercept and fixed effects for period and treatment. The model is

$$y_{ijk} = \mu + \pi_j + \tau_k + s_i + \varepsilon_{ijk},$$

with period π_j, treatment τ_k, subject random intercept s_i, and residual error ε_{ijk}. **Carryover** — a residual treatment effect from period 1 lingering into period 2 — biases the within-subject estimate; it is formally tested by the sequence $\times$ period interaction but the test is under-powered, and adequate washout is the primary defence.

14.29.4 Assumptions

No carryover (washout long enough to eliminate the first-period treatment's effect), the condition is stable between periods (no progressive disease, no natural recovery), treatment effects are independent of period, and observations within each subject share a Normal distribution with constant variance.

14.29.5 R Implementation

```
library(nlme)

set.seed(2026)
n <- 20
sequence <- sample(c("AB", "BA"), n, replace = TRUE)
subj_eff <- rnorm(n, 0, 1)

y_A <- subj_eff + rnorm(n, 0, 0.5)
y_B <- subj_eff + 0.5 + rnorm(n, 0, 0.5)

df <- data.frame(
  subject  = rep(1:n, each = 2),
  period   = rep(1:2, n),
  treatment = unlist(lapply(sequence, function(s) strsplit(s, "")[[1]])),
  sequence = rep(sequence, each = 2),
  y        = unlist(lapply(1:n, function(i)
    if (sequence[i] == "AB") c(y_A[i], y_B[i]) else c(y_B[i], y_A[i])))
)

fit <- lme(y ~ treatment + period, random = ~ 1 | subject, data = df)
summary(fit)$tTable
```

14.29.6 Output & Results

The mixed-effects fit returns the treatment effect with within-subject standard error and a separate period effect that adjusts for any drift between the two periods. The random subject intercept absorbs between-subject variation and is the source of the crossover design's power advantage; reporting the variance components alongside the fixed-effect estimate makes the design's gain explicit.

14.29.7 Interpretation

A reporting sentence: "The two-period crossover analysis with mixed-effects modelling estimated the B–A treatment difference as 0.48 (95 % CI 0.22 to 0.74, $p = 0.002$), achieving over three-fold more precision than an equivalent parallel-group design with the same number of subjects. The period effect was small and non-significant ($p = 0.51$), and the sequence × period interaction (carryover diagnostic) was non-significant ($p = 0.78$), supporting the no-carryover assumption. Reporting follows the CONSORT extension for crossover trials." Always report period and carryover.

14.29.8 Practical Tips

- Test the carryover hypothesis formally via the sequence × period interaction, but rely on design — an adequately long washout, conventionally at least five half-lives of the active compound — as the primary defence rather than the underpowered post-hoc test.
- Unbalanced sequences (very different counts of AB and BA participants) reduce design efficiency and complicate analysis; aim for sequence balance via stratified randomisation on sequence.
- More than two periods (Latin-square or Williams designs) improve power and allow comparison of more than two treatments, at the cost of complexity, longer trial duration, and more potential for dropout — which crossover designs handle poorly because dropouts lose paired information.
- If the underlying condition evolves substantially within the trial timeframe (progressive disease, recovery, growth), the crossover design is inappropriate; the stability assumption is hard to defend and biases the estimate.
- Report per the CONSORT extension for crossover trials, including the trial flow diagram (per period), the sequence allocation, washout duration, and the carryover diagnostic.
- For ordinal or binary outcomes in a crossover design, generalised mixed-effects models (`glmer`) or paired analyses on the within-subject contingency table (McNemar) are the appropriate analysis approaches.

14.29.9 R Packages Used

`nlme::lme()` and `lme4::lmer()` for mixed-effects analysis with subject random intercepts; `Crossover` for canonical crossover-design construction including Williams squares and higher-order designs; `crossdes` for systematic generation of balanced crossover layouts; `bear` for end-to-end bioequivalence analysis on crossover data; `Mediana` for trial-design simulation including crossover designs.

14.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.29.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.30 Parallel-Group RCT Design

14.30.1 Introduction

The parallel-group randomised controlled trial (RCT) is the gold standard for evaluating interventions. Participants are randomly allocated to one of two (or more) arms – typically a new intervention vs control – and followed for a pre-specified outcome. Randomisation balances known and unknown confounders in expectation; blinding further reduces bias.

14.30.2 Prerequisites

Randomisation; hypothesis testing; sample-size calculation.

14.30.3 Theory

Essential elements: - **Primary outcome** (continuous, binary, time-to-event) with a clinically meaningful effect size. - **Allocation ratio** (usually 1:1, occasionally 2:1 or 3:1 for rare interventions). - **Randomisation list** (pre-generated, concealed at allocation time). - **Blinding** (single, double, triple). - **Pre-registered statistical analysis plan**.

Primary analysis is typically intent-to-treat (ITT), comparing groups as randomised.

14.30.4 Assumptions

Participants are exchangeable post-randomisation; allocation is fully concealed; outcome assessment is blinded; follow-up is complete or MAR.

14.30.5 R Implementation

```
library(pwr)

# Sample-size for two-arm comparison of means
# Effect: difference of 0.5 SD, alpha = 0.05, power = 0.80
pwr.t.test(d = 0.5, sig.level = 0.05, power = 0.80,
           type = "two.sample", alternative = "two.sided")

# Simulate an RCT with continuous outcome
set.seed(2026)
n_per_arm <- 64
arm <- rep(c("ctrl", "trt"), each = n_per_arm)
y   <- rnorm(2 * n_per_arm, mean = ifelse(arm == "trt", 0.5, 0))

# ITT analysis: two-sample t-test
t.test(y ~ arm, var.equal = TRUE)
```

```r
# Adjusted analysis with a pre-specified covariate (ANCOVA)
covar <- rnorm(2 * n_per_arm)
summary(lm(y ~ arm + covar))$coefficients
```

14.30.6 Output & Results

Sample size ~64 per arm for 80 % power; ITT t-test recovers the effect; ANCOVA-adjusted analysis gives a similar point estimate with smaller SE when the covariate is prognostic.

14.30.7 Interpretation

"The trial randomised 128 participants 1:1 to intervention vs control; the intervention arm showed a 0.52 SD improvement (95 % CI 0.18-0.86, p = 0.003) on the primary outcome, analysed by ANCOVA adjusting for baseline value."

14.30.8 Practical Tips

- Register the protocol and SAP **before** enrolment (clinicaltrials.gov, EUDRACT).
- Pre-specify the primary outcome and analysis; secondary outcomes are exploratory.
- Blinding is protective; document how it was broken (unblinding events, assessment).
- Report per CONSORT 2010 guidelines; flow diagram is mandatory.
- ITT is the primary analysis; supportive per-protocol analyses are sensitivity.

14.30.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.30.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.31 Reliability and Cronbach's Alpha

14.31.1 Introduction

Cronbach's alpha, introduced by Lee Cronbach in 1951, summarises the internal consistency of a multi-item scale by quantifying how strongly the items co-vary after accounting for the total number of items. The intuition is that items intended to measure the same underlying construct should correlate with each other; alpha rises with the average inter-item correlation and with the number of items. Cronbach's alpha is now ubiquitous in questionnaire validation, patient-reported-outcome (PRO) instrument development, psychometric evaluation of clinical scales, and any setting where a composite score is formed from multiple ordinal or continuous items. Despite well-known statistical limitations, it remains the single most reported reliability statistic in the clinical-research literature.

14.31.2 Prerequisites

A working understanding of classical test theory, the concept of a true score plus measurement error, and the construction of composite scores from multi-item rating scales.

14.31.3 Theory

Cronbach's alpha is

$$\alpha = \frac{k}{k-1}\left(1 - \frac{\sum_{i=1}^{k}\sigma_i^2}{\sigma_T^2}\right),$$

with k the number of items, σ_i^2 the variance of item i, and σ_T^2 the variance of the total summed score. The statistic ranges from $-\infty$ (theoretically; in practice 0) to 1. Conventional thresholds are 0.70 (acceptable), 0.80 (good), and 0.90 (excellent), with values above 0.95 typically indicating redundant items rather than superior reliability. McDonald's omega is a more flexible alternative when the assumption of tau-equivalence (equal true-score variances across items) is doubtful.

14.31.4 Assumptions

Items are tau-equivalent (each item measures the same true construct with the same loading), the scale is unidimensional (a single underlying factor explains the inter-item covariance structure), and items are continuous or quasi-continuous (Likert with at least 5 levels). Violations are common, and McDonald's omega or hierarchical-omega estimators give a more honest reliability estimate when the assumptions are not met.

14.31.5 R Implementation

```r
library(psych)

set.seed(2026)
n <- 200
theta <- rnorm(n)
items <- sapply(1:8, function(i) 0.7 * theta + rnorm(n, 0, 0.7))
colnames(items) <- paste0("q", 1:8)

alpha_res <- psych::alpha(items)
alpha_res$total
alpha_res$alpha.drop[, c("raw_alpha", "std.alpha")]
```

14.31.6 Output & Results

`psych::alpha()` returns the raw and standardised Cronbach's alpha for the full scale, plus an "alpha drop" table showing how alpha would change if each item were removed. A large positive drop (alpha increases without an item) flags that item as inconsistent with the rest of the scale and a candidate for revision or removal. The output also includes 95 % confidence intervals (via Feldt's method) and the average inter-item correlation, which is often more informative than alpha itself.

14.31.7 Interpretation

A reporting sentence: "The eight-item PRO scale had Cronbach's alpha 0.82 (95 % CI 0.78 to 0.86, Feldt method), indicating good internal consistency. The average inter-item correlation was 0.36, supporting the tau-equivalence assumption qualitatively. No single item substantially changed alpha when dropped (largest drop −0.01), so all items were retained for the final scale. McDonald's omega-total was 0.83, in close agreement with alpha." Always report both alpha and omega when feasible.

14.31.8 Practical Tips

- Alpha depends on the number of items: long scales inflate alpha mechanically, even when items are only weakly inter-correlated. Reporting the average inter-item correlation alongside alpha gives readers a fairer picture of true item coherence.
- Low alpha (< 0.70) may reflect a multi-dimensional scale rather than unreliable items. Always check the factor structure with exploratory or confirmatory factor analysis before concluding that items are unreliable.
- Very high alpha (> 0.95) suggests redundant items measuring essentially the same content; consider trimming the scale by removing the most redundant items, which improves administrative efficiency without sacrificing

reliability.
- For ordinal items with fewer than five categories, the standard Pearson-correlation-based alpha is biased downward; use ordinal alpha (computed from polychoric correlations) or McDonald's omega instead.
- Pair Cronbach's alpha with confirmatory factor analysis (CFA) to verify the assumed unidimensional structure; alpha computed on a multidimensional scale is uninterpretable as a reliability statistic.
- For test-retest reliability and inter-rater reliability, the appropriate statistics are the intraclass correlation coefficient (ICC) and Cohen's or Fleiss's kappa, respectively; alpha measures only internal consistency, not stability or agreement.

14.31.9 R Packages Used

`psych::alpha()` for the canonical Cronbach's alpha with drop analysis, `psych::omega()` for McDonald's omega and hierarchical-omega; `ltm::cronbach.alpha()` as a lightweight alternative; `MBESS::ci.reliability()` for advanced CIs (Feldt, bootstrap, Bonett); `lavaan` for confirmatory factor analysis to verify the unidimensional assumption.

14.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.31.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.32 ROC Analysis

14.32.1 Introduction

The receiver-operating-characteristic (ROC) curve traces sensitivity against $1 - $ specificity across all possible decision thresholds of a continuous-valued diagnostic test or risk score. Where a single sensitivity-specificity pair characterises performance at one cut-off, the ROC curve summarises performance across every cut-off and therefore captures the discriminative ability of the underlying continuous measurement independently of any chosen threshold. The area under

the curve (AUC) reduces this curve to a single number with a clean probabilistic interpretation: it equals the probability that a randomly chosen diseased case has a higher biomarker value than a randomly chosen non-diseased case. AUC values of 0.5 indicate chance-level discrimination and 1.0 indicates perfect separation; values between 0.7 and 0.9 are typical for clinically useful biomarkers.

14.32.2 Prerequisites

A working understanding of sensitivity and specificity, the role of decision thresholds in diagnostic-test performance, and the trade-off between true and false positive rates that the ROC curve makes explicit.

14.32.3 Theory

For a continuous biomarker X and binary disease status D, the ROC curve is the parametric plot

$$\mathrm{ROC}(c) = \big(\, 1 - F_0(c),\, 1 - F_1(c) \,\big) \quad \text{for all } c,$$

where F_0 and F_1 are the CDFs of X in non-diseased and diseased populations, respectively. The AUC has the equivalent representation

$$\mathrm{AUC} = P(X_1 > X_0),$$

with X_1 a random observation from the diseased and X_0 from the non-diseased population — the probability of correct ranking. Confidence intervals on AUC are typically computed via the DeLong non-parametric method or by bootstrap; bootstrap is also the standard approach for inference on the curve itself.

14.32.4 Assumptions

Test values are measured on a continuous scale (or at least an ordinal scale with many levels), disease status is correctly classified by a reliable gold-standard reference, and observations are independent. The interpretation as a discrimination measure is independent of disease prevalence — a feature of AUC that distinguishes it from predictive values.

14.32.5 R Implementation

```
library(pROC)

set.seed(2026)
n <- 200
disease <- factor(sample(c(0, 1), n, replace = TRUE, prob = c(0.6, 0.4)))
biomarker <- rnorm(n, mean = ifelse(disease == 1, 1.0, 0), sd = 1)
```

```
roc_obj <- roc(response = disease, predictor = biomarker,
               levels = c("0", "1"), direction = "<")
auc(roc_obj)
ci.auc(roc_obj, method = "delong")

plot(roc_obj, col = "#2A9D8F", lwd = 2, legacy.axes = TRUE,
     main = "ROC curve for biomarker")
abline(0, 1, lty = 2, col = "grey60")
```

14.32.6 Output & Results

`roc()` constructs the ROC object; `auc()` returns the area under the curve and `ci.auc()` provides the DeLong or bootstrap confidence interval. The standard plot shows sensitivity on the vertical axis and $1 - $ specificity on the horizontal axis, with the chance-diagonal as a reference line. Points along the curve correspond to sensitivity-specificity trade-offs at different cut-off values.

14.32.7 Interpretation

A reporting sentence: "The biomarker showed good discrimination for the binary disease outcome with AUC 0.77 (95 % CI 0.70 to 0.84, DeLong method); this exceeds the conventional 'fair' threshold of 0.70 but falls short of 'excellent' ($\geq$ 0.90). At the Youden-optimal cut-off (biomarker $\geq$ 0.42), sensitivity was 73 % and specificity 70 %; alternative cut-offs prioritising specificity (e.g., $\geq$ 1.0, sensitivity 51 %, specificity 87 %) may be preferred for screening applications." Always report both AUC and at least one operating point.

14.32.8 Practical Tips

- Report AUC with a 95 % confidence interval — DeLong's non-parametric method is the standard for paired comparisons and bootstrap is preferable for very small or imbalanced samples; AUC without uncertainty bounds is uninterpretable.
- Test AUC against 0.5 (chance) using `pROC::roc.test()`; compare two AUCs from the same cases using a paired DeLong test, which respects the correlation induced by shared subjects.
- Partial AUC over a clinically relevant region — for example, AUC restricted to specificity above 0.9 in a screening context — is often more informative than full AUC, because clinical use rarely spans the full operating range.
- AUC is threshold-independent; combine it with a calibration analysis (Hosmer-Lemeshow, calibration intercept and slope, calibration plot) or a decision-curve analysis when threshold-dependent decisions matter for the clinical context.

- For heavily imbalanced data (rare disease, screening contexts), precision-recall AUC is often more informative than ROC AUC because the ROC can look deceptively good when most subjects are non-diseased; PRAUC focuses on the positive class.
- When comparing biomarkers, include the increment in AUC (Δ AUC), the integrated discrimination index (IDI), and the net reclassification improvement (NRI); each captures a different facet of incremental performance.

14.32.9 R Packages Used

`pROC` for canonical ROC analysis with DeLong and bootstrap CIs, partial AUC, and paired comparisons; `ROCR` for an alternative interface with comprehensive performance-measure support; `PRROC` for precision-recall AUC and area-under-the-PR-curve analyses; `cutpointr` for principled threshold selection; `rms::lrm()` for AUC reporting integrated with logistic-regression model evaluation.

14.32.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.32.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.33 Sample Size Re-Estimation

14.33.1 Introduction

Sample-size re-estimation (SSR) is an adaptive-design technique that updates a clinical trial's planned sample size mid-study using interim estimates of nuisance parameters such as the within-group standard deviation (continuous outcomes) or the control-arm event rate (binary outcomes). It is one of the most common and least controversial trial adaptations because, when done in the blinded form, it does not access treatment-effect information and therefore preserves the type-I error rate under mild and well-understood conditions. SSR is now standard practice when pilot-data estimates of nuisance parameters are uncertain at the planning stage and the trial is otherwise large enough that the consequences of an under- or over-estimated nuisance parameter would be substantial.

14.33.2 Prerequisites

A working understanding of sample-size calculation, adaptive trial design, the distinction between blinded and unblinded interim analyses, and the regulatory framework around protocol-pre-specified adaptations.

14.33.3 Theory

Blinded SSR uses interim estimates of nuisance parameters from pooled (across-arm) data only, without revealing any arm-level information. For continuous outcomes the pooled standard deviation suffices; for binary outcomes, the pooled event rate. Blinded SSR does not inflate type-I error under standard conditions and is widely accepted by regulators with minimal formal control.

Unblinded SSR lets a Data Monitoring Committee see interim results by arm, supporting more flexible re-estimation rules at the cost of formal multiplicity control. The standard implementation is a combination-test or promising-zone design (Mehta and Pocock, 2011) that preserves conditional type-I error through explicit weighting of the interim and final test statistics.

14.33.4 Assumptions

The relevant nuisance parameter is unknown at planning but estimable from interim data; the interim analysis preserves blinding where required; the SSR rule is pre-specified in the protocol and statistical analysis plan; and a maximum sample size cap is set in advance to avoid unlimited re-estimation.

14.33.5 R Implementation

```
library(rpact)

n_initial <- ceiling(
  getSampleSizeMeans(alternative = 0.3, stDev = 1,
                     alpha = 0.025, beta = 0.2,
                     groups = 2)$numberOfSubjects
)
n_initial

n_updated <- ceiling(
  getSampleSizeMeans(alternative = 0.3, stDev = 1.4,
                     alpha = 0.025, beta = 0.2,
                     groups = 2)$numberOfSubjects
)
n_updated

c(planned = n_initial, revised = n_updated)
```

14.33.6 Output & Results

The script computes the planned sample size given the protocol-assumed standard deviation and the revised sample size given the interim-observed standard deviation. The ratio of the two ($1.4^2 = 1.96$) drives the proportional increase in required sample size — a familiar "variance is squared in the sample-size formula" relationship that makes SSR especially valuable when the within-group SD was uncertain at planning.

14.33.7 Interpretation

A reporting sentence: "A pre-specified blinded sample-size re-estimation at 50 % information accrual revealed a pooled within-group SD of 1.38, compared with the protocol-assumed 1.00. Per the pre-specified rule, the sample size was increased from 350 to 680 to maintain 80 % power against the originally specified treatment effect of 0.3 SD. The increase did not access treatment-arm information and therefore did not inflate the type-I error rate. The blinded SSR was conducted by an unblinded statistician within the DMC operating manual." Always report the blinding status and the cap.

14.33.8 Practical Tips

- Use blinded SSR whenever possible; it is operationally simpler, less controversial with regulators, and adequate for the most common SSR application (revising the within-group SD or pooled event rate).
- Unblinded SSR requires a statistical method that explicitly preserves the type-I error rate — typically a combination-test design (Cui-Hung-Wang or Mehta-Pocock promising-zone) that weights the interim and final test statistics in a pre-specified way.
- Pre-specify the SSR trigger condition, the re-estimation rule, and the maximum sample-size cap in the protocol and SAP; unlimited re-estimation without a cap is not acceptable to regulators and creates an open-ended commitment that sponsors rarely want to make.
- Document the SSR decision rationale in the final trial report — whether the SSR triggered, what the interim nuisance-parameter estimate was, and what the revised sample size became — so reviewers can assess the decision.
- SSR is especially valuable when pilot-data estimates of nuisance parameters are uncertain or when the trial population is expected to differ from the pilot in ways that affect variance or event rate; reliable prior data make SSR less necessary.
- Combine SSR with group-sequential efficacy and futility boundaries for a fully adaptive design that handles both nuisance-parameter uncertainty and effect-size revision; the combination is now standard in many phase-3 trials.

14.33.9 R Packages Used

`rpact` for canonical adaptive-design analysis including blinded and unblinded SSR with built-in type-I error control; `gsDesign` and `adaptTest` for combination-test SSR with promising-zone analysis; `Mediana` for trial-design simulation including SSR strategies; `RPACT::getDataset()` for stage-data integration in re-estimation workflows; `Hmisc` and `pwr` for the underlying classical sample-size formulas.

14.33.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.33.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.34 Sensitivity Analyses in Clinical Trials

14.34.1 Introduction

Sensitivity analyses complement the primary analysis by varying key assumptions – missing-data mechanism, model form, inclusion criteria – to assess how robust conclusions are. Tipping-point analyses identify how extreme assumptions must be before the primary conclusion flips, providing a concrete interpretation.

14.34.2 Prerequisites

Primary analysis; missing-data mechanisms; multiple imputation.

14.34.3 Theory

Tipping-point analysis under MI: impute missing outcomes in the experimental arm with an increasing shift δ (less favourable); re-pool. The smallest δ that makes the treatment effect no longer significant is the "tipping point". A large tipping point means the conclusion is robust.

Other sensitivity analyses: different imputation methods, PP vs ITT, different covariate adjustments, varying inclusion criteria, alternative parametric models.

14.34.4 Assumptions

Sensitivity analyses are pre-specified; tipping points are interpreted clinically, not mechanically.

14.34.5 R Implementation

```r
library(mice)

set.seed(2026)
n <- 200
arm <- factor(rep(c("ctrl", "trt"), each = n/2))
baseline <- rnorm(n, 5, 1)
outcome  <- 0.6 * baseline + ifelse(arm == "trt", 0.8, 0) +
            rnorm(n, 0, 1)
outcome[sample(n, 40)] <- NA

df <- data.frame(arm, baseline, outcome)

# Base analysis under MAR
imp <- mice(df, m = 20, method = "pmm", printFlag = FALSE)
summary(pool(with(imp, lm(outcome ~ arm + baseline))))$estimate[2]

# Tipping-point analysis: penalise imputed trt-arm outcomes by delta
deltas <- seq(0, 2, by = 0.25)
effs <- sapply(deltas, function(d) {
  imp2 <- imp
  for (k in 1:20) {
    idx <- which(is.na(df$outcome) & df$arm == "trt")
    imp2$imp$outcome[[k]][df$arm[idx] == "trt"] <-
      imp2$imp$outcome[[k]][df$arm[idx] == "trt"] - d
  }
  summary(pool(with(imp2, lm(outcome ~ arm + baseline))))$estimate[2]
})
data.frame(delta = deltas, trt_effect = round(effs, 3))
```

14.34.6 Output & Results

Treatment effect declines linearly with δ; the tipping point is where the effect crosses zero (or loses significance).

14.34.7 Interpretation

"A delta shift of 1.6 on the imputed intervention-arm outcomes was required to eliminate significance; clinically this would require intervention dropouts to fare 1.6 SD worse than MAR predicts. The primary conclusion is robust to plausible MNAR mechanisms."

14.34.8 Practical Tips

- Pre-specify all sensitivity analyses in the SAP; post-hoc analyses are exploratory.
- Tipping points are most informative when expressed on the clinical scale.
- ICH E9 R1 promotes sensitivity-analysis thinking tied to the estimand.
- Running many sensitivity analyses is fine; interpret them holistically.
- Reporting a tipping-point figure in the primary paper is increasingly standard for high-missingness trials.

14.34.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.34.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.35 Stepped-Wedge Trial

14.35.1 Introduction

Stepped-wedge cluster-randomised trials (SW-CRTs) randomise not whether each cluster receives the intervention but the time at which each cluster transitions from control to intervention. By the end of the trial all clusters have received the intervention, which gives the design particular ethical and pragmatic appeal: it is well suited to settings where investigators believe the intervention is likely beneficial (so withholding it from some clusters indefinitely would be ethically uncomfortable), where logistical constraints prevent simultaneous rollout across all clusters, or where a programme rollout is happening anyway and the trial is exploiting the staggered implementation to learn the effect. Analysis

must carefully separate the secular time trend that affects all clusters from the treatment effect that is realised at different calendar times in different clusters.

14.35.2 Prerequisites

A working understanding of cluster-randomised trials, time-trend modelling, and mixed-effects models with cluster random intercepts and (optionally) cluster-time random effects.

14.35.3 Theory

Clusters are randomised to sequence rather than to arm; at each step (period) a new subset of clusters crosses over from control to intervention. The resulting data structure provides two complementary contrasts: a between-cluster comparison at each period (like a parallel-arm CRT at that period) and a within-cluster before-after comparison around each cluster's switch-point. The standard analysis is a mixed-effects model with fixed effects for time period and treatment status and a random intercept for cluster:

$$y_{ijk} = \mu + \tau_t + \beta \cdot X_{jk} + u_j + \varepsilon_{ijk},$$

with τ_t the period effect, X_{jk} the treatment indicator for cluster j at time k, and $u_j \sim N(0, \sigma_c^2)$ the cluster random effect. Including the period fixed effects is mandatory because secular trends confound the treatment estimate.

14.35.4 Assumptions

Secular time trends are common across clusters (any cluster-specific time trend should be modelled explicitly), the treatment effect is immediate and stable after switch (or any fade-in/fade-out is explicitly modelled), and the within-cluster correlation structure is correctly specified.

14.35.5 R Implementation

```
library(lme4); library(lmerTest)

set.seed(2026)
n_cl <- 10; n_per <- 20; n_period <- 5
cluster <- rep(1:n_cl, each = n_period * n_per)
period  <- rep(rep(1:n_period, each = n_per), n_cl)

start_t <- sample(2:5, n_cl, replace = TRUE)
trt <- as.numeric(period >= rep(start_t, each = n_period * n_per))

time_trend <- 0.1 * (period - 1)
```

```
cl_re      <- rep(rnorm(n_cl, 0, 0.5), each = n_period * n_per)

y <- cl_re + time_trend + 0.4 * trt +
     rnorm(length(cluster), 0, 1)

df <- data.frame(cluster = factor(cluster),
                 period  = factor(period),
                 trt     = trt, y = y)

fit <- lmer(y ~ trt + period + (1 | cluster), data = df)
summary(fit)$coefficients["trt", ]
```

14.35.6 Output & Results

The mixed-effects model returns the treatment effect estimate with a standard error that reflects both the within-cluster and between-cluster information available in the staggered design. Including the period fixed effects absorbs the secular time trend; the random cluster intercept absorbs the between-cluster baseline variation; the residual captures within-cluster, within-period noise.

14.35.7 Interpretation

A reporting sentence: "The stepped-wedge mixed-effects analysis estimated a treatment effect of 0.38 SD (95 % CI 0.19 to 0.57, $p < 0.001$), adjusting for the calendar-period secular trend (which itself was significant, $\hat{\tau}_5 - \hat{\tau}_1 = 0.41$) and the cluster random intercepts (cluster ICC 0.20). Reporting follows the CONSORT extension for stepped-wedge CRTs." Always report both the secular trend and the treatment effect.

14.35.8 Practical Tips

- Always adjust for time period in the analysis; an unadjusted analysis confounds treatment with secular trend, and the bias can be substantial in any health-system setting where outcomes are improving (or worsening) over time independent of the intervention.
- Report per the CONSORT extension for stepped-wedge cluster-randomised trials (Hemming et al., 2018), which specifies the trial design figure, time-by-cluster matrix, and standard reporting requirements.
- Consider modelling time-varying treatment effects (a ramp-up over a few periods after the switch) for interventions that take time to implement fully; assuming an immediate stable effect when the intervention requires phased rollout biases the estimate downward.
- Sample-size calculation for stepped-wedge designs is intrinsically more complex than for parallel CRTs because it depends on the design matrix,

the within-cluster correlation, and the number of steps; use `swCRTdesign::swPwr()` or the Hussey-Hughes (2007) closed-form formula, and avoid naive parallel-CRT power approximations.
- The ethical advantage — all clusters eventually receive the intervention — is real but does not eliminate the need for equipoise; if investigators are confident the intervention works, the trial is arguably unnecessary regardless of design.
- Sensitivity analyses to the assumed correlation structure (compound symmetry vs Hooper-Girling vs more complex) are increasingly required by reviewers; report several specifications and check that conclusions are robust.

14.35.9 R Packages Used

`lme4::lmer()` and `lmerTest` for canonical stepped-wedge mixed-effects analysis; `swCRTdesign::swPwr()` and `swCRTdesign::swSummary()` for design and power calculation; `clusterPower` for general cluster-trial power including stepped wedge; `geepack::geeglm()` for GEE-based marginal-model analysis as an alternative; `glmmTMB` for stepped-wedge analyses with non-Normal outcomes (count, binary).

14.35.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.35.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.36 Stratified Randomisation

14.36.1 Introduction

Stratified randomisation runs a separate block-randomisation list within each stratum defined by one or more baseline covariates — typically centre, sex, age category, disease severity, or other strong prognostic factors. By randomising within strata, the design guarantees balance of the stratification variables across treatment arms, which stabilises subgroup inference, pre-empts the situation in which a strongly prognostic covariate drives apparent arm differences, and is

increasingly required by regulators and journals for any multi-centre or prognostically heterogeneous trial. The trade-off is a slight increase in implementation complexity and the risk of empty strata in small trials, both manageable with care.

14.36.2 Prerequisites

A working understanding of simple and block randomisation, the role of baseline covariates as potential confounders or effect modifiers, and the analytical principle that the design should be reflected in the analysis model.

14.36.3 Theory

Strata are defined by a cross-tabulation of one or more pre-specified factors — typically 4 to 8 strata total in a real trial, formed by combining centre with one or two prognostic factors. Within each stratum, block randomisation proceeds independently with its own variable-sized blocks, ensuring that arm counts are balanced both globally and within every stratum at every block boundary. The method trades a small amount of design simplicity for guaranteed marginal balance on the stratification variables and substantial protection against centre-by-treatment confounding in multi-centre trials.

14.36.4 Assumptions

Stratification variables are known and recorded before randomisation (post-hoc stratification is not stratified randomisation but rather post-hoc adjustment), the strata are clinically meaningful and prognostically important, and the trial is large enough that no stratum will end up with too few subjects to support stable within-stratum analysis.

14.36.5 R Implementation

```
library(blockrand)

set.seed(2026)

strata <- c("A", "B", "C")
schedule <- do.call(rbind, lapply(strata, function(s) {
  b <- blockrand(n = 20, num.levels = 2,
                 levels = c("ctrl", "trt"),
                 block.sizes = c(2, 3))
  b$centre <- s
  b
}))

table(schedule$centre, schedule$treatment)
```

```
head(schedule, 8)
```

14.36.6 Output & Results

The script generates three centre-specific allocation schedules and combines them into a master list. The cross-tabulation of centre by treatment shows equal arm counts within each centre — the design's signature property — and the master list is then exported to the trial's interactive web-response system for execution.

14.36.7 Interpretation

A reporting sentence: "Treatment allocation was stratified by centre (three sites) and by baseline disease severity (mild, moderate, severe), with variable-sized permuted blocks of 2 and 3 within each of the six strata. This guaranteed equal arm allocation at every centre and within every severity stratum, preventing centre-by-treatment and severity-by-treatment confounding. Final arm counts were exactly balanced within every stratum (50 patients per arm per stratum)." Always describe the stratification scheme.

14.36.8 Practical Tips

- Stratify on the one to three strongest prognostic variables, and not more; more strata mean smaller stratum sizes, more empty cells (especially in small trials), and progressively diminishing protection against the very imbalance the stratification was meant to prevent.
- Centre is the standard stratification variable for multi-centre trials and is virtually always recommended; centre-specific outcome differences are common and centre-by-treatment confounding can substantially bias the overall estimate.
- Always analyse with the stratification variables as covariates in the analysis model; stratified randomisation by itself does not produce the correct standard error if the analysis ignores the stratification — analysing as if the trial were simple-randomised understates the precision of the estimate.
- Do not stratify on a variable you intend to adjust for analytically without that variable being prognostic — redundant stratification dilutes randomisation entropy without analytic benefit. Conversely, every variable used for stratification should also enter the analysis as a covariate.
- For small trials (typically fewer than 100 subjects total) where stratification on multiple factors would create empty strata, minimisation (Pocock-Simon) is a compromise that balances multiple covariates without forcing strict block structure.
- The trial's interactive web-response system (IWRS) handles the multi-stratum allocation in real time; running stratified randomisation by hand

in a multi-centre trial is operationally fragile and a frequent source of allocation-concealment failures.

14.36.9 R Packages Used

`blockrand` for canonical stratified block randomisation with built-in stratum looping; `randomizr` for tidyverse-friendly stratified randomisation with explicit unit-of-randomisation control; `Minirand::Pocock` for minimisation as an alternative when stratum cells would be too sparse; `bcrm` and related packages for biased-coin variants; `Mediana` for trial-design simulation including stratified-randomisation strategies.

14.36.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.36.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.37 Subgroup Analyses

14.37.1 Introduction

Subgroup analyses in clinical trials assess whether the overall treatment effect varies across pre-specified baseline characteristics — age, sex, disease severity, comorbidity, biomarker status. They provide important insight into treatment heterogeneity, support guideline development, and inform precision-medicine decisions. They are also notoriously prone to over-interpretation: with enough subgroups and enough cut-points, false-positive heterogeneity findings appear by chance alone, and the literature is replete with cautionary tales of subgroup claims that failed to replicate. Modern guidance (CONSORT, ICH-E9, regulatory subgroup analysis frameworks) emphasises pre-specification, formal interaction tests, and graphical communication via forest plots, while warning against per-subgroup hypothesis testing as a substitute for the interaction test.

14.37.2 Prerequisites

A working understanding of treatment effect estimation in randomised trials, interaction terms in regression, the multiple-testing problem, and the regulatory framework around pre-specified versus post-hoc analyses.

14.37.3 Theory

The statistically appropriate test for effect heterogeneity is the **treatment-by-subgroup interaction** in a regression of the outcome on treatment, subgroup, and their product. The standard reporting set includes the overall treatment effect with its 95 % CI, the subgroup-specific effects with CIs, and the interaction p-value. Comparing within-subgroup p-values across strata (the "significant in one, not the other" fallacy) is statistically incorrect because each within-subgroup test is under-powered and the comparison ignores the multiplicity.

Pre-specification is the key safeguard: a small number of biologically-motivated subgroups documented in the SAP carry interpretable evidentiary weight, while post-hoc subgroup discovery is at best hypothesis-generating and at worst misleading.

14.37.4 Assumptions

The subgroups are pre-specified in the protocol or SAP, the subgrouping covariates are measured at baseline rather than on-treatment (avoiding immortal-time bias and other post-randomisation issues), and the trial is large enough that the interaction test has at least minimal power — usually not the case in practice, which is why subgroup tests rarely reach significance.

14.37.5 R Implementation

```
set.seed(2026)
n <- 400
arm <- factor(rep(c("ctrl", "trt"), each = n/2))
sex <- factor(sample(c("M", "F"), n, replace = TRUE))

y <- ifelse(arm == "trt",
            ifelse(sex == "F", 1.0, 0.3), 0) +
     rnorm(n)

fit <- lm(y ~ arm * sex)
summary(fit)$coefficients

by(data.frame(y, arm), sex, function(df) {
  t.test(y ~ arm, data = df)$estimate
})
```

14.37.6 Output & Results

The interaction term in the regression model is the formal test of effect modification by subgroup; the per-subgroup t-tests give the subgroup-specific point estimates that populate the forest plot. Reporting both — interaction p-value plus subgroup-specific estimates with CIs — is the standard expected by trial reporting guidelines.

14.37.7 Interpretation

A reporting sentence: "The overall treatment effect was 0.65 (95 % CI 0.45 to 0.85, $p < 0.001$); the pre-specified treatment-by-sex interaction was significant ($p = 0.02$), with the effect in women (0.98, 95 % CI 0.69 to 1.27) approximately three-fold larger than in men (0.32, 95 % CI 0.04 to 0.61). This sex-by-treatment heterogeneity was hypothesised in the protocol on the basis of pharmacokinetic differences and is reported here as a confirmatory rather than exploratory finding; replication in an independent trial is desirable." Always state pre-specification status.

14.37.8 Practical Tips

- Pre-specify all subgroup analyses in the protocol and SAP; post-hoc subgroups are exploratory at best and should be flagged as such in any reporting, ideally in a separate section labelled "exploratory."
- Always interpret the interaction p-value as the test of heterogeneity, not the per-subgroup p-values; the per-subgroup tests are nearly always underpowered, and comparing their significance across subgroups is a well-known statistical fallacy.
- Forest plots are the standard way to communicate subgroup effects visually; they make magnitudes and uncertainties immediately legible and are now expected by most clinical-trial reporting guidelines.
- Limit pre-specified subgroups to four to six biologically motivated factors; a list of 20+ subgroups is a fishing expedition that nearly guarantees at least one false-positive interaction by chance, and reviewers will flag it.
- Heterogeneity-of-treatment-effect (HTE) methods — causal forests, BART, model-based recursive partitioning, the SIDES algorithm — are emerging for principled data-driven subgroup discovery, with appropriate multiplicity control built in. They are increasingly accepted as exploratory complements to traditional pre-specified subgroup analyses.
- For survival or time-to-event subgroup analyses, fit a Cox model with treatment, subgroup, and treatment × subgroup terms; the per-subgroup hazard ratios should be reported with 95 % CIs alongside the joint interaction test.

14.37.9 R Packages Used

Base R `lm()`, `glm()`, and `t.test()` for canonical subgroup analysis; `survival::coxph()` with interaction terms for survival subgroup analyses; `forestplot` and `forrester` for publication-quality subgroup forest plots; `grf` for generalised random forests with treatment-effect estimation; `SIDES` and `model4you` for principled exploratory subgroup discovery.

14.37.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.37.11 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.38 Weighted Kappa

14.38.1 Introduction

Weighted kappa extends Cohen's kappa to ordinal data by crediting partial agreement: near-miss disagreements (mild vs moderate) weigh less than dramatic disagreements (mild vs severe). It is the standard inter-rater agreement statistic for Likert-type scales, radiology grading, and symptom-severity ratings.

14.38.2 Prerequisites

Cohen's kappa; ordinal measurement.

14.38.3 Theory

$$\kappa_w = 1 - \frac{\sum_{ij} w_{ij} f_{ij}}{\sum_{ij} w_{ij} e_{ij}},$$

where w_{ij} is the disagreement weight between category i and j, f_{ij} observed cell frequency, e_{ij} expected under chance.

Weight schemes: - **Linear** $w_{ij} = |i-j|/(k-1)$ for k categories. - **Quadratic** $w_{ij} = (i-j)^2/(k-1)^2$ – more forgiving of near-miss disagreements.

Quadratic is most common for multi-category ordinal scales.

14.38.4 Assumptions

Category ordering is meaningful and equally spaced; two raters; independent ratings.

14.38.5 R Implementation

```
library(psych)

set.seed(2026)
n <- 100
# Two raters on a 5-point ordinal scale
rater1 <- sample(1:5, n, replace = TRUE,
                 prob = c(0.1, 0.2, 0.4, 0.2, 0.1))
# Rater 2 agrees within +-1 with prob 0.8, otherwise random
rater2 <- ifelse(rbinom(n, 1, 0.8) == 1,
                 pmax(1, pmin(5, rater1 + sample(-1:1, n, replace =
                  ↪  TRUE))),
                 sample(1:5, n, replace = TRUE))

cohen.kappa(cbind(rater1, rater2))$kappa    # unweighted
cohen.kappa(cbind(rater1, rater2),
            w = "squared")$weighted.kappa
cohen.kappa(cbind(rater1, rater2),
            w = "linear")$weighted.kappa
```

14.38.6 Output & Results

Unweighted kappa (~0.35), linear-weighted (~0.55), quadratic-weighted (~0.70); quadratic weights reward near-miss agreement more heavily.

14.38.7 Interpretation

"Weighted kappa with quadratic weights was 0.72 (95 % CI 0.62-0.82), consistent with substantial agreement; unweighted kappa of 0.35 understates agreement by not crediting near-miss ratings."

14.38.8 Practical Tips

- Use quadratic weights for most clinical ordinal scales; they reflect that 1-step disagreements matter far less than 3-step.
- Linear weights are appropriate when category spacing is more uniform-linear.
- Always specify the weight scheme when reporting weighted kappa.

- For continuous data with measurement error, use the intraclass correlation coefficient (ICC) instead.
- Quadratic-weighted kappa equals ICC(3, 1) under certain assumptions – the two methods converge for ordinal Likert scales.

14.38.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

14.38.10 See also — labs in this chapter

- Diagnostic testing: Se, Sp, PPV, NPV, LR
- Kappa, ICC, Bland–Altman
- Biomarker statistics (Youden, NRI, decision curves)
- TRIPOD-AI, fairness auditing, reproducibility at scale

14.39 Week 3, Session 3 — Biomarker statistics (Youden, NRI, decision curves)

Testing labs use the main template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

14.39.1 Learning objectives

- Compute ROC-AUC, Youden's index, sensitivity, and specificity at the optimal cut-point.
- Compute a net reclassification index between two predictive models.
- Construct a decision curve and interpret net benefit at a range of threshold probabilities.

14.39.2 Prerequisites

Binary classification and ROC curves; logistic regression.

14.39.3 Background

A candidate biomarker is not useful until it is tied to a decision. ROC-AUC summarises discrimination across all thresholds but is insensitive to *where* on the curve the action happens. Youden's index (sensitivity + specificity − 1) picks the threshold that maximises the equal-weighted sum. The net reclassification index (NRI) quantifies whether a new model reclassifies cases and non-cases

in the correct direction relative to a baseline. Decision curve analysis plots *net benefit* as a function of the threshold probability, and lets a reader compare strategies ("treat all", "treat none", "treat by model") across a clinically relevant range.

Discrimination, calibration, and net benefit are three complementary axes. A biomarker with high AUC that is poorly calibrated can produce harmful decisions; a perfectly calibrated biomarker with low AUC gives no useful ranking. Reporting all three keeps the evaluation honest.

Decision curves are not hypothesis tests. They are a principled way to put a clinical question — what is the harm-to-benefit ratio of acting on this prediction? — into the analysis, and to see how the answer depends on that ratio.

14.39.4 Setup

```
library(tidyverse)
library(pROC)
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

14.39.5 1. Hypothesis

Can a logistic model on `Pima.tr` (glucose, BMI, age) distinguish diabetic from non-diabetic patients well enough to support a screening decision?

14.39.6 2. Visualise

```
d <- as_tibble(MASS::Pima.tr)
ggplot(d, aes(glu, bmi, colour = type)) + geom_point(alpha = 0.7)
```

14.39.7 3. Assumptions

Independence of observations; probability of diabetes is monotone in the linear predictor; no missingness.

14.39.8 4. Conduct

Fit a simple logistic regression and compute discrimination.

```
fit <- glm(type ~ glu + bmi + age, data = d, family = binomial())
d$p <- predict(fit, type = "response")
r  <- roc(d$type, d$p, direction = "<", quiet = TRUE)
auc(r)
coords(r, "best", ret = c("threshold", "sensitivity", "specificity",
```

```
                              "youden"), transpose = FALSE)
```

NRI against a glucose-only baseline.

```
fit0 <- glm(type ~ glu, data = d, family = binomial())
p0 <- predict(fit0, type = "response")
p1 <- d$p
# Continuous NRI
case   <- d$type == "Yes"
nri_up <- mean(p1[case]  > p0[case])   - mean(p1[case]  < p0[case])
nri_dn <- mean(p1[!case] < p0[!case])  - mean(p1[!case] > p0[!case])
nri <- nri_up + nri_dn
c(nri_cases = nri_up, nri_noncases = nri_dn, nri_total = nri)
```

A manual decision curve.

```
thr <- seq(0.05, 0.6, by = 0.02)
dca <- sapply(thr, function(t) {
  treat <- p1 > t
  tp <- sum(treat &  case); fp <- sum(treat & !case); N <- length(case)
  tp / N - (fp / N) * (t / (1 - t))
})
nb_all <- sapply(thr, function(t) {
  tp <- sum(case); fp <- sum(!case); N <- length(case)
  tp / N - (fp / N) * (t / (1 - t))
})
tibble(threshold = thr, model = dca, treat_all = nb_all, treat_none = 0)
 ↪  |>
  pivot_longer(-threshold) |>
  ggplot(aes(threshold, value, colour = name)) + geom_line() +
  labs(x = "threshold probability", y = "net benefit")
```

14.39.9 5. Concluding statement

> A logistic model using glucose, BMI, and age discriminated diabetic from non-diabetic patients in `MASS::Pima.tr` with AUC `round(as.numeric(auc(r)),3)`. The Youden-optimal cut-point occurred at a predicted probability of `round(coords(r,"best",ret="threshold",transpose=FALSE)[1,1],2)`. Adding BMI and age to a glucose-only baseline produced an NRI of `round(nri,2)`; the decision curve showed net benefit above "treat all" for threshold probabilities from roughly 0.15 to 0.5.

Decision curves give the clinical context: if the decision to intervene at, say, $p = 0.2$ is under discussion, the model is useful; at $p = 0.05$ or $p = 0.7$, it is barely distinguishable from treat-all or treat-none.

14.39.10 Common pitfalls

- Reporting AUC without calibration or decision curves.
- Computing NRI with a categorical risk cut-point and failing to disclose the cut-off.
- Using the same data to develop and evaluate the biomarker (apparent performance).

14.39.11 Further reading

- Pencina MJ, D'Agostino RB Sr, et al. (2008), *Evaluating the added predictive ability of a new marker*.
- Vickers AJ, Elkin EB (2006), *Decision curve analysis*.

14.39.12 Session info

```
sessionInfo()
```

14.39.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

14.40 Week 2, Session 3 — Diagnostic testing: Se, Sp, PPV, NPV, LR

Inference labs use the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

14.40.1 Learning objectives

- Compute sensitivity, specificity, PPV, NPV, and positive and negative likelihood ratios from a 2x2 table.
- Convert pre-test probability to post-test probability with an LR.
- Sketch a receiver-operating characteristic curve from a continuous test statistic.

14.40.2 Prerequisites

Lab 2.2.

14.40.3 Background

A diagnostic test has two operating characteristics intrinsic to the test itself: *sensitivity* is the probability that a diseased person tests positive; *specificity* is

the probability that a disease-free person tests negative. These quantities are properties of the test. They do not change with prevalence.

Two other quantities are properties of the test *and* the population in which it is applied: *positive predictive value* is the probability of disease given a positive test; *negative predictive value* is the probability of no disease given a negative test. These change with prevalence, sometimes dramatically.

Likelihood ratios unify the two pairs. $LR+$ is sens / (1 − spec); $LR-$ is (1 − sens) / spec. They convert pre-test odds to post-test odds by multiplication, which is the cleanest way to combine a test result with prior information. An LR+ greater than 10 is a strong positive; less than 0.1 is a strong negative; values near 1 are uninformative.

14.40.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

14.40.5 1. Hypothesis

Question of interest: how does a continuous biomarker behave as a diagnostic test? We are not running an inferential test; we are characterising a test's discrimination.

14.40.6 2. Visualise

Simulate a biomarker that is higher in diseased cases than in disease-free controls, with overlap.

```
N <- 500
prev <- 0.2
pop <- tibble(
  id = seq_len(N),
  disease = rbinom(N, 1, prev),
  biomarker = rnorm(N, mean = if_else(disease == 1, 7, 5), sd = 1)
)
```

```
pop |>
  mutate(status = if_else(disease == 1, "disease", "no disease")) |>
  ggplot(aes(biomarker, fill = status)) +
  geom_density(alpha = 0.5, colour = NA) +
  geom_vline(xintercept = 6, linetype = 2) +
  labs(x = "Biomarker level", y = "Density", fill = NULL)
```

14.40.7 3. Assumptions

The gold standard for disease status is assumed perfect. The biomarker is continuous and must be dichotomised at some cutoff to behave like a positive/negative test. We choose 6 as the cutoff for illustration; in practice, the cutoff is itself an outcome of the analysis.

```
cutoff <- 6
pop <- pop |> mutate(test = as.integer(biomarker > cutoff))
tab <- table(disease = pop$disease, test = pop$test)
tab
```

14.40.8 4. Conduct

```
TP <- tab["1", "1"]; FN <- tab["1", "0"]
FP <- tab["0", "1"]; TN <- tab["0", "0"]

sens <- TP / (TP + FN)
spec <- TN / (TN + FP)
ppv  <- TP / (TP + FP)
npv  <- TN / (TN + FN)
lrp  <- sens / (1 - spec)
lrn  <- (1 - sens) / spec

diag_tbl <- tibble(
  quantity = c("Sensitivity", "Specificity",
               "PPV", "NPV", "LR+", "LR-"),
  value = c(sens, spec, ppv, npv, lrp, lrn)
)
diag_tbl
```

Convert pre-test odds to post-test odds with the LR.

```
pre_prob <- 0.1
pre_odds <- pre_prob / (1 - pre_prob)
post_odds_pos <- pre_odds * lrp
post_prob_pos <- post_odds_pos / (1 + post_odds_pos)
post_odds_neg <- pre_odds * lrn
post_prob_neg <- post_odds_neg / (1 + post_odds_neg)

tibble(
  pre_prob,
  post_prob_if_positive = post_prob_pos,
  post_prob_if_negative = post_prob_neg
)
```

Sketch an ROC by sweeping the cutoff.

```
roc <- tibble(
  cut = seq(min(pop$biomarker), max(pop$biomarker), length.out = 200)
) |>
  rowwise() |>
  mutate(
    tp = sum(pop$biomarker > cut & pop$disease == 1),
    fn = sum(pop$biomarker <= cut & pop$disease == 1),
    fp = sum(pop$biomarker > cut & pop$disease == 0),
    tn = sum(pop$biomarker <= cut & pop$disease == 0),
    sens = tp / (tp + fn),
    fpr  = fp / (fp + tn)
  ) |>
  ungroup()
```

```
ggplot(roc, aes(fpr, sens)) +
  geom_path(linewidth = 1) +
  geom_abline(linetype = 2, colour = "grey50") +
  coord_equal() +
  labs(x = "False positive rate (1 - specificity)",
       y = "Sensitivity")
```

14.40.9 5. Concluding statement

> With a cutoff of `cutoff`, the biomarker had sensitivity `round(sens,2)`, specificity `round(spec,2)`, PPV `round(ppv,2)`, and NPV `round(npv,2)`. The positive likelihood ratio was `round(lrp,2)` and the negative `round(lrn,2)`. A pre-test probability of 10% becomes `round(post_prob_pos,2)` after a positive test and `round(post_prob_neg,3)` after a negative test.

A single cutoff collapses a rich continuous score into two states. The ROC curve shows the trade-off across all cutoffs; the area under it summarises overall discrimination without committing to a threshold.

14.40.10 Common pitfalls

- Quoting a single cutoff's sensitivity and specificity as if they were fixed properties of the test, ignoring that a different cutoff gives different numbers.
- Confusing sensitivity with PPV in everyday speech.
- Forgetting that PPV and NPV depend on prevalence.
- Using an ROC to compare tests with different prevalence in each sample.

14.40.11 Further reading

- Altman DG & Bland JM, *Diagnostic tests* series, BMJ.

- Pepe MS. *The Statistical Evaluation of Medical Tests for Classification and Prediction*.

14.40.12 Session info

```
sessionInfo()
```

14.40.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

14.41 Week 4, Session 2 — Kappa, ICC, Bland–Altman

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

14.41.1 Learning objectives

- Compute Cohen's kappa for categorical agreement and explain its chance-correction.
- Compute an intraclass correlation coefficient for continuous agreement and distinguish consistency from absolute agreement.
- Draw a Bland–Altman plot and report limits of agreement.

14.41.2 Prerequisites

Basic R and ggplot2.

14.41.3 Background

Measurement-agreement studies ask whether two raters, two methods, or two instruments give the same answer on the same units. The choice of statistic depends on the scale of the measurement. Cohen's kappa adjusts simple percent agreement for the agreement expected by chance given the marginal frequencies; it ranges from -1 to 1 with common landmarks at 0.4 and 0.6. Its main weakness is sensitivity to prevalence.

For continuous measurements, the intraclass correlation (ICC) and the Bland–Altman plot answer complementary questions. The ICC is a single-number summary of reliability, defined in several flavours (one-way, two-way, consistency vs absolute). The Bland–Altman plot shows *pattern*: it plots the difference between two raters against their mean and marks the limits of agreement (typically mean

± 1.96 SD). It reveals bias, proportional bias, and heteroscedasticity that ICCs hide.

Reliability is not the same as agreement. Two raters can be highly correlated (one is always twice the other) and have a terrible agreement. Always report both and let the picture tell the pattern.

14.41.4 Setup

```
library(tidyverse)
library(broom)
library(psych)
library(irr)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

14.41.5 1. Goal

Build two small rater datasets — one categorical, one continuous — and compute the matching agreement statistics.

14.41.6 2. Approach

For the categorical example, simulate 100 radiograph classifications (3 categories) by two readers with substantial but not perfect agreement. For the continuous example, simulate 60 measurements by two instruments, one with a small constant bias.

```
# categorical
cats <- c("normal", "mild", "severe")
truth <- sample(cats, 100, replace = TRUE, prob = c(0.5, 0.3, 0.2))
r1 <- ifelse(runif(100) < 0.2, sample(cats, 100, replace = TRUE), truth)
r2 <- ifelse(runif(100) < 0.25, sample(cats, 100, replace = TRUE), truth)
kap_tbl <- tibble(r1 = factor(r1, levels = cats),
                  r2 = factor(r2, levels = cats))

# continuous
n <- 60
true_val <- rnorm(n, 100, 15)
inst1 <- true_val + rnorm(n, 0, 3)
inst2 <- true_val + 2 + rnorm(n, 0, 3)    # small positive bias
meas <- tibble(inst1, inst2)
```

14.41.7 3. Execution

Cohen's kappa:

```
kappa2(kap_tbl[, c("r1", "r2")])
```

ICC via `psych`:

```
ICC(as.matrix(meas))
```

Bland–Altman:

```
ba <- meas |>
  mutate(mean_val = (inst1 + inst2) / 2,
         diff_val = inst2 - inst1)
loa <- mean(ba$diff_val) + c(-1.96, 0, 1.96) * sd(ba$diff_val)

ggplot(ba, aes(mean_val, diff_val)) +
  geom_point(alpha = 0.7) +
  geom_hline(yintercept = loa[1], linetype = 2, colour = "firebrick") +
  geom_hline(yintercept = loa[2], linetype = 1, colour = "steelblue") +
  geom_hline(yintercept = loa[3], linetype = 2, colour = "firebrick") +
  labs(x = "Mean of two instruments",
       y = "Difference (inst2 − inst1)")
```

14.41.8 4. Check

The ICC should be high (> 0.9) because the raters are well correlated, but the Bland–Altman plot shows a small positive bias (inst2 reads about 2 units higher on average).

14.41.9 5. Report

> Cohen's kappa for the two radiograph readers was `round(kappa2(kap_tbl[,c("r1","r2")])$value,2)`. For the two instruments, the ICC (absolute agreement, two-way random) was `round(ICC(as.matrix(meas))$results$ICC[2],2)`, but the Bland–Altman plot revealed a mean bias of `round(mean(ba$diff_val),1)` units with 95% limits of agreement from `round(loa[1],1)` to `round(loa[3],1)`.

14.41.10 Common pitfalls

- Reporting percent agreement instead of kappa.
- Using Pearson r on two raters and calling it agreement.
- Omitting the limits of agreement from a Bland–Altman plot.

14.41.11 Further reading

- Bland JM, Altman DG (1986), *Statistical methods for assessing agreement…*

- Shrout PE, Fleiss JL (1979), *Intraclass correlations…*
- McGraw KO, Wong SP (1996), *Forming inferences about some ICCs.*

14.41.12 Session info

```
sessionInfo()
```

14.41.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

14.42 Week 4, Session 5 — TRIPOD-AI, fairness auditing, reproducibility at scale

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

14.42.1 Learning objectives

- Enumerate the TRIPOD-AI reporting items relevant to a prediction-model manuscript.
- Compute group-stratified AUC and calibration as a fairness audit.
- Sketch a reproducible analysis pipeline with the `targets` package.

14.42.2 Prerequisites

External validation; biomarker evaluation.

14.42.3 Background

TRIPOD-AI extends the original TRIPOD statement to cover machine- learning prediction models. It asks authors to describe the data source, the participants, the outcome, the predictors, sample size and missing data, the model specification and its hyperparameter tuning, the performance on internal and external data, and the intended use of the model. A report that fails on any of these items is difficult to reproduce and difficult to deploy safely.

Fairness auditing extends validation to population subgroups. A model with strong overall AUC can have markedly worse performance in a minority subgroup; the remedy is first to detect the gap and then to decide whether to retrain, reweight, or accept the limitation explicitly.

The `targets` package is the modern R approach to reproducible pipelines. It builds a directed acyclic graph of analysis steps, caches intermediate outputs,

and reruns only what has changed. This separation between pipeline definition and execution is what lets a study survive the months between submission and revision.

Reproducibility at scale is not a purity test. It is an insurance policy: when a reviewer asks for a recomputed sensitivity, or when a colleague tries to replicate the analysis two years later, the cost of doing the work as a scripted DAG is paid back many times.

14.42.4 Setup

```
library(tidyverse)
library(pROC)
library(MASS)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

14.42.5 1. Goal

Audit a logistic prediction model on `Pima.tr` by a simulated subgroup attribute, and sketch a `targets` pipeline for the full analysis.

14.42.6 2. Approach

Attach a synthetic subgroup label — imagine this were clinic of enrolment — and compare performance.

```
d <- as_tibble(MASS::Pima.tr) |>
  mutate(subgroup = sample(c("A", "B"), n(), replace = TRUE,
                           prob = c(0.7, 0.3)))
ggplot(d, aes(glu, fill = subgroup)) +
  geom_histogram(alpha = 0.7, bins = 20, position = "identity")
```

14.42.7 3. Execution

```
fit <- glm(type ~ glu + bmi + age, data = d, family = binomial())
d$p <- predict(fit, type = "response")

auc_overall <- as.numeric(auc(roc(d$type, d$p, quiet = TRUE)))
auc_by <- d |>
  group_by(subgroup) |>
  summarise(auc = as.numeric(auc(roc(type, p, quiet = TRUE))),
            n   = n(), .groups = "drop")
auc_by
```

Calibration stratified by subgroup.

```
d |>
  mutate(bin = cut(p, quantile(p, seq(0, 1, by = 0.2)),
                   include.lowest = TRUE)) |>
  group_by(subgroup, bin) |>
  summarise(pred = mean(p), obs = mean(type == "Yes"),
            n = n(), .groups = "drop") |>
  ggplot(aes(pred, obs, colour = subgroup)) +
  geom_point(aes(size = n)) + geom_line() +
  geom_abline(slope = 1, intercept = 0, colour = "grey50") +
  labs(x = "mean predicted", y = "observed proportion")
```

A minimal `targets` pipeline (sketch).

```
library(targets)
tar_script({
  library(tidyverse); library(MASS); library(pROC)
  list(
    tar_target(raw, as_tibble(MASS::Pima.tr)),
    tar_target(fit, glm(type ~ glu + bmi + age, data = raw, family =
     ↪  binomial())),
    tar_target(auc_overall,
               as.numeric(auc(roc(raw$type, predict(fit, type =
                ↪  "response"), quiet = TRUE)))),
    tar_target(report, tibble(auc = auc_overall))
  )
})
tar_make()
tar_read(report)
```

14.42.8 4. Check

TRIPOD-AI-style checklist (abbreviated).

```
checklist <- tribble(
  ~item,                          ~status,
  "Study design stated",          "yes",
  "Source and eligibility",       "yes",
  "Outcome definition",           "yes",
  "Predictor definitions",        "yes",
  "Sample size justified",        "partial",
  "Missing-data handling",        "yes",
  "Model specification",          "yes",
  "Hyperparameter tuning",        "NA (no tuning)",
  "Internal validation",          "yes",
  "External validation",          "NOT in this lab",
  "Calibration reported",         "yes",
```

```
  "Fairness audit by subgroup",     "yes",
  "Code available",                 "yes"
)
checklist
```

14.42.9 5. Report

> A logistic prediction model on `Pima.tr` achieved overall AUC `round(auc_overall,2)`. A fairness audit by synthetic subgroup revealed AUCs of `round(auc_by$auc[1],2)` in subgroup A (n = `auc_by$n[1]`) and `round(auc_by$auc[2],2)` in subgroup B (n = `auc_by$n[2]`). A `targets` pipeline capturing raw data, fit, evaluation, and report would make the entire analysis re-runnable by any collaborator.

TRIPOD-AI, fairness auditing, and a pipeline tool are not independent initiatives; they are three faces of the same commitment to make modelling decisions legible, auditable, and reproducible.

14.42.10 Common pitfalls

- Reporting overall metrics and stopping; fairness gaps are only visible after stratification.
- Using `targets` as a static pipeline and not updating the DAG when inputs change.
- Treating TRIPOD-AI as a post-hoc checklist rather than a planning document written before analysis.

14.42.11 Further reading

- Collins GS et al. (2024), *TRIPOD+AI statement: updated guidance for reporting clinical prediction models that use regression or machine learning methods.*
- Obermeyer Z et al. (2019), *Dissecting racial bias in an algorithm used to manage the health of populations.*
- Landau WM (2021), *The targets R package: a dynamic make-like function-oriented pipeline toolkit.*

14.42.12 Session info

```
sessionInfo()
```

14.42.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 15

Meta-Analysis

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Pairwise random- and fixed-effects meta-analysis, network meta-analysis, individual-patient-data meta-analysis, and PRISMA-aligned reporting. Heterogeneity is treated as the main result, not a nuisance.

This chapter contains **25 method pages** and **3 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

15.1 Method pages

Method	Source slug
Bivariate SROC Curves	`bivariate-sroc`
Converting Between Effect Sizes	`effect-size-conversion`
Cumulative Meta-Analysis	`cumulative-meta-analysis`
Egger's Test for Funnel Asymmetry	`egger-test`
Fixed-Effect vs. Random-Effects Meta-Analysis	`fixed-vs-random-effects`
Forest Plots in Meta-Analysis	`forest-plot-meta`
Funnel Plots	`funnel-plot-meta`
Individual Participant Data Meta-Analysis	`ipd-meta-analysis`
Leave-One-Out Meta-Analysis	`leave-one-out`
Meta-Analysis of Diagnostic Accuracy	`diagnostic-test-meta`
Meta-Regression	`meta-regression`
Network Meta-Analysis: Introduction	`network-meta-analysis-intro`
NMA Consistency and Inconsistency	`nma-consistency`
NMA League Tables	`nma-league-tables`

Method	Source slug
Pooling Correlation Coefficients	`correlation-pooling`
Pooling Log-Odds Ratios	`log-odds-ratio-pooling`
Pooling Risk Ratios	`risk-ratio-pooling`
Prediction Intervals in Meta-Analysis	`prediction-interval-meta`
Quantifying Heterogeneity: Q and I^2	`heterogeneity-q-i2`
Selection Models for Publication Bias	`selection-models-bias`
Standardised Mean Difference: Hedges' g	`smd-hedges-g`
Subgroup Meta-Analysis	`subgroup-meta-analysis`
SUCRA Rankings	`nma-sucra`
Tau-Squared Estimation	`tau-squared-estimation`
Trim-and-Fill	`trim-and-fill`

15.2 Labs

Lab
Systematic reviews and PRISMA
Meta-analysis basics
Network meta-analysis

15.3 Bivariate SROC Curves

15.3.1 Introduction

The summary ROC (SROC) curve visualises the relationship between pooled sensitivity and specificity across a meta-analysis of diagnostic studies. Modern SROCs are derived from bivariate or HSROC models that properly account for the correlation between sensitivity and specificity as thresholds vary.

15.3.2 Prerequisites

Sensitivity and specificity; bivariate meta-analysis.

15.3.3 Theory

The bivariate model fits

$$\begin{pmatrix} \operatorname{logit}(\mathrm{Sens}_i) \\ \operatorname{logit}(1-\mathrm{Spec}_i) \end{pmatrix} \sim N(\mu, \Sigma).$$

The SROC curve is derived by mapping a range of the logit(1 - Spec) axis back to expected logit(Sens), given the bivariate mean and covariance. The curve shows how sensitivity trades against specificity as threshold varies.

15.3.4 Assumptions

Logit sensitivity and specificity are bivariate normal; studies vary by threshold; between-study heterogeneity captured by the covariance matrix.

15.3.5 R Implementation

```r
library(mada)

set.seed(2026)
df <- data.frame(
  TP = c(45, 60, 32, 80, 28, 55, 70, 50),
  FN = c( 5, 10,  8, 20, 12,  5,  8, 15),
  FP = c(10, 15,  8, 20, 10, 12, 18,  9),
  TN = c(140, 165, 152, 180, 150, 138, 200, 175)
)

fit <- reitsma(df)

# Summary statistics
summary(fit)$coefficients[c("tsens", "tfpr"), ]

# SROC plot
plot(fit, sroclty = 1, sroclwd = 2, predict = TRUE, predlty = 2,
     main = "Bivariate SROC with 95% confidence and prediction regions")
points(fpr(df), sens(df), cex = 0.8, pch = 19, col = "grey40")
```

15.3.6 Output & Results

Pooled sensitivity and FPR estimates; SROC curve with confidence (mean) and prediction (future-study) regions; individual study points overlaid.

15.3.7 Interpretation

"The SROC curve had AUC = 0.94, with pooled sensitivity 0.84 and specificity 0.91 at the summary operating point. Individual studies clustered tightly along the curve, consistent with threshold variation rather than diagnostic accuracy variation."

15.3.8 Practical Tips

- Always display both confidence (mean) and prediction (future-study) regions on the SROC.
- Individual study points should lie close to the curve; large off-curve outliers indicate effect modifiers.

- Do not average sensitivity and specificity independently – their correlation matters.
- For partial-threshold studies (e.g., only low-Sens/high-Spec), restrict the SROC to the covered threshold range.
- Report per PRISMA-DTA with a clear operating-point recommendation.

15.3.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.3.10 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.4 Pooling Correlation Coefficients

15.4.1 Introduction

Pearson correlation coefficients are bounded to $[-1, 1]$, and their sampling distributions are skewed with heteroscedastic variance — the variance depends on the underlying ρ and on n. These properties make raw correlations unsuitable for inverse-variance meta-analysis. Fisher's z transformation converts each correlation to an approximately Normal scale with variance independent of ρ, enabling standard random-effects pooling. After pooling on the z scale, the result is back-transformed to a correlation for interpretation.

15.4.2 Prerequisites

A working understanding of Pearson correlation, variance-stabilising transformations, and inverse-variance meta-analysis under random effects.

15.4.3 Theory

The Fisher z transformation is

$$z = \tfrac{1}{2}\log\!\left(\frac{1+r}{1-r}\right) = \tanh^{-1}(r), \qquad \mathrm{Var}(z) \approx \frac{1}{n-3}.$$

The variance is constant (depends only on n, not ρ), so the inverse-variance weighting in meta-analysis is well-defined. Pool z values across studies via fixed-effect or random-effects meta-analysis, then back-transform $r = \tanh(z)$ for reporting on the natural correlation scale.

The back-transformation preserves coverage at the original scale provided the transformed pooling is correctly weighted; reporting both z (with SE) and r (with CI) is standard.

15.4.4 Assumptions

Bivariate-Normal data in each study (the basis of the variance formula); independent studies; sufficient sample size per study ($n > 10$ as a rough minimum) for the asymptotic distribution to be reasonable.

15.4.5 R Implementation

```r
library(metafor)

set.seed(2026)
k <- 10
# Simulated correlations and sample sizes
df <- data.frame(
  ri = c(0.25, 0.32, 0.18, 0.40, 0.28, 0.35, 0.22, 0.30, 0.45, 0.38),
  ni = sample(50:300, k, replace = TRUE)
)

es <- escalc(measure = "ZCOR", ri = ri, ni = ni, data = df)
re <- rma(yi, vi, data = es, method = "REML")

# Convert pooled z back to correlation
pooled_z  <- re$b
pooled_r  <- tanh(pooled_z)
ci_r      <- tanh(c(re$ci.lb, re$ci.ub))

c(pooled_r = round(pooled_r, 3),
  lower = round(ci_r[1], 3), upper = round(ci_r[2], 3))
re$I2
```

15.4.6 Output & Results

`escalc(measure="ZCOR")` computes Fisher z and its variance per study; `rma()` fits the random-effects model on the z scale. The pooled correlation is approximately 0.31 with a tight 95 % CI; I^2 quantifies heterogeneity.

15.4.7 Interpretation

A reporting sentence: "Random-effects meta-analysis of 10 studies (REML, n total = 1{,}523) gave a pooled correlation of $r = 0.31$ (95 % CI 0.25 to 0.37) with modest heterogeneity ($I^2 = 28\%$, $\tau^2 = 0.012$); pooling on the Fisher z scale and back-transforming maintains correct coverage despite the $[-1, 1]$ bound on r." Always pool on the z scale and report both Fisher z and back-transformed r.

15.4.8 Practical Tips

- Always pool on the Fisher z scale; pooling raw r biases estimates for high correlations and produces incorrect coverage.
- `metafor::escalc(measure="ZCOR")` handles the Fisher z transformation and variance computation in a single call.
- Report both z (with SE) and back-transformed r (with CI); r is more interpretable for substantive readers.
- For partial correlations, apply the same approach with the degrees-of-freedom-adjusted sample size $n-p$ where p is the number of partialled-out variables.
- Hartung-Knapp small-sample adjustment (`rma(...,test="knha")`) is especially important when studies are few or unevenly sized; standard inverse-variance gives over-confident intervals in those settings.
- For meta-analysis of intraclass correlations or other correlation-like statistics, the same Fisher z framework applies with appropriate variance formulae.

15.4.9 R Packages Used

`metafor` for `escalc()`, `rma()`, and the canonical correlation-pooling workflow; `meta` for an alternative interface with cleaner forest-plot output; `psychmeta` for psychometric meta-analysis with Hunter-Schmidt artefact corrections; `clubSandwich` for robust variance estimation in correlated-effect meta-analyses.

15.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.4.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics

- Network meta-analysis

15.5 Cumulative Meta-Analysis

15.5.1 Introduction

Cumulative meta-analysis plots the running pooled estimate as studies are added one by one in chronological (or any pre-specified) order. Each successive pool incorporates one more study, producing a sequence of estimates that traces how the body of evidence evolved over time. The resulting forest plot reveals when the cumulative effect first crossed a meaningful threshold — clinical relevance, statistical significance, or stability — and whether later studies overturned, attenuated, or simply added precision to the early consensus. Cumulative meta-analysis is widely used in retrospective evidence syntheses to assess research efficiency: if the evidence had been pooled in real time, when would the answer have been clear, and how many subsequent confirmatory trials were arguably redundant?

15.5.2 Prerequisites

A working understanding of inverse-variance meta-analytic pooling, the sequential-analysis idea, and an awareness of why repeated looks at accumulating evidence raise multiplicity concerns when the goal is decision-making rather than description.

15.5.3 Theory

For an ordering of studies $1, \dots, k$, the cumulative estimate at step t is the pooled effect computed from studies $1, \dots, t$ under the chosen meta-analysis model. Plotting these $\widehat{\theta}_{(1:t)}$ with their confidence intervals along the ordering axis (typically year) reveals stability and convergence: a horizontal band of overlapping CIs indicates the answer settled early, while a shifting estimate signals that later evidence materially changed conclusions.

The same procedure can use precision-based ordering (largest studies first) to assess sensitivity to the small-study effect, or risk-of-bias ordering (low-risk studies first) to assess whether pooled conclusions hinge on lower-quality evidence.

15.5.4 Assumptions

Study ordering is correctly recorded; studies are independent; the meta-analysis model used at each step is the same as for the full pool. Cumulative meta-analysis is descriptive, not inferential — formal sequential-testing methods such as trial-sequential analysis adjust for repeated looks; ordinary cumulative meta-analysis does not.

15.5.5 R Implementation

```r
library(metafor)

set.seed(2026)
k <- 12
year <- sort(sample(1990:2023, k))
yi <- rnorm(k, 0.35, 0.2)
vi <- 1 / sample(50:250, k, replace = TRUE)

df <- data.frame(year, yi, vi)

res <- rma(yi, vi, data = df, method = "REML")
cum <- cumul(res, order = df$year)

plot(cum, main = "Cumulative meta-analysis by year",
     xlab = "Year")
```

15.5.6 Output & Results

`cumul()` returns a running-pooled estimate, confidence interval, and heterogeneity statistics at each step in the specified ordering. The associated `plot()` method renders a forest-style display where each row corresponds to "evidence after the t-th study", making convergence visually obvious. Many pooled effects stabilise after four to six well-powered studies; remaining studies typically tighten the CI without shifting the centre.

15.5.7 Interpretation

A reporting sentence: "Cumulative meta-analysis ordered by publication year showed that the pooled effect became statistically significant after the third study (1994) and remained essentially unchanged through 2023; the 18 trials published thereafter added precision but did not alter the direction or magnitude of the conclusion, suggesting an opportunity for earlier evidence-based decision-making had a living synthesis been in place." Always describe both when the evidence stabilised and what later studies contributed.

15.5.8 Practical Tips

- Order primarily by publication date for the temporal interpretation; precision-based and risk-of-bias orderings are useful complementary sensitivity analyses.
- Cumulative meta-analysis is descriptive — for inferential stopping rules under repeated looks, use trial-sequential analysis (TSA) with appropriate alpha-spending.

- Convergence to stability after the first few studies often reveals research waste: many subsequent trials confirmed what was already known.
- Combine with meta-regression on year to formally test whether effect sizes drifted with time, e.g. as practice patterns or populations changed.
- Plot both point estimate and confidence interval; a shrinking CI without estimate movement is a different signal from a moving estimate.
- Use cumulative meta-analysis to motivate living-systematic-review infrastructure: continuous evidence updating prevents the "many years late" problem that retrospective cumulative plots so often expose.

15.5.9 R Packages Used

`metafor::cumul()` for the canonical cumulative pooling and its `plot()` method; `meta::metacum()` for an alternative interface aligned with the `meta` package's forest-plot conventions; `RTSA` and `gsDesign` for trial-sequential analysis when inferential adjustment for multiple looks is required; `metaviz` for richer cumulative-forest visualisations.

15.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.5.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.6 Meta-Analysis of Diagnostic Accuracy

15.6.1 Introduction

Meta-analysis of diagnostic test accuracy pools per-study sensitivity and specificity to summarise how well a test discriminates disease from non-disease across multiple primary studies. Unlike meta-analysis of treatment effects, the two paired outcomes are intrinsically correlated: sensitivity and specificity are determined jointly by the test threshold, so a study choosing a stricter cut-off will trade higher specificity for lower sensitivity. Naively pooling each metric separately ignores this trade-off and produces estimates that misrepresent the test's underlying performance. The modern approach is bivariate random-effects modelling (Reitsma et al., 2005) or hierarchical summary ROC modelling (Rutter

and Gatsonis, 2001), both of which respect the correlation structure and yield interpretable summary points and curves.

15.6.2 Prerequisites

A working understanding of sensitivity, specificity, and the receiver-operating characteristic (ROC) curve; familiarity with the logit transformation; and basic mixed-effects modelling concepts.

15.6.3 Theory

The bivariate random-effects model jointly fits logit-sensitivity and logit-specificity using a bivariate Normal random effect across studies, estimating pooled sensitivity, pooled specificity, between-study variances for each, and the correlation between them. The HSROC model parameterises the underlying ROC curve directly with random study-specific thresholds and accuracy parameters, producing a summary curve that explicitly accommodates threshold variation. Reitsma's bivariate model and the HSROC model are mathematically equivalent under standard parameterisation; the choice between them is mainly one of presentation — pooled point with confidence ellipse versus summary curve.

15.6.4 Assumptions

Each study contributes a 2×2 table of true positives, false negatives, false positives, and true negatives. Random effects on the logit scale are bivariate Normal; threshold variation across studies is captured implicitly by the random-effects structure; case ascertainment and reference-standard verification are comparable across studies (otherwise applicability concerns enter, captured by the QUADAS-2 framework).

15.6.5 R Implementation

```
library(mada)

set.seed(2026)
df <- data.frame(
  TP = c(45, 60, 32, 80, 28, 55),
  FN = c(5, 10, 8, 20, 12, 5),
  FP = c(10, 15, 8, 20, 10, 12),
  TN = c(140, 165, 152, 180, 150, 138)
)

fit <- reitsma(df)
summary(fit)
```

```r
plot(fit, sroclty = 1, sroclwd = 2,
     main = "Summary ROC curve")
```

15.6.6 Output & Results

`reitsma()` returns pooled logit-sensitivity and logit-specificity with their variances and the between-study correlation; back-transformation gives the pooled point estimates with confidence intervals and a confidence ellipse on the ROC plane. The summary ROC curve traces the implied trade-off across thresholds, and a summary AUC quantifies discrimination across the full operating range.

15.6.7 Interpretation

A reporting sentence: "Bivariate meta-analysis of six diagnostic-accuracy studies (Reitsma model, REML) pooled sensitivity 0.84 (95 % CI 0.78 to 0.89) and specificity 0.91 (95 % CI 0.87 to 0.94) on the logit scale; the summary ROC area under the curve was 0.94. The between-study correlation −0.3 indicates threshold variation contributes meaningfully to between-study heterogeneity, which the bivariate model accommodates explicitly. Reporting follows PRISMA-DTA." Always report both summary points and the SROC curve, plus the between-study correlation.

15.6.8 Practical Tips

- Always use bivariate or HSROC models; separate univariate pooling of sensitivity and specificity is statistically incorrect because it ignores the threshold-induced correlation between the two.
- Report the pooled point estimate, the 95 % confidence ellipse on the ROC plane, and the summary curve — together they convey both central performance and uncertainty.
- Threshold variability is a major source of heterogeneity in DTA meta-analyses; subgroup analyses by pre-specified threshold or by clinical setting are often more interpretable than a single overall pool.
- For studies with zero counts in any cell, `mada` applies a 0.5 continuity correction by default; sparse-data settings may benefit from Bayesian DTA-NMA models in `bamdit` or `INLA`.
- PRISMA-DTA is the reporting standard for diagnostic-accuracy systematic reviews; QUADAS-2 is the standard risk-of-bias assessment tool for individual primary studies.
- For diagnostic-accuracy NMA — comparing multiple competing tests on the same disease — `BUGSnet`, `bamdit`, and recent extensions in `multinma` provide the appropriate framework.

15.6.9 R Packages Used

`mada::reitsma()` for the canonical bivariate random-effects model and `mada` for SROC plotting and AUC; `mada::HSROC()` for the equivalent HSROC parameterisation; `bamdit` for Bayesian DTA meta-analysis with rich diagnostics; `meta::metaprop()` for univariate proportion pooling when only one of sensitivity or specificity is of interest; `multinma` for Bayesian DTA-NMA across multiple competing tests.

15.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.6.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.7 Converting Between Effect Sizes

15.7.1 Introduction

A single meta-analysis often spans studies that report effect sizes in different metrics: some randomised trials report standardised mean differences (Cohen's d or Hedges's g), some observational analyses report correlations (r), and some binary-outcome studies report odds ratios (OR) or risk ratios (RR). Pooling such heterogeneous evidence requires converting every per-study effect to a common metric — typically d when most studies are continuous, OR when most are binary, or r when most are correlational. Analytical conversion formulas, derived under standard latent-distribution assumptions, map between metrics together with their variances. Doing the conversion correctly — and documenting the assumptions — is critical because naively recycling an effect size between scales without proper variance propagation produces falsely precise pooled estimates.

15.7.2 Prerequisites

A working understanding of the principal effect-size metrics (d, r, OR, RR), their variance formulas, and the delta method for propagating variance through non-linear transformations.

15.7.3 Theory

The standard conversion formulas, all derived under the assumption that the underlying latent variable is approximately Normal, are:

$$d \to r: \quad r = \frac{d}{\sqrt{d^2 + a}}, \qquad a = \frac{(n_1 + n_2)^2}{n_1 n_2};$$

$$\log \mathrm{OR} \to d: \quad d = \log(\mathrm{OR}) \cdot \frac{\sqrt{3}}{\pi};$$

$$r \to d: \quad d = \frac{2r}{\sqrt{1 - r^2}}.$$

Each conversion has a corresponding variance formula obtained by the delta method; using these (rather than the un-transformed variance) is essential, because the non-linear mapping changes the precision of the resulting estimate.

15.7.4 Assumptions

The underlying latent distributions are approximately Normal (for $d \leftrightarrow r$ conversions) or follow the standard logistic-Normal correspondence (for OR conversions), the dichotomisation underlying any reported OR was approximately at the median of the latent variable, and the variance is propagated through the conversion using the appropriate formula rather than the original metric's variance.

15.7.5 R Implementation

```
library(esc)

set.seed(2026)
d <- 0.5; var_d <- 0.04; n1 <- 50; n2 <- 50
conv_dr <- esc::convert_d2r(d = d, v = var_d,
                            grp1n = n1, grp2n = n2)
conv_dr

log_or <- log(2.5); var_logor <- 0.15
conv_od <- esc::convert_or2d(or = exp(log_or), v = var_logor)
conv_od

r <- 0.3; var_r <- 0.004
conv_rd <- esc::convert_r2d(r = r, v = var_r)
conv_rd
```

15.7.6 Output & Results

The `esc` package returns a list with the converted point estimate, its delta-method variance, and the corresponding standard error and confidence interval. Each conversion preserves both the central effect and a properly propagated variance, allowing inverse-variance pooling on the harmonised metric without artificially inflating precision.

15.7.7 Interpretation

A reporting sentence: "Effect sizes were harmonised to standardised mean differences. Continuous-outcome studies contributed $d = 0.50$ directly; binary-outcome studies were converted from OR $= 2.5$ to $d = 0.51$ using $d = \log(\text{OR}) \cdot \sqrt{3}/\pi$; correlational studies were converted from $r = 0.30$ to $d = 0.63$ using $d = 2r/\sqrt{1 - r^2}$. Variances were propagated by the delta method. The common metric enabled inverse-variance pooling across study designs and is documented in PRISMA Supplementary Table S2." Always document every conversion explicitly.

15.7.8 Practical Tips

- Use validated conversion functions from `esc`, `compute.es`, or `metafor::escalc()`; manual arithmetic with these formulas is highly error-prone, especially for variance propagation.
- Always propagate variance through the conversion using the delta-method-derived formulas; using the original metric's variance produces falsely narrow confidence intervals on the converted scale.
- Converting between binary and continuous metrics assumes a logistic-Normal latent-variable structure; for outcomes that obviously violate this — categorical with more than two levels treated as binary, very skewed continuous variables — the conversion is approximate at best.
- For highly skewed continuous outcomes (e.g., counts, costs, biomarkers with long tails), d-based conversions may misrepresent the substantive effect; consider rank-based effect sizes or per-domain pooling instead.
- Document every conversion in a PRISMA supplementary table, including the formula used, the input quantities, the converted estimate, and its propagated variance — methodological reviewers routinely check this.
- When the meta-analysis is mostly one metric with a few studies in another, conversion is appropriate; when the metrics are evenly split, consider whether a combined-metric analysis (e.g., a generalised linear mixed model on the underlying outcome) might be more defensible.

15.7.9 R Packages Used

`esc` for conversion functions across d, r, OR, and other effect sizes with delta-method variance; `compute.es` for an alternative comprehensive conversion li-

brary; `metafor::escalc()` and `metafor::conv.delta()` for canonical conversions and general delta-method variance propagation; `effectsize` for tidyverse-style effect-size computation and conversion alongside primary analyses.

15.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.7.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.8 Egger's Test for Funnel Asymmetry

15.8.1 Introduction

Egger's regression test, introduced by Egger and colleagues in 1997, provides a formal hypothesis test for funnel-plot asymmetry — the most commonly cited indirect symptom of publication bias and small-study effects. Where the funnel plot offers a visual diagnostic that depends on the eye of the beholder, Egger's test reduces the question to a regression intercept whose distribution under symmetry is known. A statistically significant intercept indicates that small studies systematically report effects of different magnitude than larger studies, which may reflect selective publication, methodological heterogeneity confounded with sample size, or genuine small-study heterogeneity. Although interpretation requires nuance, Egger's test remains the de facto standard quantitative complement to a funnel plot and is recommended by the Cochrane Handbook for meta-analyses with at least ten studies.

15.8.2 Prerequisites

Familiarity with funnel plots, the small-study-effect concept, and ordinary least-squares regression with weighted variants.

15.8.3 Theory

The test regresses the standardised effect $y_i/\mathrm{SE}(y_i)$ on the precision $1/\mathrm{SE}(y_i)$. Under the null hypothesis of funnel symmetry the regression line passes through the origin; the intercept tests this directly. A non-zero intercept indicates that

the standardised effect depends on precision — exactly the pattern produced when small, imprecise studies are over-represented at one tail of the funnel because of selective publication.

Statistical power is low with fewer than ten studies, and inflated when between-study heterogeneity is large; $k \geq 10$ is the conventional minimum for interpretable use, and the test should never be the sole basis for declaring publication bias.

15.8.4 Assumptions

Studies are independent; the effect estimate is approximately normally distributed on the analysis scale; the relationship between standardised effect and precision is linear under the null; and the meta-analysis contains enough studies that any asymmetry is detectable above the random-sampling noise. For binary outcomes the standard Egger test is biased — Harbord's and Peters's modifications are preferred.

15.8.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 20
yi <- rnorm(k, 0.4, 0.2)
vi <- 1 / sample(20:300, k, replace = TRUE)

keep <- !(sqrt(vi) > 0.1 & abs(yi) < 0.2)
yi_b <- yi[keep]; vi_b <- vi[keep]

res <- rma(yi_b, vi_b, method = "REML")

egger <- regtest(res, model = "lm")
print(egger)

fsn(yi_b, vi_b)
```

15.8.6 Output & Results

`regtest()` returns the regression intercept, its standard error, the test statistic, and the two-sided p-value for the null of funnel symmetry. The companion fail-safe-N statistic from `fsn()` reports how many additional null-effect studies would be needed to drive the pooled estimate to non-significance — a complementary, intuitive sensitivity measure that helps readers gauge robustness.

15.8.7 Interpretation

A reporting sentence: "Egger's regression suggested significant funnel asymmetry (intercept 1.28, 95 % CI 0.21 to 2.35, $p = 0.02$), consistent with small-study effects or publication bias. A fail-safe N of 42 indicates the pooled conclusion would be diluted to non-significance only by 42 unpublished null studies, suggesting moderate but not unlimited robustness; trim-and-fill and PET-PEESE sensitivity analyses are reported in the supplement." Always pair the test result with at least one bias-adjusted sensitivity analysis.

15.8.8 Practical Tips

- Egger's test has low power for $k < 10$; the Cochrane Handbook explicitly discourages its use below this threshold and recommends visual funnel inspection only.
- For binary outcomes, prefer Harbord's modified test (continuous data) or Peters's test (rare events); the standard Egger test inflates the type-I error rate when the effect measure is a log-OR or log-RR.
- A significant intercept is not synonymous with publication bias — true between-study heterogeneity correlated with study size (small studies systematically using different populations or methods) can produce the same signal.
- Sensitivity-analyse with trim-and-fill and PET-PEESE; both adjust the pooled estimate for apparent asymmetry and should be reported alongside the unadjusted estimate when Egger's test is significant.
- For low-prevalence binary outcomes, asymmetry can arise from continuity-correction artefacts rather than genuine bias; Mantel–Haenszel without correction is a useful diagnostic alternative.
- Report the intercept estimate with its CI, not only its p-value; the magnitude of asymmetry conveys more than dichotomous significance.

15.8.9 R Packages Used

`metafor::regtest()` for Egger's, Harbord's, Peters's, and the more general regression-based asymmetry tests; `metafor::fsn()` for fail-safe N across multiple definitions (Rosenthal, Orwin, Rosenberg); `meta::metabias()` as the alternative interface with built-in plotting; `puniform` and `weightr` for selection-model-based bias adjustment; `metasens` for trim-and-fill and PET-PEESE workflows.

15.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.8.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.9 Fixed-Effect vs. Random-Effects Meta-Analysis

15.9.1 Introduction

A meta-analysis pools effect-size estimates from multiple studies into a single summary. The two classical approaches – the fixed-effect model and the random-effects model – rest on different assumptions about what those studies represent. Choosing between them is not a technical afterthought; it is a substantive decision about the scope of inference, and it can change both the point estimate and its uncertainty.

15.9.2 Prerequisites

The reader should know what an effect size is (mean difference, standardised mean difference, log-odds ratio, etc.), how its standard error is computed, and how a weighted average works. Familiarity with the `meta` or `metafor` packages is helpful.

15.9.3 Theory

Let $\hat{\theta}_i$ be the effect-size estimate from study i with variance v_i, for $i = 1, \dots, k$.

15.9.3.1 Fixed-effect model

Assumes all studies estimate the *same* true effect θ:

$$\hat{\theta}_i = \theta + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, v_i).$$

The optimal pooled estimate is the inverse-variance-weighted mean

$$\hat{\theta}_{\text{FE}} = \frac{\sum w_i \hat{\theta}_i}{\sum w_i}, \qquad w_i = 1/v_i,$$

with standard error $1/\sqrt{\sum w_i}$. Scope of inference: *the set of studies included.*

15.9.3.2 Random-effects model

Assumes each study estimates its own study-specific true effect, drawn from a distribution of true effects across a population of plausible studies:

$$\hat{\theta}_i = \theta_i + \varepsilon_i, \qquad \theta_i \sim \mathcal{N}(\mu, \tau^2), \qquad \varepsilon_i \sim \mathcal{N}(0, v_i).$$

The weights become $w_i^* = 1/(v_i + \tau^2)$, where τ^2 is the between-study variance, estimated by DerSimonian-Laird, REML, Paule-Mandel, or Sidik-Jonkman. The pooled estimate is $\hat{\mu} = \sum w_i^* \hat{\theta}_i / \sum w_i^*$. Scope of inference: *the population of studies from which the included ones are a sample.*

When $\tau^2 = 0$ (no heterogeneity), the random-effects model reduces to the fixed-effect model. When τ^2 is large, random-effects weights are more balanced (small studies get relatively more weight), and the random-effects CI is wider than the fixed-effect CI – properly so, since it reflects between-study variability.

15.9.4 Assumptions

Both models assume:

- Study-level effect estimates are approximately normal with known variance.
- Studies are independent of each other.
- No publication bias or other selection distorting the set of included studies.

The fixed-effect model additionally assumes a single true effect. The random-effects model assumes a normal distribution of true effects; the Hartung-Knapp adjustment corrects CIs under small k or imprecise τ^2 estimates.

15.9.5 R Implementation

```r
library(meta)

dat <- data.frame(
  study = paste("Study", 1:8),
  mean_t = c(11.2, 9.8, 12.5, 10.1, 13.0, 8.7, 11.6, 10.9),
  sd_t   = c(2.1, 2.4, 2.0, 2.6, 1.9, 2.8, 2.2, 2.3),
  n_t    = c(40, 35, 50, 30, 45, 25, 38, 42),
  mean_c = c(9.8, 9.5, 10.1, 9.9, 10.5, 9.0, 10.0, 10.2),
  sd_c   = c(2.0, 2.3, 2.1, 2.5, 2.0, 2.7, 2.1, 2.2),
  n_c    = c(40, 35, 50, 30, 45, 25, 38, 42)
)

m <- metacont(n.e = n_t, mean.e = mean_t, sd.e = sd_t,
              n.c = n_c, mean.c = mean_c, sd.c = sd_c,
              studlab = study, data = dat,
```

```
              sm = "MD",
              method.tau = "REML",
              method.random.ci = "HK")

summary(m)

forest(m, leftcols = c("studlab", "n.e", "mean.e", "sd.e",
                       "n.c", "mean.c", "sd.c"))
```

`metacont()` handles continuous outcomes with mean differences. Setting `method.tau="REML"` uses REML for τ^2; `method.random.ci="HK"` applies the Hartung-Knapp correction to the random-effects confidence interval.

15.9.6 Output & Results

Typical output:

```
                        MD            95%-CI %W(fixed) %W(random)
Fixed effect model  1.210 [ 0.802; 1.618]    100.0       --
Random effects model 1.195 [ 0.567; 1.822]      --    100.0

Quantifying heterogeneity:
 tau^2 = 0.3521; tau = 0.5934;
 I^2 = 52.8% [0.0%; 78.4%]; H = 1.46 [1.00; 2.15]

Test of heterogeneity:
     Q d.f.  p-value
 14.83    7   0.0381
```

The fixed-effect and random-effects point estimates are similar here (1.21 vs. 1.20), but the random-effects confidence interval is more than 50% wider, reflecting the moderate heterogeneity ($I^2 = 53\%$).

15.9.7 Interpretation

Report both models when I^2 is non-negligible, and emphasise the random-effects result with its Hartung-Knapp CI as the primary inference when heterogeneity is plausible on substantive grounds. Include the τ^2 estimate and the prediction interval – the latter answers "what true effect should I expect in a future, similar study?" and is often the most informative summary for a clinician.

15.9.8 Practical Tips

- Choose the model based on subject-matter plausibility, not on whether heterogeneity is statistically significant – Q-tests are underpowered with few studies.

- Report the prediction interval alongside the pooled estimate; it is the honest summary of what the pooled result implies for a new study.
- Use REML or Paule-Mandel for τ^2; DerSimonian-Laird is the historical default but understates τ^2 with few studies.
- Never pool fewer than 3 studies without explicit caveat; with 5 or fewer, consider narrative synthesis rather than meta-analysis.
- When heterogeneity is high, investigate it with subgroup analysis and meta-regression before pooling.

15.9.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.9.10 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.10 Forest Plots in Meta-Analysis

15.10.1 Introduction

The forest plot is the canonical meta-analysis figure and the single most important visual artefact a systematic review delivers. Each included study appears as a row with its point estimate, confidence interval, and pooling weight; the overall summary is rendered at the bottom as a diamond whose width represents the pooled confidence interval. This compact layout simultaneously conveys per-study effects, study-level precision, between-study scatter, and the overall conclusion — all in a glance, without needing to consult a separate table. Editorial standards in major clinical journals routinely require a forest plot for every meta-analytic synthesis, and reviewers rely on it to judge whether the pooled estimate fairly represents the constituent studies.

15.10.2 Prerequisites

Inverse-variance pooling, the distinction between fixed-effect and random-effects models, and an understanding of how meta-analytic weights translate study precision into influence on the summary estimate.

15.10.3 Theory

Each study's square is plotted at its point estimate with size proportional to its meta-analytic weight (inverse variance under fixed-effect, or DerSimonian–Laird / REML weights under random-effects). Horizontal whiskers extend to the 95 % confidence interval. The pooled estimate is a diamond centred at the summary, with horizontal extent equal to the pooled CI; under random-effects models, an additional prediction interval is often overlaid as a dashed bar to convey where the next study's true effect is plausibly expected.

Standard columns include: study label (often author + year), sample sizes per arm, point estimate with CI, and the percent weight contribution. Heterogeneity statistics (Q, I^2, τ^2) typically appear in the figure footer.

15.10.4 Assumptions

Study-level estimates and their variances are on the same scale, expressed in the same direction (so that "to the right of zero" or "to the right of one" has a consistent clinical meaning), and pooled under a stated model. Forest plots themselves do not impose extra assumptions — they visualise whatever model produced the inputs.

15.10.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 8
yi <- c(0.32, 0.51, 0.28, 0.45, 0.38, 0.22, 0.60, 0.35)
vi <- c(0.02, 0.03, 0.015, 0.025, 0.04, 0.018, 0.05, 0.022)
authors <- paste("Study", LETTERS[1:k])

res <- rma(yi, vi, method = "REML")

forest(res, slab = authors,
       header = c("Study", "SMD [95% CI]"),
       xlab = "Standardised mean difference (Hedges' g)",
       mlab = "Random-effects (REML)",
       cex = 0.9)
```

15.10.6 Output & Results

The `forest()` method renders study-wise rows with squares sized by random-effects weight, whiskers showing 95 % CIs, and a summary diamond at the bottom. The header and label arguments customise the axis label and column heads, and `addpred=TRUE` overlays the random-effects prediction interval as a

dashed bar around the diamond.

15.10.7 Interpretation

A reporting sentence: "The forest plot summarises eight randomised trials; seven favour the intervention with confidence intervals excluding the null, and the random-effects pooled SMD is 0.39 (95 % CI 0.28 to 0.50, $I^2 = 27\%$, $\tau^2 = 0.012$). The 95 % prediction interval (0.10 to 0.68) suggests the true effect in a future trial is likely positive but variable in magnitude." Always pair the forest plot with prose that quantifies both the central effect and the heterogeneity context.

15.10.8 Practical Tips

- Sort studies by publication year (for temporal reading), by effect size (for visual scanning of dispersion), or by subgroup (for pre-specified comparisons); the choice should match the scientific question.
- Always include heterogeneity statistics (I^2, τ^2, Q) in the figure footer; readers cannot judge "is this pool reasonable?" without them.
- Overlay the prediction interval (`addpred=TRUE` in `metafor`); a CI describes uncertainty about the mean, but the prediction interval is what readers actually want to know about future studies.
- For complex layouts — multiple subgroups, side-by-side outcomes, risk-of-bias columns — `forestplot`, `forester`, or `metafor::forest()` with custom `ilab` arguments give finer control than the default `meta::forest()`.
- Always include a clear directional legend ("favours intervention ← → favours control"); ambiguity here is the most common cause of reader misinterpretation.
- Round to a sensible number of digits (typically 2 for SMDs, 2 for log-ratios back-transformed); excessive precision suggests false confidence and clutters the plot.

15.10.9 R Packages Used

`metafor::forest()` for the canonical method, with `addpred`, `ilab`, and slab customisation; `meta::forest()` for the alternative `meta`-package interface with simpler defaults; `forestplot` and `forester` for highly customised layouts including categorical variables, multiple effects, and risk-of-bias columns; `metaviz` for ggplot-based forest plots with publication-ready theming.

15.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.10.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.11 Funnel Plots

15.11.1 Introduction

The funnel plot, introduced by Light and Pillemer in 1984, is the classical visual diagnostic for publication bias and small-study effects in meta-analysis. Each study contributes a single point whose horizontal coordinate is the effect estimate and whose vertical coordinate is precision (typically standard error or its inverse). Under the null of no bias and homogeneous true effect, the plot resembles a symmetric inverted funnel: large, precise studies cluster near the pooled estimate at the top and smaller, less precise studies fan out symmetrically below. Asymmetry — a visible empty zone in one of the two lower corners — is the canonical visual red flag that smaller studies with effects in one direction may be missing from the synthesis, often because they were never published.

15.11.2 Prerequisites

A working understanding of pooled meta-analysis, the relationship between sample size and standard error, and the small-study-effect concept that links study size to the magnitude of reported effects.

15.11.3 Theory

The horizontal axis carries the effect estimate on its analysis scale (log-OR, log-RR, SMD); the vertical axis carries precision, typically standard error inverted so that large, precise studies sit at the top. Pseudo-confidence-interval lines fan out symmetrically from the pooled estimate, giving the funnel its name. Under symmetry, small studies scatter on both sides; an empty lower corner suggests small studies with null or opposite-direction effects may be missing from the synthesis, the visual signature of publication bias.

A central conceptual point: asymmetry indicates a small-study effect, of which publication bias is one possible cause among several. Treating asymmetry as a synonym for bias is a common interpretive error.

15.11.4 Assumptions

Studies are independent; under the null of no bias the effect estimates scatter symmetrically around the true pooled effect; precision is well measured. The interpretation requires enough studies that asymmetry is detectable above sampling noise — typically at least ten.

15.11.5 R Implementation

```r
library(metafor)

set.seed(2026)
k <- 20
yi <- rnorm(k, 0.4, 0.2)
vi <- 1 / sample(20:300, k, replace = TRUE)

keep <- !(sqrt(vi) > 0.1 & abs(yi) < 0.2)
yi_biased <- yi[keep]; vi_biased <- vi[keep]

res_biased <- rma(yi_biased, vi_biased, method = "REML")
funnel(res_biased, main = "Funnel plot (potentially biased)")

regtest(res_biased, model = "lm")
```

15.11.6 Output & Results

`funnel()` renders the canonical scatter with pseudo-CI lines. Visual asymmetry — typically an empty lower corner — combined with a significant Egger or Harbord regression test on the same data quantifies the impression. Reporting both views together (visual plus formal test) is the standard approach because the two are complementary diagnostics rather than redundant.

15.11.7 Interpretation

A reporting sentence: "The funnel plot was visibly asymmetric with an empty region in the small-study, low-effect quadrant; Egger's regression test confirmed asymmetry (intercept 1.28, $p = 0.02$). Publication bias is a plausible explanation, but true heterogeneity correlated with study size cannot be ruled out; trim-and-fill and PET-PEESE sensitivity analyses are reported in the supplement and showed an attenuated but still positive pooled effect." Always describe both the visual pattern and the formal-test result, and acknowledge alternative explanations.

15.11.8 Practical Tips

- Funnel plots require at least ten studies for meaningful interpretation; with fewer studies, the visual is dominated by sampling noise and asymmetry impressions are unreliable.
- Asymmetry has many possible causes besides publication bias: true between-study heterogeneity correlated with size, selective outcome reporting, methodological quality differences, or chance with small k. Discuss each explicitly when reporting an asymmetric funnel.

- Combine visual inspection with at least one formal test: Egger's regression for continuous outcomes, Harbord's modification for binary outcomes (log-OR), and Peters's test for rare events.
- For binary outcomes, plot log(OR) versus standard error on the log scale; raw OR funnels are intrinsically asymmetric for mathematical reasons unrelated to bias.
- Include sensitivity analyses — trim-and-fill, PET-PEESE, or Vevea–Hedges weight-function selection models — when asymmetry is detected; the difference between adjusted and unadjusted estimates is what readers actually need.
- Contour-enhanced funnel plots (Peters et al., 2008) overlay significance contours and help distinguish missingness near the null (suggestive of publication bias) from missingness elsewhere (suggestive of other explanations).

15.11.9 R Packages Used

`metafor::funnel()` for the canonical and contour-enhanced funnel plot, with `regtest()` for Egger/Harbord/Peters tests; `meta::funnel()` as the alternative interface; `metafor::trimfill()` for trim-and-fill adjustment; `metasens` for limit-meta-analysis and Copas-model bias adjustment alongside the funnel; `metaviz` for ggplot-based publication-ready funnel-plot output.

15.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.11.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.12 Quantifying Heterogeneity: Q and I^2

15.12.1 Introduction

Heterogeneity in meta-analysis refers to the variation in true effects across studies that exceeds what would be expected from sampling variability alone. Cochran's Q test asks whether any heterogeneity is present at all; I^2 quantifies how much of the total observed variability is attributable to genuine

between-study differences rather than within-study sampling noise; and τ^2 provides the actual variance of true effects on the analysis scale. Together these three statistics inform the choice between fixed-effect and random-effects models, the width of prediction intervals, and the appropriateness of subgroup or meta-regression follow-up analyses. Reporting all three is now the universal standard in systematic-review publications.

15.12.2 Prerequisites

A working understanding of fixed-effect and random-effects meta-analytic pooling, the chi-squared distribution, and inverse-variance weighting.

15.12.3 Theory

Cochran's Q is the weighted sum of squared deviations of study effects from the fixed-effect pooled estimate:

$$Q = \sum_i w_i (\theta_i - \hat{\theta}_{\text{FE}})^2.$$

Under the null of homogeneity, Q follows a χ^2_{k-1} distribution. The I^2 statistic rescales Q to a proportion:

$$I^2 = \max\left\{0,\ \frac{Q - (k-1)}{Q}\right\} \cdot 100\%,$$

interpretable as the proportion of total variation attributable to between-study heterogeneity rather than within-study error. Higgins's benchmarks are: 0–40 % low, 30–60 % moderate, 50–90 % substantial, 75–100 % considerable. The variance component τ^2 — typically estimated by REML, DerSimonian–Laird, or Paule–Mandel — gives heterogeneity on the effect-size scale and feeds directly into prediction-interval computation.

15.12.4 Assumptions

Studies are independent and within-study variances are known (or precisely estimated). The Q test is famously underpowered for small numbers of studies and famously over-powered for large meta-analyses with very precise studies, so its p-value should never be the sole criterion for a heterogeneity decision.

15.12.5 R Implementation

```
library(metafor)

set.seed(2026)
```

```r
k <- 12
theta <- rnorm(k, 0.4, 0.3)
n_per <- sample(50:400, k, replace = TRUE)
var_i <- 4 / n_per

yi <- theta + rnorm(k, 0, sqrt(var_i))
vi <- var_i

res <- rma(yi, vi, method = "REML")
cat("Q =", round(res$QE, 2),
    "(df=", res$k - 1, ", p =", round(res$QEp, 3), ")\n")
cat("I^2 =", round(res$I2, 1), "%;  tau^2 =", round(res$tau2, 3), "\n")
```

15.12.6 Output & Results

`rma()` returns the Q statistic with its p-value, I^2 as a percentage, and τ^2 on the effect-size scale. Confidence intervals on τ^2 and I^2 are obtained from `confint(res)`, and the prediction interval comes from `predict(res)`. Reporting all four — Q (with its df and p), I^2, τ^2, and the prediction interval — is now standard practice.

15.12.7 Interpretation

A reporting sentence: "Between-study heterogeneity was substantial: $Q = 31.4$ ($\text{df} = 11$, $p = 0.001$), $I^2 = 65\%$, $\tau^2 = 0.07$ ($\tau = 0.26$). The 95 % prediction interval for the true effect in a future study was $[-0.10, 0.85]$, far wider than the confidence interval on the pooled mean ($[0.21, 0.55]$). The random-effects model is therefore appropriate, and a pre-specified meta-regression on study year and population age was conducted to explain the heterogeneity." Always combine the three statistics with the prediction interval.

15.12.8 Practical Tips

- Do not rely on Q's p-value alone; it is underpowered when k is small and over-powered when individual studies are very precise. Always interpret it alongside I^2 and τ^2.
- Report Q, I^2, τ^2, and the 95 % prediction interval together; each tells a different and complementary story about heterogeneity.
- Substantial heterogeneity should prompt follow-up analyses: pre-specified meta-regression for continuous moderators, subgroup analysis for categorical ones, and explicit reporting of prediction intervals to communicate the practical range of true effects.
- I^2 depends on the precision of individual studies as well as on τ^2: large, precise studies inflate I^2 for the same true heterogeneity. The variance component τ^2 is therefore more directly comparable across meta-analyses

than I^2.
- Confidence intervals on I^2 and τ^2 via `confint()` in `metafor` are routinely reported; a wide CI on I^2 should temper interpretation of any single point estimate.
- Use REML for τ^2 as the default; DerSimonian–Laird is convenient and traditional but can be biased downward when heterogeneity is large, and the Paule–Mandel estimator is also widely accepted.

15.12.9 R Packages Used

`metafor::rma()` for the canonical Q, I^2, and τ^2 output, with `confint()` for variance-component intervals and `predict()` for prediction intervals; `meta::metagen()` and related `meta` functions for the alternative interface; `dmetar` for Knapp–Hartung-corrected heterogeneity reporting; `metafor permutest()` for permutation-based heterogeneity inference under small k.

15.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.12.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.13 Individual Participant Data Meta-Analysis

15.13.1 Introduction

Individual participant data (IPD) meta-analysis pools raw participant-level data from multiple studies, enabling analyses that aggregate-data meta-analyses cannot – time-to-event across studies, individual-level effect modification, and harmonised outcome definitions. It is the gold standard when feasible but requires data-sharing agreements.

15.13.2 Prerequisites

Meta-analysis pooling; mixed-effects models.

15.13.3 Theory

Two-stage: analyse each study separately to obtain effect and variance; pool with standard meta-analysis methods.

One-stage: fit a single hierarchical model on the combined data with study as a random effect. Allows individual-participant-level interactions.

Both approaches typically yield similar results for main effects; one-stage is preferred for individual-level moderator analyses.

15.13.4 Assumptions

Data-sharing agreements in place; common outcome definitions across studies; correct handling of study-level heterogeneity.

15.13.5 R Implementation

```r
library(lme4)

set.seed(2026)
n_studies <- 5
participants_per <- 100

# Simulate IPD
df <- do.call(rbind, lapply(1:n_studies, function(s) {
  n <- participants_per
  data.frame(
    study = s,
    arm = factor(sample(c("trt", "ctrl"), n, replace = TRUE)),
    age = rnorm(n, 50, 10)
  )
}))
df$arm_num <- as.integer(df$arm == "trt")
df$y <- 0.4 * df$arm_num + 0.05 * df$age +
        rnorm(nrow(df), 0, 1) +
        rep(rnorm(n_studies, 0, 0.3), each = participants_per)

# One-stage mixed-effects model
one_stage <- lmer(y ~ arm + age + (1 | study), data = df)
summary(one_stage)$coefficients

# Two-stage: per-study effects, then pool
per_study <- split(df, df$study) |> sapply(function(s) {
  m <- lm(y ~ arm + age, data = s)
  c(est = coef(m)["armtrt"], var = vcov(m)["armtrt", "armtrt"])
})
```

```r
library(metafor)
ma <- rma(yi = per_study["est.armtrt", ], vi = per_study["var", ],
          method = "REML")
ma$b
```

15.13.6 Output & Results

One-stage arm coefficient and the equivalent two-stage pooled estimate; typically close agreement for simple models.

15.13.7 Interpretation

"IPD meta-analysis estimated the intervention effect at 0.42 (95 % CI 0.28-0.56, $p < 0.001$); two-stage and one-stage approaches agreed closely, supporting the finding's robustness to analytic framing."

15.13.8 Practical Tips

- Prospective IPD consortium studies avoid selection bias from data-availability issues.
- Harmonise outcomes before pooling; different outcome definitions across studies cause bias.
- One-stage model allows individual-level interactions (effect modifiers); more valuable than ecological moderators.
- For survival endpoints, IPD is required (aggregate hazard ratios are difficult to pool correctly).
- Report PRISMA-IPD checklist for transparency.

15.13.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.13.10 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.14 Leave-One-Out Meta-Analysis

15.14.1 Introduction

Leave-one-out (LOO) meta-analysis systematically removes each study and recomputes the pooled estimate using the remaining $k-1$ studies. The resulting per-removal estimates show how much each study drives the overall conclusion. Studies whose removal substantially changes the pooled effect — magnitude, significance, or both — are influential and warrant careful interpretation. LOO is a routine sensitivity analysis in modern meta-analysis reporting.

15.14.2 Prerequisites

A working understanding of pooled meta-analysis (fixed or random effects) and the concept of influence diagnostics from regression analysis.

15.14.3 Theory

For each study $i = 1, \ldots, k$, compute $\hat{\theta}_{(-i)}$ by pooling the other $k-1$ studies under the same meta-analysis model. Plot $\hat{\theta}_{(-i)}$ with its confidence interval across i; large deviations from the full-model pooled estimate indicate influential studies.

LOO complements other influence measures: Cook's distance and DFBETAS quantify how much each study perturbs the overall fit, and Baujat plots show each study's contribution to heterogeneity (Q) versus its contribution to the pooled estimate. Used together, these diagnostics provide a comprehensive picture of which studies the conclusion depends on.

15.14.4 Assumptions

Same as the underlying random-effects meta-analysis; each leave-one-out fit uses the same model and method as the full pooling.

15.14.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 10
yi <- c(0.35, 0.42, 0.25, 0.55, 0.28, 0.39, 0.95, 0.31, 0.44, 0.36)
vi <- c(0.02, 0.03, 0.015, 0.02, 0.025, 0.018, 0.01, 0.04, 0.02, 0.022)

res <- rma(yi, vi, method = "REML")
loo <- leave1out(res)

print(loo[, c("estimate", "ci.lb", "ci.ub", "I2")])
```

```r
# Influence diagnostics: Cook's distance, DFBETAS
inf <- influence(res)
print(inf$inf)
```

15.14.6 Output & Results

`leaveout()` returns a per-study table of the recomputed pooled estimate, confidence interval, and I^2 when each study is omitted. `influence()` provides Cook's distance, DFBETAS, hat values, and other diagnostics computed jointly.

15.14.7 Interpretation

A reporting sentence: "Leave-one-out analysis revealed study 7 (SMD = 0.95, the largest single estimate) as highly influential; removing it reduced the pooled SMD from 0.41 to 0.35 and shrunk I^2 from 38 % to 18 %. Conclusions are robust to removal of any other study; the sensitivity analysis without study 7 is reported as supplementary." Always describe what an influential study removal does to both effect size and heterogeneity.

15.14.8 Practical Tips

- Pair LOO with Cook's distance and DFBETAS from `metafor::influence()`; the combination flags both effect-size influence and overall fit influence.
- If removing any single study crosses the significance line, the conclusion is fragile and should be reported as such.
- LOO is descriptive — it does not correct any bias. Use it to inform readers about robustness, not to "clean up" the meta-analysis post-hoc.
- Report LOO results only if at least one study is markedly influential; otherwise the table conveys no useful information beyond the headline pooled estimate.
- Combine with Baujat plots (`metafor::baujat()`) for a richer visualisation: studies in the upper-right corner contribute disproportionately to both heterogeneity and effect.
- For meta-analyses with fewer than 5 studies, LOO is unstable and largely uninformative; report it but interpret cautiously.

15.14.9 R Packages Used

`metafor::leaveout()` for the canonical LOO output; `metafor::influence()` for Cook's distance, DFBETAS, and hat values; `metafor::baujat()` for Baujat plots; `meta` for analogous sensitivity tools in an alternative interface; `dmetar::infA()` for combined influence and outlier diagnostics in an integrated workflow.

15.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.14.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.15 Pooling Log-Odds Ratios

15.15.1 Introduction

For binary outcomes, the odds ratio (OR) is the most widely reported effect measure across case-control studies, randomised trials with binary endpoints, and observational comparative studies. Because the OR is bounded below at zero with a long right tail, its sampling distribution is markedly skewed and inverse-variance pooling on the natural scale would be poorly behaved. The standard approach is therefore to pool on the log scale, where the distribution is approximately Normal, and back-transform the pooled estimate and its confidence interval to the OR scale for reporting. Pooling on the log scale stabilises variance, allows symmetric confidence intervals on the analysis scale, and keeps the multiplicative-on-back-transform interpretation that clinicians and epidemiologists expect.

15.15.2 Prerequisites

A working understanding of odds ratios as a measure of association in 2×2 tables, inverse-variance meta-analytic weighting, and the distinction between fixed-effect and random-effects models.

15.15.3 Theory

For a study with a_i, b_i, c_i, d_i in the four cells of the exposure-by-outcome 2×2 table, the per-study log-OR and its variance are

$$\log \mathrm{OR}_i = \log \frac{a_i d_i}{b_i c_i}, \qquad \mathrm{Var}(\log \mathrm{OR}_i) = \frac{1}{a_i} + \frac{1}{b_i} + \frac{1}{c_i} + \frac{1}{d_i}.$$

A fixed-effect pool is $\hat{\theta} = \sum w_i \theta_i / \sum w_i$ with $w_i = 1/\mathrm{Var}(\log \mathrm{OR}_i)$. A random-effects pool adds the between-study variance τ^2 to each study's weight, widening

the confidence interval to reflect between-study uncertainty about the underlying effect. Both pools are fit on the log scale and back-transformed via $\exp(\cdot)$ for reporting.

15.15.4 Assumptions

Studies are independent; the log-OR is approximately Normal (typically holds when each cell has ≥ 5 events); the within-study variance is accurately estimated; the pooled population is reasonably homogeneous (fixed-effect) or, more commonly, is heterogeneous with effects drawn from a Normal distribution around a common mean (random-effects).

15.15.5 R Implementation

```r
library(metafor)

set.seed(2026)
k <- 10
df <- data.frame(
  ai = rpois(k, 30), bi = rpois(k, 170),
  ci = rpois(k, 20), di = rpois(k, 180)
)

es <- escalc(measure = "OR", ai = ai, bi = bi,
             ci = ci, di = di, data = df)

fe <- rma(yi, vi, data = es, method = "FE")
re <- rma(yi, vi, data = es, method = "REML")

cat("FE pooled OR:", round(exp(fe$b), 2),
    " 95% CI:", paste(round(exp(c(fe$ci.lb, fe$ci.ub)), 2), collapse =
    ↪  "-"), "\n")
cat("RE pooled OR:", round(exp(re$b), 2),
    " 95% CI:", paste(round(exp(c(re$ci.lb, re$ci.ub)), 2), collapse =
    ↪  "-"), "\n")
```

15.15.6 Output & Results

`escalc(measure="OR")` computes the per-study log-OR and its variance, applying continuity correction to zero cells when needed. `rma()` fits the meta-analysis on the log scale; back-transformation via `exp()` yields the pooled OR and its confidence interval on the natural scale. The random-effects model additionally returns τ^2 and I^2 describing between-study heterogeneity.

15.15.7 Interpretation

A reporting sentence: "Random-effects meta-analysis of 10 studies (REML, total n = 4,000) gave a pooled odds ratio of 1.56 (95 % CI 1.31 to 1.87) with modest between-study heterogeneity (I^2 = 22%, τ^2 = 0.01); the fixed-effect estimate was almost identical (1.54), confirming that heterogeneity did not materially shift the central effect." Always report both the pooled estimate and the heterogeneity metrics; readers cannot judge the pool without them.

15.15.8 Practical Tips

- Always pool on the log scale; ORs are multiplicative and pooling on the natural scale produces biased and asymmetrically distributed estimates.
- Apply a continuity correction (typically add 0.5 to every cell of a study with any zero cell) for studies with zero events; `metafor::escalc()` handles this automatically by default.
- Choose between fixed-effect and random-effects models based on the substantive research question and expected sources of heterogeneity, not on the Q-test p-value alone — I^2 and τ^2 are more informative about the magnitude of heterogeneity.
- Always report heterogeneity statistics (τ^2, I^2, and the prediction interval) alongside the pooled point estimate.
- Apply the Hartung–Knapp–Sidik–Jonkman adjustment for random-effects pooling with small k (usually fewer than 20 studies); it produces wider, more conservative CIs that better match coverage in simulations.
- Mantel–Haenszel pooling (`metafor::rma.mh()`) is preferable for sparse-data settings with many zero cells, where continuity correction biases the standard inverse-variance pool.

15.15.9 R Packages Used

`metafor::escalc()` and `rma()` for the canonical log-OR pooling workflow; `metafor::rma.mh()` for Mantel–Haenszel pooling and `rma.peto()` for the Peto OR method on rare events; `meta::metabin()` as the alternative interface with cleaner forest-plot defaults; `metafor` Hartung–Knapp via `test="knha"`.

15.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.15.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.16 Meta-Regression

15.16.1 Introduction

Meta-regression extends the random-effects meta-analysis by allowing study-level covariates — moderators such as publication year, mean participant age, average dose, or risk-of-bias score — to enter the model as predictors of the underlying effect. The fitted slope quantifies how much the true study effect changes per unit of moderator, and a residual between-study variance τ^2 describes whatever heterogeneity the moderator did not explain. Meta-regression is the appropriate tool whenever the candidate moderator is continuous, and is generally more powerful than dichotomising into subgroups, which discards information and can produce a categorisation-dependent answer. It is now standard practice for any meta-analysis with $k \geq 10$ studies and a pre-specified hypothesis about effect modification.

15.16.2 Prerequisites

A working understanding of random-effects meta-analysis (notably the role of τ^2), ordinary linear regression, and the difference between fixed-effect, random-effects, and mixed-effects models in the meta-analytic context.

15.16.3 Theory

The mixed-effects meta-regression model is

$$y_i = \beta_0 + \beta_1 x_i + u_i + \varepsilon_i, \qquad u_i \sim N(0, \tau^2), \quad \varepsilon_i \sim N(0, s_i^2),$$

where u_i is between-study residual heterogeneity and ε_i is within-study sampling error with known variance s_i^2. REML estimates τ^2 jointly with the regression coefficients; Wald or likelihood-ratio tests assess moderator effects. The pseudo-$R^2 = 1 - \tau^2_{\text{residual}}/\tau^2_{\text{null}}$ summarises the proportion of between-study variance explained by the moderator and is the analogue of the regression R^2.

15.16.4 Assumptions

The moderator is pre-specified rather than data-mined; the functional form (linear unless explicitly modelled otherwise) is correct; the residual heterogeneity is normal; and there are sufficient studies — a frequently-cited rule of thumb is at least ten studies per moderator coefficient — to support reliable estimation.

15.16.5 R Implementation

```r
library(metafor)

set.seed(2026)
k <- 15
year <- sample(1995:2023, k, replace = TRUE)
mean_age <- runif(k, 40, 70)

yi <- 0.5 - 0.02 * (year - 1995) + rnorm(k, 0, 0.15)
vi <- 1 / sample(50:250, k, replace = TRUE)

df <- data.frame(yi, vi, year, mean_age)

res1 <- rma(yi, vi, mods = ~ year, data = df, method = "REML")
summary(res1)

res2 <- rma(yi, vi, mods = ~ year + mean_age, data = df, method = "REML")
summary(res2)

cat("R^2:", round(res1$R2, 3), "\n")
```

15.16.6 Output & Results

`rma()` with a `mods` formula returns moderator coefficients with confidence intervals and p-values, the residual heterogeneity statistics (τ^2, I^2, residual Q_E), an omnibus moderator test (Q_M), and the pseudo-R^2 summarising explained heterogeneity. Reporting all four elements — slope, residual τ^2, Q_M, and R^2 — is standard practice.

15.16.7 Interpretation

A reporting sentence: "Year of publication significantly moderated the pooled effect (slope -0.017, 95 % CI -0.029 to -0.005, $p = 0.01$): each additional year reduced the standardised mean difference by 0.017 units. Meta-regression explained 52 % of the between-study variance (R^2); residual heterogeneity remained moderate ($I^2_{\text{residual}} = 21\%$). The temporal trend was pre-specified in the PROSPERO protocol." Always report both the slope and the residual heterogeneity to convey what the moderator did and did not explain.

15.16.8 Practical Tips

- Restrict to roughly one moderator per ten studies; meta-regression overfits aggressively because between-study sample sizes are tiny by regression standards.

- Pre-specify all moderators in the PROSPERO protocol; post-hoc moderators are at best exploratory and should be flagged as such in reporting.
- Centre continuous moderators (e.g., year minus mean year) so the intercept has a meaningful interpretation as the effect at the mean moderator value.
- For small k, permutation tests via `permutest()` provide more conservative inference than the Wald-based default and are widely recommended.
- Beware ecological fallacy: a study-level moderator effect does not imply the same effect at the individual-participant level; that requires IPD meta-analysis.
- Plot bubble plots (`metafor::regplot()`) showing study effects, moderator value, and weighted regression line; they make the moderator relationship visible and reveal influential outliers.

15.16.9 R Packages Used

`metafor::rma()` with `mods=~...` for the canonical mixed-effects meta-regression; `metafor::regplot()` for bubble plots; `metafor::permutest()` for permutation-based inference under small k; `meta::metareg()` for the alternative `meta`-package interface; `robumeta` for robust-variance meta-regression with dependent effect sizes.

15.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.16.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.17 Network Meta-Analysis: Introduction

15.17.1 Introduction

Network meta-analysis (NMA), or mixed-treatment comparison (MTC), pools evidence across studies comparing different pairs of treatments to estimate a coherent set of relative effects. It enables comparisons for which no direct head-to-head trials exist, provided indirect paths in the evidence network connect the treatments.

15.17.2 Prerequisites

Pairwise meta-analysis; graph structure; consistency concept.

15.17.3 Theory

A network has nodes (treatments) and edges (pairwise comparisons with at least one study). Direct evidence = studies comparing treatments directly; indirect evidence = inferred via a common comparator. The pooled estimate combines both under an assumption of **consistency** (direct = indirect estimates).

Implementation: frequentist (`netmeta`) or Bayesian (`gemtc`, `BUGSnet`).

15.17.4 Assumptions

Transitivity: studies in the network are sufficiently similar that indirect comparisons are valid. **Consistency**: direct and indirect estimates of the same contrast agree within sampling error.

15.17.5 R Implementation

```r
library(netmeta)

# Example dataset
set.seed(2026)
net_df <- data.frame(
  study = c("S1", "S1", "S2", "S2", "S3", "S3", "S4", "S4", "S5", "S5"),
  treat = c("A", "B", "A", "C", "B", "C", "A", "D", "C", "D"),
  TE    = c(0, -0.3, 0, -0.5, 0.4, 0.2, 0, -0.2, 0, 0.3),
  seTE  = c(0.2, 0.2, 0.25, 0.25, 0.3, 0.3, 0.2, 0.2, 0.22, 0.22)
)
# Rearrange to pairs
p_df <- pairwise(treat = treat, TE = TE, seTE = seTE,
                 studlab = study, data = net_df)

net <- netmeta(p_df, common = FALSE, reference = "A")
summary(net)
netgraph(net)
```

15.17.6 Output & Results

A network graph, pooled treatment effects relative to the reference, and overall I^2.

15.17.7 Interpretation

"Network meta-analysis of 5 studies covering 4 treatments (A reference, B, C, D) yielded pooled SMDs: B vs A -0.22 (-0.41, -0.03); C vs A -0.35 (-0.58, -0.12); D vs A 0.05 (-0.15, 0.25). Transitivity was plausible; consistency was not violated."

15.17.8 Practical Tips

- Always produce a network graph first; disconnected components cannot be compared.
- Check transitivity via study characteristics (populations, doses, outcomes).
- Use SUCRA to rank treatments by their probability of being best.
- Report PRISMA-NMA checklist; network meta-analyses are complex and demand transparency.
- Bayesian NMA via `gemtc` is more flexible for hierarchical models and missing-arm data.

15.17.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.17.10 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.18 NMA Consistency and Inconsistency

15.18.1 Introduction

Network meta-analysis combines direct evidence from head-to-head trials with indirect evidence inferred through common comparators. The validity of this synthesis hinges on the consistency assumption: across closed loops in the network, the direct and indirect estimates of any pairwise contrast must agree up to sampling error. When they disagree systematically — when, for example, the direct A-vs-C estimate from head-to-head trials differs materially from the indirect estimate inferred via B — the network is inconsistent, and the pooled NMA estimates cannot be interpreted at face value. Detecting and characterising inconsistency is therefore a mandatory step in any rigorous NMA workflow,

and modern guidance (NICE DSU, ISPOR, PRISMA-NMA) requires that consistency be reported alongside the league table and SUCRA results.

15.18.2 Prerequisites

A working understanding of network meta-analysis structure (treatments, designs, contrasts), the difference between direct and indirect evidence, and the transitivity assumption that underpins indirect comparisons.

15.18.3 Theory

Three complementary tests address different scales of inconsistency. The loop-specific Bucher test, for a triangle A-B-C, compares the direct A-vs-C estimate against the indirect estimate $\hat{\theta}_{AB} + \hat{\theta}_{BC}$ via a Wald-type difference. Node-splitting (Dias et al., 2010) generalises this for any contrast: it splits the evidence for one contrast into direct-only and indirect-only components and tests whether they agree. The design-by-treatment interaction model (Higgins et al., 2012) provides a global test via the generalised-inconsistency framework, comparing the consistent NMA model against an unconstrained model that allows different contrasts to take different values across study designs.

15.18.4 Assumptions

The network contains at least one closed loop (otherwise consistency cannot be tested), transitivity holds in principle, and the underlying meta-analytic model is correctly specified. Inconsistency tests have low power when loops are short or sparsely populated; absence of detected inconsistency does not prove consistency.

15.18.5 R Implementation

```
library(netmeta)

set.seed(2026)
net_df <- data.frame(
  study = c("S1", "S1", "S2", "S2", "S3", "S3", "S4", "S4",
            "S5", "S5", "S6", "S6"),
  treat = c("A", "B", "A", "C", "B", "C", "A", "D",
            "B", "D", "C", "D"),
  TE    = c(0, -0.3, 0, -0.5, 0.4, 0.2, 0, -0.2,
            0.1, 0.3, 0, 0.4),
  seTE  = rep(0.25, 12)
)

p_df <- pairwise(treat = treat, TE = TE, seTE = seTE,
                 studlab = study, data = net_df)
```

```r
net  <- netmeta(p_df, common = FALSE, reference = "A")

netsplit(net)

decomp.design(net)
```

15.18.6 Output & Results

`netsplit()` returns per-comparison direct vs indirect estimates, their difference, and a Wald test on the difference. `decomp.design()` decomposes the heterogeneity statistic Q into within-design and between-design components and reports the global design-by-treatment interaction test. Non-significant results in both diagnostics support the consistency assumption.

15.18.7 Interpretation

A reporting sentence: "Node-splitting (`netsplit`) showed agreement between direct and indirect estimates for all six pairwise comparisons (smallest two-sided p-value = 0.15). The global design-by-treatment interaction test (`decomp.design`) was non-significant ($p = 0.36$), and the between-design heterogeneity component was negligible ($Q_{\text{inc}} = 1.2$, df = 3). The consistency assumption is therefore supported, and the random-effects league table can be interpreted as reflecting the joint direct and indirect evidence." Always report both tests.

15.18.8 Practical Tips

- Always test consistency; reporting an NMA without any consistency check is no longer acceptable in major journals or HTA submissions.
- If inconsistency is detected, investigate study-level effect modifiers (population age, baseline severity, dose, year, risk of bias) that differ across designs; meta-regression within the NMA framework can then partially resolve the inconsistency.
- `netsplit` provides per-comparison tests (high resolution, low power per test); `decomp.design` provides the global test (low resolution, higher power overall) — they are complementary, not redundant.
- For simple triangles with sufficient direct evidence, the Bucher test is a quick first-pass diagnostic; it is what node-splitting reduces to in the three-treatment case.
- Inconsistency can be local (a single loop) or global (the whole network); the diagnostic patterns differ, and the discussion section should reflect which kind was found.
- Bayesian implementations in `gemtc` and `multinma` provide UME (unrelated mean effects) consistency models that compare against the consistent NMA via DIC; this is the Bayesian analogue of the design-by-treatment test.

15.18.9 R Packages Used

`netmeta::netsplit()` and `netmeta::decomp.design()` for the canonical frequentist consistency diagnostics, alongside `netmeta()` itself; `gemtc` and `multinma` for Bayesian NMA with built-in UME and node-splitting; `pcnetmeta` for component-NMA inconsistency analysis; `BUGSnet` for combined Bayesian NMA workflow including consistency reporting.

15.18.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.18.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.19 NMA League Tables

15.19.1 Introduction

The league table is the standard summary display for a network meta-analysis (NMA). It packs every pairwise treatment contrast — direct, indirect, and mixed — into a single triangular or square grid, making the relative ranking of all interventions visible at a glance. Each diagonal cell labels a treatment; each off-diagonal cell reports the pooled contrast between row and column with its 95 % confidence (or credible) interval. League tables are now a standard reporting requirement in the PRISMA-NMA extension because they communicate the entire network's information in a way that no single forest plot can.

15.19.2 Prerequisites

A working understanding of network meta-analysis (transitivity, consistency), how mixed treatment-comparison estimates combine direct and indirect evidence, and the conventions of pairwise effect sizes (OR, RR, SMD).

15.19.3 Theory

For T treatments the league table is a $T \times T$ matrix. The convention used by `netmeta` and most software places the random- or common-effect mixed estimates in the upper triangle and direct-only pooled estimates in the lower triangle, with

the same treatment ordering on both axes. Each off-diagonal cell reports the pooled contrast (row-vs-column or column-vs-row, with the convention noted in the caption) and its uncertainty.

Ordering treatments by their rank (e.g., SUCRA, P-score, or median ranking from a Bayesian NMA) groups the best-performing treatments in one corner, making the table substantially easier to interpret than alphabetical ordering.

15.19.4 Assumptions

The underlying NMA is valid: transitivity holds across studies that compare different treatment subsets, consistency holds between direct and indirect evidence, and all pairwise contrasts are estimable from the connected network. Reporting league tables for a disconnected network is meaningless.

15.19.5 R Implementation

```
library(netmeta)

set.seed(2026)
net_df <- data.frame(
  study = c("S1", "S1", "S2", "S2", "S3", "S3", "S4", "S4",
            "S5", "S5", "S6", "S6"),
  treat = c("A", "B", "A", "C", "B", "C", "A", "D",
            "B", "D", "C", "D"),
  TE    = c(0, -0.3, 0, -0.5, 0.4, 0.2, 0, -0.2,
            0.1, 0.3, 0, 0.4),
  seTE  = rep(0.25, 12)
)

p_df <- pairwise(treat = treat, TE = TE, seTE = seTE,
                 studlab = study, data = net_df)
net  <- netmeta(p_df, common = FALSE, reference = "A")

league <- netleague(net, bracket = "(",
                    separator = " to ", digits = 2)
league$random
```

15.19.6 Output & Results

`netleague()` returns a structured matrix of pairwise contrast estimates with their confidence intervals, separately for the common-effect and random-effects models. The diagonal carries treatment labels; off-diagonal cells contain the pooled contrast and its CI in a formatted string. Exporting the table to LaTeX or HTML for publication uses standard formatting tools.

15.19.7 Interpretation

A reporting sentence: "The league table summarises all six pairwise contrasts in the network of four treatments. Treatment C was consistently the most effective: C vs A had a pooled effect of -0.55 (95 % CI -0.88 to -0.22) and C vs B was -0.34 (95 % CI -0.67 to -0.01); both contrasts excluded the null. The lower-triangle direct-only estimates were qualitatively concordant, suggesting consistency of direct and indirect evidence." Always pair league-table reading with a node-splitting consistency check.

15.19.8 Practical Tips

- Order treatments by their NMA-derived ranking (e.g., SUCRA from highest to lowest); the best-vs-worst contrast then sits in one corner and visual scanning becomes intuitive.
- Report both the point estimate and the CI; a significant contrast is one whose CI excludes the null on the analysis scale, but readers benefit from seeing the magnitude as well.
- For binary outcomes, exponentiate to OR or RR for clinical interpretation; on the multiplicative scale, a CI excluding 1 is the significance criterion.
- Complement the league table with a forest plot of contrasts versus a clinically meaningful reference treatment; the two views together communicate more than either alone.
- For networks with more than ten treatments, a full league table becomes unwieldy; consider a heatmap visualisation or restrict the table to clinically relevant pairs.
- Always cite the consistency assumption in the caption and report a node-splitting or design-by-treatment interaction check elsewhere in the manuscript.

15.19.9 R Packages Used

`netmeta::netleague()` for the canonical league-table output and `netmeta()` for the underlying frequentist NMA; `gemtc` and `multinma` for Bayesian NMA league-table equivalents using rank tables; `pcnetmeta` for component network meta-analysis with extended league-table summaries; `dmetar` for combined NMA workflow and reporting helpers.

15.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.19.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.20 SUCRA Rankings

15.20.1 Introduction

SUCRA — the Surface Under the Cumulative Ranking curve, introduced by Salanti and colleagues in 2011 — summarises a network meta-analysis's treatment ranking into a single interpretable number between 0 and 1. A SUCRA of 1 means the treatment is always ranked best across the posterior or simulation distribution; a SUCRA of 0 means it is always ranked worst. Because clinicians, guideline panels, and patients want to know "which treatment should I use?", and a triangular league table cannot answer that directly, SUCRA quickly became the most-cited summary in NMA papers. Its widespread use also brings risks: SUCRA is sensitive to the number of treatments included, can exaggerate small effect-size differences, and must always be interpreted alongside the absolute effect estimates and their uncertainty.

15.20.2 Prerequisites

A working understanding of network meta-analysis, posterior or simulated rank distributions, and the conceptual difference between a treatment ranking and a treatment effect.

15.20.3 Theory

For each treatment j in a network of T treatments, the rank distribution gives $P(\text{rank}_j = k)$ for $k = 1, \dots, T$ — typically derived from a Bayesian posterior with MCMC or from frequentist parametric simulation. SUCRA is

$$\text{SUCRA}_j = \frac{1}{T-1} \sum_{k=1}^{T-1} P(\text{rank}_j \le k),$$

the area under the cumulative rank curve scaled to $[0, 1]$. Rücker and Schwarzer's P-score (2015) is the frequentist analogue computed directly from NMA point estimates and standard errors; it does not require MCMC and reproduces SUCRA values closely in most networks.

15.20.4 Assumptions

The underlying NMA is valid (transitivity across studies, consistency between direct and indirect evidence), and the network is connected so all rankings are

estimable. The clinical interpretability of "best treatment" requires that the treatments are exchangeable in indication and that no contraindications make a high-ranked option unusable for some patient groups.

15.20.5 R Implementation

```r
library(netmeta)

set.seed(2026)
net_df <- data.frame(
  study = c("S1", "S1", "S2", "S2", "S3", "S3", "S4", "S4",
            "S5", "S5", "S6", "S6"),
  treat = c("A", "B", "A", "C", "B", "C", "A", "D",
            "B", "D", "C", "D"),
  TE    = c(0, -0.3, 0, -0.5, 0.4, 0.2, 0, -0.2,
            0.1, 0.3, 0, 0.4),
  seTE  = rep(0.25, 12)
)

p_df <- pairwise(treat = treat, TE = TE, seTE = seTE,
                 studlab = study, data = net_df)
net  <- netmeta(p_df, common = FALSE, reference = "A",
                small.values = "good")

rankings <- netrank(net, method = "P-score", small.values = "good")
rankings$ranking.random
```

15.20.6 Output & Results

`netrank()` returns a P-score per treatment (between 0 and 1) under both common-effect and random-effects models. Higher scores indicate better ranking on the chosen analysis scale. For Bayesian NMA, packages such as `gemtc` and `multinma` produce SUCRA values directly, along with rank probabilities $P(\text{rank} = k)$ for each k.

15.20.7 Interpretation

A reporting sentence: "Treatment C had the highest P-score (0.89) under the random-effects NMA, identifying it as most likely the best of the four candidates. However, the pairwise contrast C vs B (P-score 0.61) had a 95 % CI that crossed the null, so the ranking is uncertain and absolute differences between top-ranked options are small. Rankograms reporting the full distribution over ranks accompany the SUCRA summary in the supplement." Always report rankings together with absolute effect estimates and the surrounding uncertainty.

15.20.8 Practical Tips

- Always report SUCRA or P-score alongside the pairwise effect estimates and their CIs; rankings alone can exaggerate small differences and obscure clinically irrelevant gaps.
- Treatments with similar point estimates can show very different SUCRA values when uncertainty differs; the more precise treatment will rank higher even if its mean effect is no better. This is a feature, not a bug — but it must be interpreted carefully.
- Use rankograms (bar plots of $P(\text{rank} = k)$ for each k) to visualise the entire rank distribution; they convey the uncertainty that the single SUCRA number compresses.
- Frequentist P-scores and Bayesian SUCRA values typically agree to within rounding error on connected networks; the choice between them is mainly methodological preference.
- For regulatory and reimbursement decisions, the absolute pairwise effect and its CI matter more than the ranking; SUCRA is a clinical-communication tool, not a decision rule.
- Check `small.values` carefully: misspecifying whether small effect sizes are "good" (e.g., for a harm outcome) or "bad" inverts the entire ranking and is one of the most common analytic errors in published NMAs.

15.20.9 R Packages Used

`netmeta::netrank()` for P-scores in frequentist NMA; `gemtc::rank.probability()` and `multinma::posterior_rank_probs()` for Bayesian SUCRA and rank tables; `BUGSnet` for rankograms with built-in convergence diagnostics; `pcnetmeta` for component-NMA ranking when treatments are decomposable into pieces.

15.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.20.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.21 Prediction Intervals in Meta-Analysis

15.21.1 Introduction

A 95 % confidence interval on a pooled meta-analytic estimate describes uncertainty about the **mean** true effect across the included studies. The prediction interval (PI), in contrast, describes the plausible range of the **true effect in a future study** drawn from the same population, propagating both the uncertainty in the mean and the between-study heterogeneity τ^2. Under non-trivial heterogeneity the PI is substantially wider than the CI, and it is far more clinically meaningful: a clinician treating the next patient cares less about uncertainty in the average across past studies than about the range of effects they might plausibly see in their own clinical context. Modern meta-analysis reporting standards (PRISMA, GRADE) increasingly require the PI alongside the CI for any random-effects synthesis.

15.21.2 Prerequisites

A working understanding of random-effects meta-analysis, the role of between-study variance τ^2, and the conceptual difference between uncertainty about a mean and uncertainty about a future realisation.

15.21.3 Theory

The 95 % prediction interval under a random-effects meta-analysis with k studies is

$$\mathrm{PI}_{95\%} = \hat{\theta} \pm t_{0.975,k-2}\sqrt{\hat{\tau}^2 + \mathrm{SE}(\hat{\theta})^2}.$$

Two features distinguish it from the CI: the use of Student's t with $k-2$ degrees of freedom rather than the normal quantile, and the addition of $\hat{\tau}^2$ inside the square root. As heterogeneity grows the PI widens considerably while the CI is comparatively unchanged, so the gap between them is itself a useful diagnostic for how much pooling has — or has not — clarified the underlying effect.

15.21.4 Assumptions

The random-effects model is valid; τ^2 is reasonably estimated (typically by REML, which is the recommended default); future studies are exchangeable with the included ones, drawing their true effects from the same Normal distribution centred on the pooled mean.

15.21.5 R Implementation

```
library(metafor)
```

```r
set.seed(2026)
k <- 10
yi <- rnorm(k, 0.4, 0.25)
vi <- 1 / sample(50:200, k, replace = TRUE)

res <- rma(yi, vi, method = "REML")
summary(res)

pi <- predict(res, addx = NULL)
pi[, c("pred", "pi.lb", "pi.ub")]
```

15.21.6 Output & Results

`predict()` on an `rma` object returns the pooled estimate, its confidence interval (for the mean), and the 95 % prediction interval (for a future study). Whenever $\tau^2 > 0$ the PI is strictly wider than the CI, and the gap is informative: a small gap means heterogeneity is negligible and the pool is informative about future studies, while a large gap means the pool tells you the average but not what to expect next.

15.21.7 Interpretation

A reporting sentence: "The pooled standardised mean difference was 0.38 (95 % CI 0.20 to 0.56) under random-effects REML pooling. The 95 % prediction interval was wider, $[-0.10, 0.86]$, indicating that in a new study the true effect could plausibly range from a small negative value to a large positive value, consistent with the moderate heterogeneity observed ($I^2 = 41\%$, $\tau^2 = 0.04$). Clinical interpretation should account for this dispersion rather than rely solely on the pooled mean." Always report the CI and PI together for random-effects pools.

15.21.8 Practical Tips

- Always report both the CI (uncertainty about the mean) and the PI (uncertainty about a future study) in a random-effects meta-analysis; they answer different questions and the difference is itself informative.
- A wide PI alongside a narrow CI is a distinctive pattern indicating heterogeneity that does not diminish with pooling; it should be acknowledged explicitly in the discussion as a limitation of the synthesis.
- For meta-analyses with very few studies (typically $k < 5$), the PI is unstable and very wide because τ^2 is poorly estimated; report it but interpret with caution and consider Hartung–Knapp adjustment.
- A PI that crosses the null while the CI does not is a common and important pattern: it means the pooled mean is "significant" but a meaningful pro-

portion of future studies might find no effect at all. This finding typically reshapes interpretation and should be highlighted, not buried.
- The Hartung–Knapp–Sidik–Jonkman adjustment widens the CI in small samples and modestly affects the PI; for $k \leq 20$, it is the recommended default.
- Display the PI graphically on the forest plot as a dashed bar around the pooled diamond (`addpred=TRUE` in `metafor::forest()`); seeing it visually conveys more than reporting it numerically.

15.21.9 R Packages Used

`metafor::predict.rma()` for the canonical PI computation alongside the CI; `metafor::forest()` with `addpred=TRUE` for graphical display; `meta::print.meta()` reports PI in summary output for `meta` objects; `dmetar::pi()` for an explicit prediction-interval helper used in teaching contexts.

15.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.21.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.22 Pooling Risk Ratios

15.22.1 Introduction

Risk ratios (RR) — also called relative risks — are more interpretable than odds ratios for common outcomes; they directly express the ratio of event probabilities between two arms. RR approximates OR only when events are rare (probability below ~10 %); for common outcomes, the two diverge substantially. Meta-analytic pooling of binary outcomes from cohort studies and randomised trials therefore typically proceeds on the log-RR scale, analogously to log-OR pooling.

15.22.2 Prerequisites

A working understanding of risk ratios, log transformations for ratio measures, and inverse-variance meta-analysis under random effects.

15.22.3 Theory

For a study with a_i events in $a_i + b_i = n_{1i}$ exposed and c_i events in $c_i + d_i = n_{2i}$ unexposed:

$$\log \mathrm{RR}_i = \log \frac{a_i/n_{1i}}{c_i/n_{2i}}, \qquad \mathrm{Var}(\log \mathrm{RR}_i) = \frac{1}{a_i} - \frac{1}{n_{1i}} + \frac{1}{c_i} - \frac{1}{n_{2i}}.$$

Inverse-variance pooling on the log-RR scale follows the same fixed-effect or random-effects distinction as any other meta-analysis. Back-transformation to the natural RR scale via exp produces interpretable summaries with confidence intervals that respect the multiplicative scale.

15.22.4 Assumptions

Studies are independent; log-RR is approximately Normal (typically a good approximation when each study has at least 5 events per arm); the population RR is the same across studies (fixed-effect) or distributed around a mean with between-study variance τ^2 (random-effects).

15.22.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 10
df <- data.frame(
  ai = rpois(k, 30), n1i = rep(100, k),
  ci = rpois(k, 20), n2i = rep(100, k)
)

es <- escalc(measure = "RR",
             ai = ai, n1i = n1i, ci = ci, n2i = n2i,
             data = df)

re <- rma(yi, vi, data = es, method = "REML")
cat("Pooled RR:", round(exp(re$b), 2),
    " 95% CI:", paste(round(exp(c(re$ci.lb, re$ci.ub)), 2), collapse =
    ↪  "-"), "\n")
cat("tau^2:", round(re$tau2, 3), "  I^2:", round(re$I2, 1), "%\n")
```

15.22.6 Output & Results

`escalc(measure="RR")` computes per-study log-RR and its variance. `rma()` fits the random-effects model on the log scale; back-transformation gives the pooled

RR with confidence intervals on the natural scale, plus τ^2 and I^2 for heterogeneity.

15.22.7 Interpretation

A reporting sentence: "Random-effects meta-analysis of 10 studies (REML, n total = 2{,}000) gave a pooled risk ratio of 1.48 (95 % CI 1.24 to 1.77) with moderate heterogeneity (I^2 = 35%, τ^2 = 0.02); the intervention is associated with a 48 % higher event rate, with the 95 % CI excluding the null." Always report the heterogeneity statistics alongside the pooled estimate.

15.22.8 Practical Tips

- Use RR for cohort studies and randomised controlled trials, where baseline risk is meaningful; use OR for case-control studies, where RR is not estimable from the design.
- For rare events (event rate below 10 %), RR ≈ OR; for common events, the two diverge and the choice matters substantively.
- Zero-event studies require a continuity correction (`escalc(...,add=0.5)`) or a rare-events method (Peto OR, Mantel-Haenszel without correction); standard log-RR is undefined when any cell is zero.
- Present forest plots and reports on the natural scale (RR), not the log scale; readers reason multiplicatively about ratios.
- Mantel-Haenszel pooling (`metafor::rma.mh()`) is a common alternative for sparse-data settings; it does not require log-transformation and handles zero events more gracefully.
- For individual-participant-data meta-analysis, fit a logistic regression with study as a random effect; this is more powerful than aggregate-data pooling when the IPD are available.

15.22.9 R Packages Used

`metafor::escalc()` with `measure="RR"` and `rma()` for the canonical workflow; `meta::metabin()` as an alternative interface with cleaner forest-plot output; `metafor::rma.mh()` for Mantel-Haenszel pooling; `metafor::rma.peto()` for the Peto method on rare events.

15.22.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.22.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.23 Selection Models for Publication Bias

15.23.1 Introduction

Selection models address publication bias more rigorously than trim-and-fill or simple regression-based asymmetry tests. They posit an explicit weight function describing how the probability of publication depends on a study's p-value or effect magnitude, then re-estimate the pooled effect under that assumed selection mechanism to recover a bias-adjusted estimate. Where trim-and-fill is descriptive and triangular in its assumption, selection models are formally model-based and yield likelihood-ratio tests of the selection hypothesis itself. The cost is stronger parametric assumptions about the form of the selection process, which means selection models are most useful when the meta-analysis is large enough that the selection function can be reasonably identified from the data.

15.23.2 Prerequisites

A working understanding of publication bias as a selection process, maximum-likelihood estimation, and the logic of inverse-probability weighting on a sampled population.

15.23.3 Theory

Given observed effects y_i with standard errors SE_i from published studies, the observed density is

$$p_{\mathrm{obs}}(y) = \frac{w(y, \mathrm{SE})\, p(y)}{\int w(y, \mathrm{SE})\, p(y)\, dy},$$

where $p(y)$ is the population density of true effects and $w(\cdot)$ is a selection weight function — typically a step function in the one-sided p-value, with selection probability dropping at conventional thresholds (0.025, 0.05, 0.10). Maximum-likelihood estimation of the joint (μ, τ^2, w) parameters then inverts the selection mechanism to recover the bias-adjusted pooled effect, and a likelihood-ratio test against the null of $w(\cdot) \equiv 1$ formally tests for selection.

15.23.4 Assumptions

The functional form of the selection weight is correctly specified (or at least a reasonable approximation), the meta-analysis contains enough studies to identify the selection parameters (typically 15–20+), and the studies are exchangeable conditional on their effect size and standard error.

15.23.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 20
yi <- rnorm(k, 0.3, 0.15)
vi <- 1 / sample(40:250, k, replace = TRUE)

res <- rma(yi, vi, method = "REML")

sel <- selmodel(res, type = "stepfun",
                steps = c(0.025, 0.10, 0.50, 1.0),
                alternative = "greater")
sel
```

15.23.6 Output & Results

`selmodel()` returns the bias-adjusted pooled estimate, its standard error and confidence interval, the estimated selection weights at each step, and a likelihood-ratio test against the null of no selection. Comparing the adjusted and unadjusted estimates side by side quantifies how much the conclusion shifts under the assumed selection mechanism.

15.23.7 Interpretation

A reporting sentence: "A step-function selection model with thresholds at $p = 0.025, 0.10, 0.50$ provided strong evidence for selective publication (likelihood-ratio test vs no-selection, $\chi^2 = 13.1$, df $= 3$, $p = 0.004$). The bias-adjusted standardised mean difference was 0.19 (95 % CI 0.05 to 0.33), substantially attenuated from the unadjusted estimate of 0.32 (0.21 to 0.43). The result suggests publication bias inflates the unadjusted pooled effect by roughly 40 %." Always report both estimates and the LR test.

15.23.8 Practical Tips

- Selection models require at least 15–20 studies for stable estimates; below this threshold the selection function is poorly identified and confidence intervals become uninformatively wide.

- Try several weight-function structures (different step thresholds, smooth-monotone forms) as sensitivity analyses; the bias-adjusted estimate can be quite sensitive to the assumed selection mechanism, and convergent results across forms are reassuring.
- Always report both the unadjusted and the selection-adjusted estimates, plus the LR test for selection; the unadjusted estimate remains the primary, and the selection-adjusted estimate is sensitivity.
- Modern alternatives — PET-PEESE regression-based correction, p-uniform and p-curve, and Vevea–Hedges weight-function models — are useful when standard step-function selection models fit poorly or fail to converge.
- Bayesian selection models with informative priors on the weight parameters (e.g., in `RoBMA` and related packages) are a stable alternative when frequentist MLE is unstable.
- Selection models adjust only for the selection mechanism they are told to assume; other forms of bias (selective outcome reporting, dependent effect-size selection) require separate handling.

15.23.9 R Packages Used

`metafor::selmodel()` for the canonical step-function and smooth selection-model implementations; `weightr` for Vevea–Hedges weight-function models; `puniform` and `puniforms` for p-uniform corrections; `RoBMA` for Bayesian model-averaged publication-bias adjustment; `metasens` for limit-meta-analysis and Copas-model alternatives.

15.23.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.23.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.24 Standardised Mean Difference: Hedges' g

15.24.1 Introduction

Hedges' g is the small-sample-corrected standardised mean difference (SMD), the preferred effect-size metric for meta-analysis of continuous outcomes. Cohen's d — the unstandardised SMD — over-estimates the population effect in small samples; Hedges' correction factor J de-biases it. The two are nearly identical for studies with combined sample size above 20 or so, but g should be used routinely whenever any included study is small.

15.24.2 Prerequisites

A working understanding of standardised effect sizes (Cohen's d), pooled standard deviation, and the bias of d in small samples.

15.24.3 Theory

Cohen's d is

$$d = \frac{\bar{X}_1 - \bar{X}_2}{s_{\text{pool}}}, \qquad s_{\text{pool}} = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}.$$

Hedges' correction is $J = 1 - 3/(4\,\text{df} - 1)$ (with $\text{df} = n_1 + n_2 - 2$), and $g = Jd$. The correction is close to 1 for $\text{df} > 20$ but can be substantially below 1 for very small studies. The variance of g is

$$\text{Var}(g) = J^2 \left[\frac{n_1 + n_2}{n_1 n_2} + \frac{d^2}{2(n_1 + n_2)} \right].$$

15.24.4 Assumptions

Outcomes are approximately Normal within groups; group variances are approximately equal (use `SMDH` for unequal-variance variants); independent observations.

15.24.5 R Implementation

```
library(metafor)

set.seed(2026)
n <- 10                  # small study
x1 <- rnorm(n, mean = 1, sd = 1)
x2 <- rnorm(n, mean = 0, sd = 1)

# Compute g and its variance via metafor
```

```r
es <- escalc(measure = "SMD",
             m1i = mean(x1), sd1i = sd(x1), n1i = n,
             m2i = mean(x2), sd2i = sd(x2), n2i = n)
es

# Manual computation
s_pool <- sqrt(((n - 1) * var(x1) + (n - 1) * var(x2)) / (2 * n - 2))
d <- (mean(x1) - mean(x2)) / s_pool
J <- 1 - 3 / (4 * (2 * n - 2) - 1)
c(d = d, g = d * J)
```

15.24.6 Output & Results

`escalc(measure="SMD")` returns Hedges' g (in the `yi` column) and its variance (`vi`) per study, ready for inverse-variance meta-analysis. The manual calculation confirms the J-correction; for $n_1 = n_2 = 10$, $J \approx 0.92$, reducing $d \approx 1.04$ to $g \approx 0.96$.

15.24.7 Interpretation

A reporting sentence: "Hedges' g for the simulated small study was 0.92 (95 % CI 0.01 to 1.83), corresponding to a large effect by Cohen's benchmarks ($d = 0.5$ medium, $d = 0.8$ large); the small-sample correction reduced the raw SMD from 1.04 to 0.92, reflecting the upward bias inherent in d for small studies." Always state the use of g vs d, especially when sample sizes are small.

15.24.8 Practical Tips

- Always use Hedges' g rather than Cohen's d for meta-analysis when any included study has $n_1 + n_2 < 20$; the bias correction is meaningful in that range.
- `metafor::escalc(measure="SMD")` computes g and its variance in a single call; this is the standard workflow.
- For unequal variances (Levene's test rejects), use `measure="SMDH"` which computes g using a pooled SD based on the separate group SDs.
- Report 95 % CIs alongside g; the effect size alone does not convey precision, and meta-analysis weighting depends on the precision.
- For pre-post or within-subject designs, use SMCR or SMCC variants in `metafor::escalc()`; they account for the within-subject correlation that ordinary SMD ignores.
- For binary outcomes, switch to log-odds ratio or risk ratio; SMD is for continuous outcomes only.

15.24.9 R Packages Used

`metafor::escalc()` and `rma()` for the canonical effect-size and meta-analysis workflow; `meta::metacont()` for an alternative interface with continuous-outcome conventions; `metan` for a tidymodels-flavoured wrapper; `dmetar` for additional meta-analysis tools and visualisations.

15.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.24.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.25 Subgroup Meta-Analysis

15.25.1 Introduction

Subgroup meta-analysis partitions the included studies according to a pre-specified categorical moderator — intervention intensity (low vs high dose), population (paediatric vs adult), risk of bias (low vs unclear vs high), or any clinically meaningful split — and pools within each level. The within-subgroup pooled effects describe how the intervention behaves in each stratum, and a formal between-subgroup test assesses whether the moderator explains a measurable share of between-study heterogeneity. Subgroup analysis is the standard tool for categorical effect modification in evidence synthesis and is required by the Cochrane Handbook whenever the protocol pre-specifies a moderator hypothesis.

15.25.2 Prerequisites

A working understanding of pooled meta-analysis (preferably random-effects), the heterogeneity statistics τ^2, I^2, and Q, and the difference between within-group and between-group sources of variation in stratified analyses.

15.25.3 Theory

Within each subgroup $g = 1, \ldots, G$ the studies are pooled to obtain $\hat{\theta}_g$ and its standard error. The between-subgroup heterogeneity statistic is

$$Q_B = \sum_g w_g(\hat{\theta}_g - \hat{\theta})^2,$$

with $G-1$ degrees of freedom; a significant Q_B indicates the moderator accounts for some of the between-study variance. Equivalently, fitting a mixed-effects meta-regression with subgroup as a categorical moderator produces a Q_M test that recovers exactly this comparison and is the modern preferred implementation, because it can also share a common τ^2 across subgroups for more stable inference.

15.25.4 Assumptions

The moderator is pre-specified rather than chosen post hoc, within-subgroup studies are reasonably homogeneous (small within-group τ^2), and each subgroup contains enough studies — typically at least four to five — to support a stable within-group pool. Subgroup analyses with only one or two studies per stratum are largely uninformative and should be reported only descriptively.

15.25.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 16
subgroup <- factor(rep(c("low_dose", "high_dose"), each = k/2))
yi <- c(rnorm(k/2, 0.2, 0.15),
        rnorm(k/2, 0.5, 0.15))
vi <- 1 / sample(50:200, k, replace = TRUE)

df <- data.frame(yi, vi, subgroup)

lapply(split(df, df$subgroup), function(d) {
  rma(yi, vi, data = d, method = "REML")$b
})

res_mod <- rma(yi, vi, mods = ~ subgroup, data = df, method = "REML")
summary(res_mod)
```

15.25.6 Output & Results

The script returns per-subgroup pooled estimates from independent random-effects pools and the joint mixed-effects meta-regression with subgroup as moderator. The latter reports the omnibus Q_M test for between-subgroup heterogeneity, the per-level coefficients, and the residual heterogeneity remaining after the moderator is included — the canonical reporting set for subgroup analyses.

15.25.7 Interpretation

A reporting sentence: "Pre-specified subgroup analysis showed significant between-subgroup heterogeneity ($Q_M = 18.4$, df $= 1$, $p < 0.001$): high-dose studies had a pooled standardised mean difference of 0.52 (95 % CI 0.39 to 0.65) compared with 0.18 (0.08 to 0.28) in low-dose studies, supporting the hypothesis that dose moderates the treatment effect. The moderator explained 47 % of the between-study heterogeneity (R^2); residual I^2 was 18 %." Always report both the within-subgroup pools and the between-subgroup test.

15.25.8 Practical Tips

- Pre-specify all subgroup analyses in the PROSPERO protocol; post-hoc subgrouping is exploratory and should be flagged as such — never as primary inference.
- Small per-subgroup study counts produce unstable pooled estimates; the Cochrane Handbook recommends at least 10 studies per subgroup before drawing conclusions, and warns explicitly against single-stratum subgroup pools.
- Prefer meta-regression over subgroup analysis whenever the moderator is continuous (age, dose, year); dichotomising loses information and depends on an arbitrary cutpoint.
- Report the between-subgroup Q_M or Q_B p-value as the primary inference; comparing within-subgroup p-values across strata ("significant in one, not the other") is a well-known statistical fallacy because tests in small subgroups are underpowered.
- Use a shared τ^2 across subgroups (the default in mixed-effects meta-regression) rather than separate estimates per subgroup unless heterogeneity is clearly different in each; the shared specification is more efficient and more conservative.
- When subgroup analysis is exploratory, apply a multiplicity adjustment or interpret with explicit caution; the false-positive rate inflates rapidly with the number of subgroup tests.

15.25.9 R Packages Used

`metafor::rma()` with `mods=~factor(subgroup)` for the canonical mixed-effects implementation; `meta::metabin()` and `meta::metacont()` with `subgroup=` for built-in subgroup pooling and forest-plot grouping; `dmetar` for Knapp–Hartung-adjusted subgroup analyses; `robumeta` for robust-variance subgroup pooling under correlated effects.

15.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.25.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.26 Tau-Squared Estimation

15.26.1 Introduction

The random-effects meta-analysis requires an estimate of τ^2, the between-study variance. Several estimators exist with different bias-variance trade-offs; the choice affects pooled estimates, CIs, and downstream inference. REML is the modern default; DerSimonian-Laird (DL) the historical default.

15.26.2 Prerequisites

Random-effects meta-analysis; method-of-moments vs maximum likelihood.

15.26.3 Theory

Common estimators: - **DerSimonian-Laird (DL)**: method-of-moments; fastest, historically default. - **REML (Restricted Maximum Likelihood)**: likelihood-based, generally less biased than DL; modern default. - **Paule-Mandel (PM)**: iterative method-of-moments; often similar to REML. - **Sidik-Jonkman (SJ)**: analytic, robust to model misspecification. - **Empirical Bayes (EB)**: iterative Bayes-style estimator.

DL systematically underestimates τ^2 under moderate heterogeneity; REML is close to unbiased.

15.26.4 Assumptions

True effects are drawn from a distribution (typically normal) with common variance τ^2; within-study variances are known.

15.26.5 R Implementation

```
library(metafor)

set.seed(2026)
k <- 10
```

```r
n_per <- sample(30:200, k, replace = TRUE)
true_theta <- rnorm(k, 0.3, 0.25)
yi <- true_theta + rnorm(k, 0, sqrt(4 / n_per))
vi <- 4 / n_per

methods <- c("DL", "REML", "PM", "SJ", "EB")
tau2_est <- sapply(methods, function(m) {
  rma(yi, vi, method = m)$tau2
})
round(tau2_est, 3)
```

15.26.6 Output & Results

τ^2 estimates under the five methods. DL typically gives the smallest value; REML, PM, SJ, EB are often close.

15.26.7 Interpretation

"τ^2 estimates ranged from 0.041 (DL) to 0.061 (SJ); we report REML (τ^2 = 0.055, 95 % CI 0.012-0.142) as the primary estimator following modern meta-analysis recommendations."

15.26.8 Practical Tips

- Use REML by default; it is less biased than DL under most conditions.
- Hartung-Knapp adjustment (`test="knha"` in metafor) improves CI coverage, especially with few studies.
- Compare multiple estimators as sensitivity; if DL and REML disagree substantially, heterogeneity is likely real.
- Report 95 % CI on τ^2 (profile likelihood or Q-profile methods) alongside the point estimate.
- For very few studies ($k < 5$), any τ^2 estimate is unreliable; use Bayesian methods with informative priors.

15.26.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.26.10 See also — labs in this chapter

- Systematic reviews and PRISMA

- Meta-analysis basics
- Network meta-analysis

15.27 Trim-and-Fill

15.27.1 Introduction

Trim-and-fill, introduced by Duval and Tweedie in 2000, is the most widely used algorithmic approach for adjusting a pooled meta-analytic estimate when the funnel plot looks asymmetric. The idea is intuitive: if some smaller studies on the side opposite the dominant tail of the funnel appear to be missing — possibly because they were never published, never indexed, or never reached the systematic-review screening stage — the procedure imputes mirror-image counterparts and re-pools the augmented set. The difference between the unadjusted and adjusted estimates quantifies how sensitive the conclusion is to the presence of these hypothetical missing studies. Crucially, trim-and-fill is a sensitivity analysis, not a method of bias correction in the strict statistical sense; it cannot recover the true unbiased effect.

15.27.2 Prerequisites

Familiarity with funnel plots, the small-study-effect concept, the inverse-variance random-effects pool, and the distinction between sensitivity analysis and bias correction.

15.27.3 Theory

The algorithm proceeds iteratively:

1. Estimate the number of asymmetric studies in the dominant tail using one of three rank-based estimators (L_0, R_0, Q_0).
2. Trim those studies from the dominant side and re-estimate the pooled effect; this estimate becomes the new symmetry centre.
3. Fill the analysis by reflecting the trimmed studies across the new centre to the opposite side.
4. Re-pool with the augmented dataset to obtain the adjusted estimate.

Iterating to convergence yields the final imputed studies and adjusted pooled estimate. The number of imputed studies is itself an interpretable summary: zero imputations indicate no detectable asymmetry, and a large number indicates substantial sensitivity.

15.27.4 Assumptions

The observed funnel asymmetry is attributable to selection or publication bias rather than to true between-study heterogeneity correlated with study size; the

underlying effect distribution is mirror-symmetric around the true effect; and the pooled model is otherwise correctly specified.

15.27.5 R Implementation

```r
library(metafor)

set.seed(2026)
k <- 18
yi <- rnorm(k, 0.35, 0.25)
vi <- 1 / sample(15:300, k, replace = TRUE)

keep <- !(sqrt(vi) > 0.1 & abs(yi) < 0.15)
yi_b <- yi[keep]; vi_b <- vi[keep]

res <- rma(yi_b, vi_b, method = "REML")
tf  <- trimfill(res)

data.frame(method = c("Original", "Trim-and-fill"),
           estimate = round(c(res$b, tf$b), 3),
           ci = c(sprintf("[%.2f, %.2f]", res$ci.lb, res$ci.ub),
                  sprintf("[%.2f, %.2f]", tf$ci.lb, tf$ci.ub)))

funnel(tf, main = "Trim-and-fill adjusted funnel")
```

15.27.6 Output & Results

`trimfill()` returns the number of imputed studies, the adjusted pooled estimate with its confidence interval, and an augmented dataset suitable for plotting on a funnel diagram. Reporting both the original pool and the adjusted pool side by side — together with the count of imputations — lets readers see exactly how much the conclusion shifts when the missing studies are introduced.

15.27.7 Interpretation

A reporting sentence: "Trim-and-fill (Duval and Tweedie, L_0 estimator) imputed four potentially missing studies on the left side of the funnel; the adjusted standardised mean difference was 0.18 (95 % CI 0.03 to 0.33), notably attenuated from the unadjusted estimate of 0.32 (0.21 to 0.43). The result suggests the pooled conclusion is sensitive to plausible publication bias, and Egger's regression and PET-PEESE were reported as triangulating sensitivity analyses." Always report the unadjusted estimate as primary and the adjusted estimate as sensitivity.

15.27.8 Practical Tips

- Trim-and-fill is a sensitivity analysis, not a proof of publication bias; report it as such, and never use the adjusted estimate as the primary effect.
- The method performs poorly when funnel asymmetry is driven by genuine between-study heterogeneity correlated with study size; in such settings it can spuriously imply bias.
- Apply alongside Egger's regression test and PET-PEESE for triangulation; consistent signals strengthen the bias interpretation, divergent signals warn against over-interpretation.
- Report both original and adjusted estimates with their CIs and the count of imputed studies; an imputation count of zero indicates the funnel was already symmetric and no adjustment was needed.
- The L_0 estimator (default in `metafor`) is the most widely used; R_0 is more conservative; Q_0 is rarely used in practice. Sensitivity to estimator choice should be noted when results are borderline.
- Selection-model approaches (such as Vevea–Hedges weight-function models) and PET-PEESE are increasingly preferred over trim-and-fill in methodological literature; consider them when the small-study effect is consequential to your conclusion.

15.27.9 R Packages Used

`metafor::trimfill()` for the canonical implementation with all three estimators and `funnel()` overlay; `meta::trimfill()` for the alternative interface within the `meta` ecosystem; `metasens` for trim-and-fill alongside copas and limit-meta-analysis bias-adjustment methods; `weightr` and `puniform` for selection-model alternatives that handle complex bias mechanisms.

15.27.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

15.27.11 See also — labs in this chapter

- Systematic reviews and PRISMA
- Meta-analysis basics
- Network meta-analysis

15.28 Week 4, Session 2 — Meta-analysis basics

Inference lab: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

15.28.1 Learning objectives

- Fit a random-effects meta-analysis with `metafor::rma`.
- Produce forest and funnel plots, and interpret between-study heterogeneity (τ^2, I^2).
- Assess small-study effects and publication bias informally.

15.28.2 Prerequisites

Comfort with effect sizes on the log scale.

15.28.3 Background

Meta-analysis pools effect estimates across studies to sharpen an overall answer. Fixed-effect models assume all studies estimate one true effect; random-effects models assume true effects vary around a common mean. In biomedical research, heterogeneity is the rule rather than the exception, so random effects are the default — and between-study variance (τ^2) and the proportion of total variance due to heterogeneity (I^2) become part of the primary reporting, not an afterthought.

Forest plots display each study's estimate with its confidence interval next to the pooled estimate. Funnel plots display study precision against effect size; asymmetry hints at small-study effects and potential publication bias. Neither plot settles the question — Egger's test, trim-and-fill, and careful inspection of which small studies are missing are all part of the routine.

15.28.4 Setup

```
library(tidyverse)
library(metafor)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

15.28.5 1. Hypothesis

H_0: pooled log risk ratio $= 0$ (no overall effect). H_1: pooled log RR $\neq 0$. $\alpha = 0.05$.

15.28.6 2. Visualise — use a built-in dataset

```
dat <- escalc(measure = "RR",
              ai = tpos, bi = tneg, ci = cpos, di = cneg,
              data = dat.bcg, slab = paste(author, year))
```

```
forest(rma(yi, vi, data = dat), header = TRUE, cex = 0.8)
```

15.28.7 3. Assumptions

- Between-study heterogeneity exists ($\tau^2 > 0$) in virtually every real meta-analysis — a random-effects model absorbs it, a fixed-effect model does not.
- Funnel-plot asymmetry does not prove publication bias; it is consistent with several mechanisms.

15.28.8 4. Conduct

```
fit <- rma(yi, vi, data = dat, method = "REML")
fit
```

```
funnel(fit, main = "Funnel plot — BCG vaccine dataset")
regtest(fit, model = "lm")
```

15.28.9 5. Concluding statement

```
rr   <- exp(fit$b[1])
lo   <- exp(fit$ci.lb)
hi   <- exp(fit$ci.ub)
```

> Across `nrow(dat)` trials of the BCG vaccine, the pooled risk ratio for tuberculosis was `round(rr,2)` (95% CI `round(lo,2)` to `round(hi,2)`; τ^2 = `round(fit$tau2,2)`, I^2 = `round(fit$I2,1)`%). Between-trial heterogeneity was substantial; the pooled estimate should be interpreted with the heterogeneity statistics in view.

15.28.10 Common pitfalls

- Fixed-effect pooling when between-study heterogeneity is real.
- Quoting a pooled effect without its heterogeneity measures.
- Treating funnel-plot symmetry as a formal test rather than a visual prompt.

15.28.11 Further reading

- Viechtbauer W. (2010). *Conducting meta-analyses in R with the metafor package.* J Stat Softw.
- Higgins JPT et al. *Cochrane Handbook for Systematic Reviews of Interventions.*

15.28.12 Session info

```
sessionInfo()
```

15.28.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

15.29 Week 4, Session 3 — Network meta-analysis

Inference lab: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

15.29.1 Learning objectives

- Fit a network meta-analysis (NMA) using `netmeta`.
- Draw and read the network graph.
- Derive SUCRA rankings and interpret them cautiously.

15.29.2 Prerequisites

Pairwise meta-analysis (W4 S2).

15.29.3 Background

Network meta-analysis extends pairwise meta-analysis to more than two treatments by combining direct comparisons (from head-to-head trials) with indirect comparisons (inferred through a shared comparator). The central assumption is *transitivity*: the studies compared indirectly are exchangeable — no moderator distinguishes them in a way that systematically shifts the effect. The statistical analogue, *consistency*, checks whether direct and indirect estimates agree.

SUCRA (surface under the cumulative ranking curve) summarises each treatment's rank distribution on a 0-1 scale, where 1 is best. SUCRA values compress a lot of uncertainty into a single number; they are useful for signalling but should be reported with interval estimates for every pairwise comparison.

15.29.4 Setup

```
library(tidyverse)
library(netmeta)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

15.29.5 1. Hypothesis

H_0: all treatments in the network have equal efficacy. H_1: at least two differ.

15.29.6 2. Visualise

```
data(Senn2013)
head(Senn2013)

net <- netmeta(TE, seTE, treat1, treat2, studlab,
               data = Senn2013, sm = "MD",
               common = FALSE, random = TRUE,
               reference.group = "plac")
```

```
netgraph(net, plastic = FALSE, thickness = "number.of.studies",
         points = TRUE, cex.points = 2, cex = 0.8)
```

15.29.7 3. Assumptions

```
decomp.design(net)
```

The within-design heterogeneity and between-design inconsistency variances are the key diagnostics.

15.29.8 4. Conduct

```
summary(net)
rank_df <- netrank(net, small.values = "good")
rank_df
```

15.29.9 5. Concluding statement

```
ranked <- as_tibble(rank_df$ranking.random, rownames = "treatment") |>
  rename(sucra = value) |>
  arrange(desc(sucra))
top <- head(ranked, 1)
```

In a network meta-analysis of `length(unique(c(Senn2013$treat1,Senn2013$treat2)))` treatments across `length(unique(Senn2013$studlab))` studies (`nrow(Senn2013)` pairwise comparisons), the treatment with the highest SUCRA for HbA1c lowering was **`top$treatment`** (SUCRA = `round(top$sucra,2)`). SUCRA rankings should be interpreted alongside the pairwise effect estimates and their intervals, not as a standalone conclusion.

15.29.10 Common pitfalls

- Treating SUCRA as if it were the effect size.
- Ignoring the transitivity assumption because the output compiled.
- Mixing direct and indirect evidence without quantifying inconsistency.

15.29.11 Further reading

- Rücker G, Schwarzer G. (2015). *Ranking treatments in frequentist network meta-analysis.*
- Salanti G. (2012). *Indirect and mixed-treatment comparison…*

15.29.12 Session info

```
sessionInfo()
```

15.29.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

15.30 Week 4, Session 1 — Systematic reviews and PRISMA

Workflow lab: Goal → Approach → Execution → Check → Report.

15.30.1 Learning objectives

- Frame a systematic-review question using PICO.
- Draft a reproducible search strategy across at least two databases.
- Produce a PRISMA flow diagram from counts at each screening stage.

15.30.2 Prerequisites

None beyond reading a paper.

15.30.3 Background

A systematic review is itself a study: it has a protocol, a pre-specified search strategy, explicit inclusion and exclusion criteria, and a plan for synthesis. The PRISMA 2020 statement is the reporting standard, which means the diagram showing what you searched, what you excluded, and why is not optional. PROSPERO (https://www.crd.york.ac.uk/prospero/) is the standard registry for review protocols — registering early protects you from post-hoc drift.

Good search strategies combine controlled-vocabulary terms (MeSH on PubMed, Emtree on Embase) with free-text terms, connect concepts with Boolean AND, and expand synonyms with OR. A librarian-reviewed strategy typically outperforms a DIY one by a wide margin in recall; co-authoring with a health-sciences librarian is the single best investment in review quality.

15.30.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

15.30.5 1. Goal

Illustrate a PICO-framed question, sketch a search strategy, and simulate screening counts to generate a PRISMA flow diagram.

15.30.6 2. Approach

A fictional question in PICO form:

> In adults with type 2 diabetes (**P**), does metformin (**I**) compared with placebo (**C**) reduce all-cause mortality (**O**) at 24 months?

A sketch of a search strategy (PubMed syntax):

```
("Diabetes Mellitus, Type 2"[MeSH] OR "type 2 diabetes"[tiab])
AND (metformin[MeSH] OR metformin[tiab])
AND (randomised[tiab] OR randomized[tiab] OR trial[tiab] OR RCT[tiab])
```

15.30.7 3. Execution — simulated screening counts

```
prisma <- tibble(
  stage = c("Identified (PubMed)", "Identified (Embase)",
            "After deduplication",
            "Title/abstract screened", "Full-text assessed",
            "Included in qualitative synthesis",
            "Included in meta-analysis"),
```

```
  n = c(812, 640, 1098, 1098, 84, 22, 18)
)
prisma
```

```
prisma |>
  mutate(stage = factor(stage, levels = rev(stage))) |>
  ggplot(aes(n, stage)) +
  geom_col(fill = "#1a73e8", alpha = 0.8) +
  geom_text(aes(label = n), hjust = -0.1, size = 3.5) +
  scale_x_continuous(expand = expansion(mult = c(0, 0.15))) +
  labs(x = "n", y = NULL)
```

15.30.8 4. Check

PRISMA 2020 checklist pointer (items most often missed):

- Item 5: eligibility criteria pre-specified.
- Item 6: information sources with date of last search.
- Item 7: complete search strategy for every database, including limits.
- Item 16a: reasons for excluding full-text articles.

15.30.9 5. Report

> A systematic review of the effect of metformin on all-cause mortality in adults with type 2 diabetes was conducted following PRISMA 2020. Database searches identified 812 records from PubMed and 640 from Embase. After deduplication, 1098 records were screened by title and abstract; 84 full texts were assessed, and 22 studies were included in the qualitative synthesis, of which 18 were pooled by meta-analysis.

15.30.10 Common pitfalls

- Ad-hoc searches that cannot be re-run.
- Failing to double-screen at each stage (reviewer drift).
- Reporting included-study counts without the flow that produced them.

15.30.11 Further reading

- Page MJ et al. (2021). *The PRISMA 2020 statement.* BMJ.
- Cochrane Handbook for Systematic Reviews of Interventions.

15.30.12 Session info

```
sessionInfo()
```

15.30.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 16

Experimental Design

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

Study design across the spectrum: observational designs and STROBE, randomised trials, missing-data mechanisms and multiple imputation, mixed-effects models, DAGs and confounding, propensity scores, the g-methods family (IV, DiD, RDD, HTE), compartmental models for outbreaks, and the pre-registration / SAP / replication-crisis literature.

This chapter contains **30 method pages** and **13 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

16.1 Method pages

Method	Source slug
2^k Factorial Designs	`factorial-2k`
3^k Factorial Designs	`factorial-3k`
Balanced Incomplete Block Designs	`balanced-incomplete-block`
Box-Behnken Designs	`box-behnken-design`
Central Composite Designs	`central-composite-design`
Completely Randomised Design	`completely-randomized-design`
Constrained Mixture Designs	`constrained-mixture`
Crossover Designs	`crossover-designs`
Desirability Functions	`desirability-function`
Fractional Factorial Designs	`fractional-factorial-design`
Graeco-Latin Square Designs	`graeco-latin-square`
Latin Square Designs	`latin-square`
Method of Steepest Ascent	`steepest-ascent`
Optimal Designs: D, A, I	`optimal-design-d-a-i`

Method	Source slug
Orthogonal Arrays	`orthogonal-arrays`
Plackett-Burman Designs	`plackett-burman`
Power Analysis for DOE	`power-in-doe`
Principles of Blocking	`blocking-principles`
Randomisation in Design	`randomization-in-design`
Randomised Complete Block Design	`randomised-complete-block`
Repeated-Measures Designs	`repeated-measures-design`
Resolution and Aliasing	`resolution-aliasing`
Response Surface Methodology	`response-surface-methodology`
Robust Parameter Design	`robust-parameter-design`
Signal-to-Noise Ratios	`signal-to-noise-ratio`
Simplex Centroid Designs	`mixture-centroid`
Simplex Lattice Mixture Designs	`mixture-designs-simplex`
Split-Plot Designs	`split-plot-design`
Strip-Plot Designs	`strip-plot-design`
Taguchi Methods	`taguchi-methods`

16.2 Labs

Lab
Observational designs and STROBE
Randomised controlled trials
Adaptive, non-inferiority, equivalence trials
Bench and translational design
MCAR, MAR, MNAR
Multiple imputation with mice
Linear mixed models with lme4
GLMMs and GEE
DAGs with dagitty and ggdag
Propensity scores and IPTW
G-methods, IV, DiD, RDD; heterogeneity of treatment effect
SIR and SEIR models with deSolve
Pre-registration and statistical analysis plans

16.3 Balanced Incomplete Block Designs

16.3.1 Introduction

Balanced incomplete block designs (BIBDs) allocate treatments to blocks when each block can hold fewer experimental units than the total number of treatments to be compared. The constraint arises constantly in practice — a litter of

animals contains fewer pups than treatments, a panel of taste-testers has limited capacity per session, a microscopy slide accommodates only a few specimens, an oven processes a limited number of samples per run. The "balanced" property requires that every pair of treatments appears together in the same number λ of blocks, ensuring that every pairwise treatment contrast is estimated with equal precision despite the block-level incompleteness. BIBDs are classical constructions in agricultural experimentation, sensory evaluation, and any setting where complete randomised blocks are physically impossible.

16.3.2 Prerequisites

A working understanding of randomised complete block designs, combinatorial design parameters, and basic mixed-effects modelling for the recovery of inter-block information.

16.3.3 Theory

A BIBD is specified by parameters (v, b, r, k, λ):

- v = number of treatments,
- b = number of blocks,
- r = replicates per treatment,
- k = block size (with $k < v$),
- λ = number of blocks in which each treatment pair co-occurs.

These parameters satisfy two necessary identities: $bk = vr$ and $\lambda(v-1) = r(k-1)$. Not every parameter combination corresponds to an existing BIBD; Fisher's inequality requires $b \geq v$, and the Bruck–Ryser–Chowla theorem provides further necessary conditions. The classical analysis fits blocks as either fixed or random effects; treating blocks as random recovers inter-block information and is generally more efficient.

16.3.4 Assumptions

Treatment pairs are exchangeable via the balance property, within-block residuals are exchangeable and approximately Normal with constant variance, and block effects are additive (no block × treatment interaction). The design is implemented as planned — running fewer or more units than specified destroys the balance and requires reanalysis as a partially balanced or unbalanced incomplete block design.

16.3.5 R Implementation

```
library(crossdes)

bibd <- find.BIB(7, 7, 3)
```

```
bibd

set.seed(2026)
df <- data.frame(
  block = rep(1:7, each = 3),
  treatment = factor(as.vector(t(bibd)))
)
df$y <- rnorm(nrow(df), mean = c(10, 11, 12, 13, 14, 15,
 ↪  16)[as.integer(df$treatment)],
              sd = 0.8)

fit <- aov(y ~ factor(block) + treatment, data = df)
summary(fit)
```

16.3.6 Output & Results

`find.BIB()` constructs a BIBD with the specified parameters when one exists. The fitted ANOVA partitions variability into block, treatment, and residual components, with the treatment F-test computed using the intra-block error. Reporting the proportion of variance absorbed by blocks alongside the treatment effect characterises the gain from incomplete blocking.

16.3.7 Interpretation

A reporting sentence: "The $(v, b, r, k, \lambda) = (7, 7, 3, 3, 1)$ BIBD — equivalent to the Fano plane in projective geometry — tested seven treatments in seven blocks of size three. ANOVA showed a significant treatment effect ($F_{6,8} = 8.2$, $p = 0.002$) with block-level variance absorbed by the block factor (block sum of squares 12.4, residual sum of squares 4.7). All 21 pairwise treatment contrasts were estimated with equal precision because of the balance property." Always report the BIBD parameters explicitly.

16.3.8 Practical Tips

- Use a BIBD whenever block size is smaller than the treatment count and balanced pairwise comparisons are required; the balance property ensures every contrast has the same standard error, which simplifies interpretation and multiple-comparison adjustment.
- Analyse with blocks as **random effects** (`(1|block)` in `lme4`) to recover inter-block information; the random-effects analysis is generally more efficient than the fixed-effect intra-block analysis, especially when the number of blocks is large.
- Not all parameter combinations (v, k, λ) correspond to an existing BIBD; consult catalogue tables or use `crossdes::find.BIB()` to check existence

before finalising the design. Fisher's inequality and the Bruck–Ryser–Chowla theorem give the necessary conditions.
- For large treatment counts where a BIBD is impractical, partially balanced incomplete block designs (PBIBDs) relax the equal-pairwise-co-occurrence requirement at the cost of unequal pairwise precision; they are far easier to find and adequate for most applied work.
- Pre-specify the block allocation in the protocol; changing mid-experiment — adding a unit, dropping a block — destroys the balance property and forces reanalysis as a general unbalanced design.
- Visualise the incidence matrix (treatment × block) before running the experiment; an inadvertent imbalance (a treatment missing from a block, a duplicated treatment within a block) is far easier to spot at the design stage than after data collection.

16.3.9 R Packages Used

`crossdes::find.BIB()` and `crossdes::isGYD()` for BIBD construction and verification; `agricolae::design.bib()` for an alternative interface integrated with the agricultural-DoE toolkit; `lme4::lmer()` for random-block analysis recovering inter-block information; `ibd` for advanced incomplete-block constructions including PBIBDs; `daewr` for textbook companion datasets.

16.3.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.3.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.4 Principles of Blocking

16.4.1 Introduction

Blocking is the experimental-design technique that groups experimental units by a known source of nuisance variation — batch of raw material, day of measurement, operator, animal litter, plot location — and then randomises treatment assignments within each block. The result is a design that systematically removes block-to-block variation from the error term, thereby shrinking the residual variance and substantially improving the power to detect genuine treatment effects. Fisher's classical mantra — "block what you can, randomise what you cannot" — captures the principle: when a nuisance source of variation is known and identifiable, controlling it through design is far more efficient than relying on randomisation alone to average it out.

16.4.2 Prerequisites

A working understanding of randomisation as the basis of comparative inference, the partition of total variability into systematic and residual components in ANOVA, and the distinction between fixed and random effects in mixed-effects models.

16.4.3 Theory

In its simplest form — the randomised complete block design (RCBD) — the model is

$$y_{ij} = \mu + \rho_i + \tau_j + \varepsilon_{ij}, \qquad \varepsilon_{ij} \sim N(0, \sigma^2),$$

with block effect ρ_i and treatment effect τ_j. Within-block variance is typically much smaller than between-block variance, so absorbing the latter into the design removes a large component from the residual error. The relative efficiency of an RCBD versus a CRD is roughly the ratio of the residual variances; gains of an order of magnitude are not unusual when blocking is well chosen.

16.4.4 Assumptions

Blocks represent exchangeable groupings of units, no block-by-treatment interaction is present (or, if it is, the interaction is modelled — typically as a random effect), and within-block residual variance is homogeneous. Violations of additivity invalidate the standard analysis but can be detected with a Tukey one-degree-of-freedom test for non-additivity.

16.4.5 R Implementation

```
set.seed(2026)
block <- factor(rep(1:4, each = 3))
trt <- factor(rep(c("A", "B", "C"), 4))

block_re <- rnorm(4, 0, 2)[as.integer(block)]
trt_eff  <- c(0, 1.5, 2.5)[as.integer(trt)]
y <- 10 + block_re + trt_eff + rnorm(12, 0, 0.5)

df <- data.frame(block, trt, y)

fit_b <- aov(y ~ block + trt, data = df)
fit_nb <- aov(y ~ trt, data = df)

c(with_block_MSE = summary(fit_b)[[1]]["Residuals", "Mean Sq"],
  without_block_MSE = summary(fit_nb)[[1]]["Residuals", "Mean Sq"])
```

16.4.6 Output & Results

The script reports residual mean-square error with and without blocking. When the block-to-block standard deviation is comparable to or larger than the within-block standard deviation, blocking produces dramatic reductions in residual MSE and correspondingly large increases in the F-statistic for the treatment effect. The same point can be visualised by comparing residual diagnostic plots from the two fits.

16.4.7 Interpretation

A reporting sentence: "Blocking on production batch reduced residual mean-square error from 3.8 to 0.24, a 16-fold reduction. The treatment F-ratio increased from $F_{2,9} = 1.1$ ($p = 0.37$) without blocking to $F_{2,6} = 17.3$ ($p = 0.003$) with blocking, demonstrating that batch was the dominant nuisance source and that blocking restored the detection power lost to batch-level drift." Always report residual MSE before and after blocking when justifying the design choice.

16.4.8 Practical Tips

- Block first on the largest known source of unwanted variation: temporal (day, week, batch), spatial (plot, shelf, freezer), or operator-related (technician, machine, lot). Unblocked sources contribute to the residual; blocked sources are absorbed into the design.
- One block per replicate is the standard RCBD; multiple blocking factors call for Latin square (two factors) or Graeco-Latin square (three factors) layouts that retain orthogonality.

- Incomplete blocks (block size smaller than the number of treatments) require balanced incomplete block designs (BIBD) or partially balanced incomplete blocks (PBIBD); the analysis is more involved but the principle is the same.
- Treat blocks as random effects (`(1|block)` in `lme4`) when blocks are sampled from a larger population of possible blocks, as is typical for batches, animals, or experimental days; this also recovers inter-block information that the fixed-effect analysis discards.
- Do not block on a variable you actually want to study: if the nuisance factor is itself of interest, treat it as a fixed treatment factor in a factorial design instead. Blocking removes inferential interest in the blocked factor.
- Incomplete or unbalanced block sizes are increasingly common in clinical trials (centre-stratified randomisation) and call for mixed-effects models rather than classical fixed-effect ANOVA.

16.4.9 R Packages Used

Base R `aov()` for fixed-effect RCBD analysis; `lme4` and `lmerTest` for random-block mixed-effects models with Satterthwaite or Kenward–Roger denominator df; `agricolae` for `design.rcbd()`, `BIB.test()`, and related agricultural design tools; `emmeans` for marginal means and contrasts under blocking; `crossdes` for systematic construction of incomplete-block layouts.

16.4.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.4.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.5 Box-Behnken Designs

16.5.1 Introduction

Box-Behnken designs (BBD), introduced by Box and Behnken in 1960, are rotatable three-level response-surface designs whose distinguishing feature is that they never run the extreme-corner combinations of the factorial cube. Each factor is set at one of three coded levels (-1, 0, or $+1$), but every experimental run leaves at least one factor at the centre level — so no run combines all factors at their most extreme settings simultaneously. This makes the design particularly useful when extreme corner conditions are infeasible (safety limits, equipment constraints, prohibitive cost, or unstable chemistry at extremes), while still allowing estimation of the full second-order response-surface model with main effects, quadratic terms, and two-factor interactions.

16.5.2 Prerequisites

A working understanding of factorial designs, response-surface methodology (RSM), the second-order polynomial model, and the concept of design rotatability — that prediction variance depends only on distance from the design centre, not on direction.

16.5.3 Theory

For k factors a Box-Behnken design places each factor at one of $\{-1, 0, +1\}$, with runs taken only at combinations where two factors are at ± 1 and all remaining factors are at the centre level 0. Replicated centre runs provide pure-error estimation and a lack-of-fit test. For $k = 3$ a BBD requires 12 factorial-style runs plus typically 3 centre points (15 runs total), comparable in size to a central composite design (CCD) with 14–20 runs.

BBDs are exactly rotatable for $k = 4$ and $k = 7$; for other values of k they are nearly rotatable. The absence of axial (star) points means prediction near the cube's corners is extrapolation with elevated variance.

16.5.4 Assumptions

A second-order polynomial response surface is adequate over the design region (no strong curvature beyond quadratic), corner-extreme runs are either infeasible or unnecessary, and the residual error is approximately Normal with constant variance. The pure-error replications at the centre allow these assumptions to be diagnosed.

16.5.5 R Implementation

```
library(rsm)
```

```
bbd <- bbd(~ x1 + x2 + x3, n0 = 3, randomize = FALSE)
bbd

set.seed(2026)
bbd$y <- 10 + 2 * bbd$x1 + 1.5 * bbd$x2 - 1 * bbd$x3 -
         0.9 * bbd$x1^2 + 0.5 * bbd$x1 * bbd$x2 +
         rnorm(nrow(bbd), 0, 0.4)

fit <- rsm(y ~ SO(x1, x2, x3), data = bbd)
summary(fit)
```

16.5.6 Output & Results

`bbd()` constructs the layout; the `rsm()` second-order model returns coefficients for all linear, quadratic, and two-factor-interaction terms together with a lack-of-fit test against the pure-error replicates. The canonical analysis includes a stationary-point analysis and an eigenvalue decomposition that classifies the surface as a maximum, minimum, or saddle.

16.5.7 Interpretation

A reporting sentence: "The 15-run Box-Behnken design with 3 centre points estimated all linear, quadratic, and pairwise-interaction effects of the three factors. The fit revealed dominant curvature in x_1 (quadratic coefficient -0.9, $p < 0.001$) with significant linear effects on all three factors and one interaction ($x_1 \times x_2 = 0.5$, $p = 0.03$). The lack-of-fit test was non-significant ($p = 0.42$), supporting the second-order model. Predicted maximum lies at $x_1 = 1$, $x_2 = 1$, $x_3 = -1$." Always report the lack-of-fit test outcome.

16.5.8 Practical Tips

- Prefer BBD over CCD when corner combinations are infeasible (extreme operating conditions, safety limits, equipment constraints, regulatory bounds, or unstable chemistry at extremes); BBD is the design of choice in such settings.
- BBD has no axial points and so does not extend beyond the cube; prediction near or beyond the cube corners is extrapolation with elevated variance, and the design region should be chosen with this in mind.
- Centre-point replications are essential — they provide pure-error estimation and the lack-of-fit test that diagnoses inadequacy of the second-order model. Three to five replicated centre points is the typical compromise between cost and inferential value.
- BBD is generally less efficient than CCD in prediction variance over the cube; the choice between them is driven by feasibility of corner runs and

the practical cost of extra axial-point experimentation.
- BBD is not defined for $k = 2$; for two-factor RSM, use a central composite design or a face-centred CCD.
- For mixed-precision designs (some factors continuous, others two-level discrete), `rsm` can fit hybrid models, but pre-specified screening followed by RSM follow-up is usually clearer.

16.5.9 R Packages Used

`rsm::bbd()` and `rsm::rsm()` for the canonical Box-Behnken construction and second-order analysis with stationary-point and canonical analysis; `DoE.base` for general orthogonal-array DoE construction; `daewr` for textbook companion examples; `qualityTools` for an alternative RSM toolkit including BBD; `Vdgrap h` for variance-dispersion graphs comparing BBD to CCD.

16.5.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.5.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.6 Central Composite Designs

16.6.1 Introduction

The central composite design (CCD), introduced by Box and Wilson in 1951, is the workhorse of response-surface methodology and arguably the most influ-

ential design construction in industrial statistics. It augments a 2^k factorial with axial (or star) points placed along each factor axis at distance α from the design centre and a number of replicated centre points. Together these three components — factorial, axial, centre — provide enough information to fit a full second-order polynomial response surface in k factors with substantially fewer runs than a 3^k full factorial would require, while still supporting estimation of curvature, lack-of-fit testing, and prediction-variance characterisation across the design region.

16.6.2 Prerequisites

A working understanding of two-level factorial designs, polynomial regression for response-surface modelling, and the concept of design rotatability — that prediction variance depends only on radial distance from the centre.

16.6.3 Theory

A CCD has three components:

- **Factorial**: 2^k runs at the corners of the unit cube (± 1 in each factor).
- **Axial**: $2k$ runs at $(\pm\alpha, 0, \ldots, 0), (0, \pm\alpha, 0, \ldots, 0), \ldots, (0, \ldots, 0, \pm\alpha)$.
- **Centre**: replicated $(0, 0, \ldots, 0)$ runs supplying pure-error estimation.

The choice of α determines the design's properties: rotatable CCDs use $\alpha = \sqrt[4]{2^k}$ (giving constant prediction variance at fixed distance from the centre), face-centred CCDs use $\alpha = 1$ (staying within the original ± 1 cube, useful when factor extremes are constrained), and orthogonal CCDs choose α to make the columns of the second-order design matrix orthogonal.

16.6.4 Assumptions

The response is well modelled by a second-order polynomial within the design region, the residual error is approximately Normal with constant variance, and the centre-point replicates provide a valid pure-error estimate against which lack-of-fit can be tested.

16.6.5 R Implementation

```
library(rsm)

ccd <- ccd(~ x1 + x2, n0 = c(3, 3), alpha = "rotatable",
           randomize = FALSE)
ccd

set.seed(2026)
ccd$y <- 10 + 2 * ccd$x1 + 1.5 * ccd$x2 - 1.2 * ccd$x1^2 -
```

```
        0.8 * ccd$x2^2 + 0.6 * ccd$x1 * ccd$x2 +
        rnorm(nrow(ccd), 0, 0.5)

fit <- rsm(y ~ SO(x1, x2), data = ccd)
summary(fit)

steepest(fit, dist = 1.0)
```

16.6.6 Output & Results

`ccd()` constructs the layout, and `rsm()` fits the second-order model with stationary-point analysis, eigenvalue decomposition, and a lack-of-fit test against the pure-error replicates. The `steepest()` function traces the path of steepest ascent, useful when the stationary point lies outside the design region and follow-up experimentation is needed in the optimal direction.

16.6.7 Interpretation

A reporting sentence: "The 14-run rotatable CCD with three centre points identified a concave response surface with stationary point at $(x_1, x_2) = (0.83, 0.69)$, predicted optimum 12.3. Quadratic terms in both factors were significant ($p < 0.01$), confirming curvature; the interaction term was marginal ($p = 0.07$). The lack-of-fit test was non-significant ($p = 0.31$), supporting the second-order model. Path-of-steepest-ascent analysis was unnecessary because the optimum lay inside the design region." Always run a lack-of-fit test and report the canonical-form classification.

16.6.8 Practical Tips

- Rotatable CCDs give equal prediction variance at fixed distance from the centre and are the default RSM choice; the rotatability property is what makes CCDs ideal for sequential exploration of the response surface.
- Face-centred CCDs ($\alpha = 1$) are useful when factor levels cannot exceed the original cube — common when factor ranges are determined by safety or feasibility limits — at the cost of slightly reduced prediction-variance properties.
- Include 3–5 replicated centre points for pure-error degrees of freedom and curvature testing; fewer than 3 leaves the lack-of-fit test underpowered.
- After fitting, use `contour()` and `persp()` plots to visualise the response surface; engineers and scientists overwhelmingly prefer surface plots to coefficient tables when interpreting RSM results.
- Follow up the second-order fit with canonical analysis to classify the stationary point as a maximum, minimum, or saddle (eigenvalue signs of the quadratic-form matrix); a saddle point usually means the optimum lies outside the design region and additional runs are warranted.

- For more than three factors, consider sequential experimentation: a fractional factorial first to screen for active factors, then a CCD or BBD on the active subset to characterise curvature.

16.6.9 R Packages Used

`rsm::ccd()` and `rsm::rsm()` for canonical CCD construction, second-order fitting, canonical analysis, and steepest-ascent paths; `qualityTools::rsmDesign()` for an alternative interface; `Vdgraph` for variance-dispersion graphs comparing rotatable, face-centred, and orthogonal CCD variants; `daewr` for textbook companion datasets and `DoE.wrapper` for higher-level DoE workflow integration.

16.6.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.6.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.7 Completely Randomised Design

16.7.1 Introduction

The completely randomised design (CRD) is the simplest and most flexible of all comparative experimental designs: each of the available experimental units is assigned to one of several treatment groups by an unrestricted random mechanism, with no blocking, stratification, or other restriction on randomisation. CRD is the default starting point in any introductory experimental-design course and

remains the right choice whenever the experimental units are reasonably homogeneous and no nuisance source of variation justifies the additional complexity of a blocked layout. Its analysis is one-way ANOVA, its assumptions are minimal, and its results are easily generalised to whatever target population the units were sampled from.

16.7.2 Prerequisites

A working understanding of randomisation as the foundation of causal inference in experiments, one-way analysis of variance with the F-test, and the multiple-comparisons problem that arises when more than two treatment groups are compared.

16.7.3 Theory

With a treatment levels and r replicates per treatment, $n = a \cdot r$ experimental units are randomly allocated to treatments without any restriction. The standard analysis model is

$$y_{ij} = \mu + \tau_i + \varepsilon_{ij}, \qquad \varepsilon_{ij} \sim N(0, \sigma^2),$$

with τ_i the fixed effect of the i-th treatment and ε_{ij} independent Normal error. The omnibus null hypothesis is $\tau_1 = \cdots = \tau_a$, tested by the one-way ANOVA F-statistic. Post-hoc pairwise comparisons (Tukey HSD, Dunnett against a control, Scheffé for arbitrary contrasts) follow when the omnibus test is significant and pairwise inference is needed.

16.7.4 Assumptions

The experimental units are exchangeable prior to randomisation (homogeneous), the errors are independent within and across treatment groups, the within-group residuals are approximately Normal, and the residual variance is the same in each group. Departures from variance homogeneity can usually be addressed by transformation, weighted ANOVA, or Welch's F-test.

16.7.5 R Implementation

```
set.seed(2026)
a <- 4; r <- 6
treatment <- factor(rep(LETTERS[1:a], each = r))
y <- rnorm(a * r, mean = rep(c(10, 12, 11, 14), each = r), sd = 2)

df <- data.frame(treatment, y)
fit <- aov(y ~ treatment, data = df)
summary(fit)
```

```
TukeyHSD(fit)
```

16.7.6 Output & Results

`aov()` returns the one-way ANOVA partitioning of the total sum of squares into between-group and within-group components, with the omnibus F-test reported as the primary inference. `TukeyHSD()` provides simultaneous 95 % confidence intervals for all pairwise differences, controlling family-wise error rate at the chosen α, and is the standard follow-up when the omnibus null is rejected.

16.7.7 Interpretation

A reporting sentence: "The one-way ANOVA showed significant treatment differences ($F_{3,20} = 5.9$, $p = 0.005$); Tukey HSD identified treatment D as significantly higher than treatment A (mean difference 3.7, adjusted $p = 0.004$, 95 % CI 1.2 to 6.2). Other pairwise contrasts did not reach the adjusted $\alpha = 0.05$ level. Residual diagnostics (Q-Q plot, Levene's test $p = 0.41$) supported the Normal-equal-variance assumptions." Always pair the omnibus test with adjusted pairwise comparisons and a brief assumption-check statement.

16.7.8 Practical Tips

- CRD is most efficient when experimental units are reasonably homogeneous; if a substantial source of nuisance variation is known in advance, a randomised complete block design will reduce error variance and increase power without changing the inferential framework.
- Balanced designs with equal r across groups are optimal in efficiency and most robust to small departures from variance homogeneity; modest imbalance is acceptable but complicates type-III sum-of-squares interpretation.
- Always check residual diagnostics: a Q-Q plot, residuals-versus-fitted plot, and Levene's or Brown–Forsythe test for variance homogeneity. Mild departures rarely affect the omnibus F-test, but severe departures motivate a transformation or robust alternative.
- For markedly non-Normal responses, log or square-root transformations often suffice; for ordinal or otherwise non-transformable outcomes, use Kruskal–Wallis as a non-parametric omnibus and Dunn's procedure for post-hoc comparisons.
- Document the randomisation procedure (seed, software, date) so that the assignment list can be reconstructed from the protocol — a basic reproducibility requirement that auditors increasingly demand.
- If interest centres on a specific contrast (treatment vs control), Dunnett's procedure is more powerful than Tukey HSD and should be pre-specified in the protocol.

16.7.9 R Packages Used

Base R `aov()`, `lm()`, and `TukeyHSD()` cover the canonical CRD analysis; `agricolae` for `LSD.test()` and Duncan-style multiple-comparison tools used in agricultural DoE; `multcomp` for Dunnett, Scheffé, and arbitrary linear contrasts with adjustment; `emmeans` for marginal-mean estimation, plotting, and flexible contrast construction across designs.

16.7.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.7.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.8 Constrained Mixture Designs

16.8.1 Introduction

In many real-world mixture experiments, each component carries individual lower and upper bounds — salt must remain at least 5 %, alcohol must not exceed 15 %, an active pharmaceutical ingredient must be between 2 and 8 % — so the feasible blend region is no longer the full simplex but a constrained convex polytope inside it. Standard simplex-lattice and simplex-centroid designs do not exploit this structure and would place runs outside the feasible region. Extreme-vertices designs (McLean and Anderson, 1966) instead enumerate the vertices of the constraint polytope and sample them, augmenting with edge midpoints, face centroids, and an overall centroid as the design budget allows. The result

is a design that is simultaneously feasible everywhere and informative about the response across the full constrained region.

16.8.2 Prerequisites

A working understanding of simplex-lattice and simplex-centroid mixture designs, the Scheffé canonical polynomial, and basic convex-geometry concepts (vertices, edges, polytopes).

16.8.3 Theory

Given lower bounds L_i and upper bounds U_i on each of the q components, the feasible region is the convex polytope $\{x : L_i \le x_i \le U_i, \sum_i x_i = 1\}$. Its vertices are computable from the constraint set and form the natural design points; the design is augmented with edge midpoints, face centroids, and the overall centroid as needed for fitting the chosen Scheffé polynomial order.

The pseudo-component transformation $x_i' = (x_i - L_i)/(1 - \sum_i L_i)$ maps the constrained region back to a standard unit simplex when only lower bounds are active, allowing standard simplex-lattice or simplex-centroid construction in the transformed coordinates. Predictions can then be back-transformed to the original-component scale.

16.8.4 Assumptions

Components are non-negative and sum to one (they are proportions of a fixed total amount), the lower and upper bounds are fixed and known in advance, and the residual error is approximately Normal with constant variance over the feasible region.

16.8.5 R Implementation

```
library(mixexp)

lb <- c(0.2, 0.1, 0.3)
ub <- c(0.5, 0.6, 0.7)

xv <- Xvert(nfac = 3, lc = lb, uc = ub, ndm = 2)
xv

set.seed(2026)
xv$y <- 10 * xv$x1 + 12 * xv$x2 + 8 * xv$x3 +
        6 * xv$x1 * xv$x2 +
        rnorm(nrow(xv), 0, 0.3)

fit <- MixModel(xv, "y",
```

```
                mixcomps = c("x1", "x2", "x3"), model = 2)
summary(fit)
```

16.8.6 Output & Results

`Xvert()` returns the vertices of the constrained polytope along with optionally requested edge midpoints and centroids. `MixModel()` fits the Scheffé linear, quadratic, or special-cubic mixture model on the design, with coefficients interpretable as blend effects. Visualising the polytope and overlaying the design points before running the experiment is a recommended sanity check.

16.8.7 Interpretation

A reporting sentence: "The extreme-vertices design identified seven feasible vertices of the constrained simplex with bounds $L = (0.2, 0.1, 0.3)$ and $U = (0.5, 0.6, 0.7)$. The fitted quadratic Scheffé model showed the response was maximised near the vertex $(0.5, 0.2, 0.3)$ with a meaningful pairwise synergy between components 1 and 2 ($\beta_{12} = 5.9$, $p = 0.01$). Linear blend coefficients indicated component 2 was individually most effective." Always describe the feasible region and report which vertex achieved the optimum.

16.8.8 Practical Tips

- Use extreme-vertices designs whenever component bounds are genuinely binding; interior-only designs (simplex-lattice or centroid) miss optima that lie on the boundary of the feasible region — a common occurrence when constraints reflect formulation requirements.
- D-optimal mixture designs (via `AlgDesign::optMonteCarlo` or `AlgDesign::optFederov`) handle complex constraints elegantly and give the most efficient design when the number of feasible candidate points exceeds the run budget.
- Pseudo-component analysis makes the fit and interpretation more intuitive when only lower bounds are active: fit on the transformed unit simplex, then back-transform predictions to original component proportions for reporting.
- For ratio constraints (e.g., $\text{solvent}_1 : \text{solvent}_2 = 2 : 1$), use a ratio reparameterisation rather than naive component-level bounds; the constraint algebra is cleaner and the design more efficient.
- Always plot the feasible region in 3D or on the simplex before finalising the design; visualising the polytope helps detect over-tight constraints (a near-empty polytope) or accidentally redundant ones, and reveals which vertices are clinically meaningful.
- When the polytope is degenerate (very thin in one direction), the design will be ill-conditioned; relax the offending constraint or accept a higher prediction variance in the squeezed direction.

16.8.9 R Packages Used

`mixexp::Xvert()` for the canonical extreme-vertices design and `MixModel()` for Scheffé-model fitting; `AlgDesign::optFederov()` and `optMonteCarlo()` for D-optimal mixture designs under arbitrary linear constraints; `qualityTools::mixDesign()` for an alternative interface; `daewr` for textbook companion datasets and worked constrained-mixture examples; `MixExpR` for a tidyverse-friendly mixture-design workflow.

16.8.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.8.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.9 Crossover Designs

16.9.1 Introduction

Crossover designs apply two or more treatments to each subject in sequence, separated by washout periods, so that every subject acts as their own control. The within-subject comparison gains substantial efficiency over parallel-group designs because between-subject variability — typically the largest component of variance in clinical-pharmacology studies — is removed from the treatment contrast. The simplest and most widely used crossover is the two-period two-treatment AB/BA design, in which half of the subjects receive treatment A in period 1 and B in period 2, and the other half receive the reverse sequence.

Crossovers are standard in bioequivalence studies, in early-phase clinical pharmacology, and in many sensory and ergonomic experiments where conditions can be evaluated repeatedly on the same subject.

16.9.2 Prerequisites

A working understanding of repeated-measures analysis, mixed-effects models with subject as a random effect, and the concept of carryover (residual treatment effect from one period influencing the next).

16.9.3 Theory

In an AB/BA crossover, the standard mixed-effects analysis model is

$$y_{ijk} = \mu + \pi_j + \tau_k + s_i + \varepsilon_{ijk},$$

with period effect π_j, treatment effect τ_k, subject random effect $s_i \sim N(0, \sigma_s^2)$, and residual error ε_{ijk}. The treatment effect is estimated as the within-subject difference, eliminating the between-subject variance. **Carryover** — a residual treatment effect from period 1 affecting period 2 — manifests as a sequence-by-period interaction; if present, it biases the simple within-subject treatment estimate and requires either an adequate washout, a more complex Williams design, or a model that includes carryover terms.

16.9.4 Assumptions

Carryover is absent or has been controlled by an adequate washout period (pharmacologically, at least five half-lives of the active compound), the underlying condition is stable across periods (no progressive disease or natural recovery), and treatment effects are independent of order. Subjects are exchangeable between sequences.

16.9.5 R Implementation

```
library(nlme)

set.seed(2026)
n <- 20
sequence <- sample(c("AB", "BA"), n, replace = TRUE)
subj_re <- rnorm(n, 0, 1)

y_A <- subj_re + rnorm(n, 0, 0.5)
y_B <- subj_re + 0.6 + rnorm(n, 0, 0.5)

df <- data.frame(
```

```
  subject = rep(1:n, each = 2),
  period  = rep(1:2, n),
  sequence = rep(sequence, each = 2),
  treatment = unlist(lapply(sequence, function(s) strsplit(s, "")[[1]])),
  y = unlist(lapply(1:n, function(i)
    if (sequence[i] == "AB") c(y_A[i], y_B[i]) else c(y_B[i], y_A[i])))
)

fit <- lme(y ~ treatment + period, random = ~ 1 | subject, data = df)
summary(fit)$tTable
```

16.9.6 Output & Results

The mixed-effects fit returns the treatment effect with within-subject standard error and the period effect as a separate fixed term. The random-intercept structure absorbs between-subject variance, dramatically tightening the treatment estimate compared with a parallel-group design of the same total sample size.

16.9.7 Interpretation

A reporting sentence: "The two-period AB/BA crossover analysis estimated the B–A treatment difference as 0.56 (95 % CI 0.32 to 0.80, $p < 0.001$), achieving roughly three-fold precision relative to a parallel-group design with the same number of subjects. The period effect was not significant ($p = 0.41$), and the sequence × period interaction (a marker of carryover) was also non-significant ($p = 0.62$), supporting the no-carryover assumption. The mandated washout of seven half-lives was sufficient." Always report period and carryover diagnostics.

16.9.8 Practical Tips

- Adequate washout — at least five half-lives of the active compound, often more for compounds with long-lived metabolites — is the best defence against carryover; designing in conservative washouts is far cheaper than dealing with biased results post hoc.
- Avoid crossover designs for conditions that evolve naturally during the study (progressive disease, healing wounds, post-surgical recovery) or for interventions whose effects are short-lived and wash out rapidly without further dosing; both violate the stability assumption.
- The 3 × 3 Williams design balances first-order carryover effects across treatment pairs when carryover is plausible and washout cannot be made arbitrarily long; it is the standard alternative to AB/BA in pharmacology.
- Always analyse with a mixed-effects model that includes subject as a random effect; pooled two-sample t-tests ignore the within-subject structure and overstate the true precision dramatically.

- Report the period effect separately from the treatment effect; a strong period effect (e.g., a drift in measurement instrument) can bias the treatment estimate if the design is imbalanced across sequences.
- Pre-specify the carryover diagnostic and what will be done if carryover is detected; reactive analysis decisions after seeing the data inflate type-I error and are increasingly flagged by regulators.

16.9.9 R Packages Used

`nlme::lme()` for mixed-effects analysis with explicit random-effect specification; `lme4::lmer()` and `lmerTest` as alternatives with Satterthwaite or Kenward-Roger denominator df; `Crossover` for canonical crossover-design construction including Williams squares; `crossdes` for systematic generation of balanced crossover layouts; `daewr` for textbook companion datasets including pharmacology examples.

16.9.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.9.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.10 Desirability Functions

16.10.1 Introduction

Many experiments produce more than one response that must be optimised jointly: maximise yield while minimising impurity, hit a target dissolution time

while keeping particle size within spec, maximise tensile strength while minimising weight. Desirability functions, introduced by Harrington in 1965 and refined by Derringer and Suich in 1980, are the standard tool for combining multiple responses into a single overall optimisation criterion. Each response is converted to a $[0, 1]$ desirability scale via a transformation specific to that response's goal (maximise, minimise, or hit a target), and the individual desirabilities are combined into an overall desirability via the geometric mean. The compound score is then maximised over the factor space, often using the response surfaces fit during the second-order RSM phase.

16.10.2 Prerequisites

A working understanding of response-surface methodology, multi-objective optimisation concepts, and the difference between geometric and arithmetic aggregation of normalised scores.

16.10.3 Theory

For each response y_i, define a desirability function $d_i(y_i) \in [0, 1]$ — larger values are more desirable, irrespective of which direction is "good" on the original scale. The standard forms (Derringer–Suich) are:

- **Maximise**: $d_i = 0$ below a lower limit, $d_i = 1$ above a target value, linearly increasing between.
- **Minimise**: mirror of the maximise form.
- **Target**: $d_i = 0$ at lower and upper limits, $d_i = 1$ at the target, with linear or power transitions.

The overall desirability is $D = \left(\prod_i d_i\right)^{1/n}$, the geometric mean. Crucially, if any single $d_i = 0$ then $D = 0$ — the geometric mean is "non-compensatory", meaning no amount of excellence on one response can rescue an unacceptable value on another.

16.10.4 Assumptions

Each d_i is appropriately parameterised with realistic limits and target values, the geometric-mean aggregation is acceptable to the decision-maker (alternative weighted-power means exist for differential weighting), and the underlying response surfaces are reasonably well estimated.

16.10.5 R Implementation

```
library(desirability)

set.seed(2026)
df <- data.frame(x1 = runif(100, -1, 1), x2 = runif(100, -1, 1))
```

```
df$yield  <- 70 + 10 * df$x1 + 5 * df$x2 + rnorm(100)
df$cost   <- 20 - 2 * df$x1 + 4 * df$x2 + rnorm(100)

d_yield <- dMax(low = 60, high = 85)
d_cost  <- dMin(low = 10, high = 25)
dOverall <- dOverall(d_yield, d_cost)

df$d_overall <- predict(dOverall, with(df,
                                       cbind(yield = yield, cost = cost)))

head(df[order(-df$d_overall), ], 5)
```

16.10.6 Output & Results

The `desirability` package returns per-response and overall desirability for each observation or grid point. Combined with a fitted response-surface model, the workflow is to predict each response over a dense grid of factor settings, compute the overall desirability at each grid point, and identify the factor combination that maximises D. Reporting the achieved desirability for each individual response alongside the overall D communicates both the compromise and its quality.

16.10.7 Interpretation

A reporting sentence: "The best factor combination achieved yield 82 (90 % of the maximum desirability range) and cost 13 (85 % of the minimum desirability range), with overall $D = 0.87$. Pure yield optimisation would have driven yield to 88 but at cost 19 ($D = 0.42$); pure cost optimisation would have minimised cost to 11 but at yield 65 ($D = 0.31$). The desirability-balanced optimum is therefore meaningfully different from either single-objective optimum and is recommended for production." Always report the per-response desirabilities alongside D.

16.10.8 Practical Tips

- Use the geometric mean rather than the arithmetic mean to combine individual desirabilities; the geometric mean penalises low-desirability responses heavily and prevents one response from dominating the aggregate, while the arithmetic mean allows compensation that the decision-maker may not actually accept.
- Set realistic lower and upper limits for each desirability function; unrealistic limits flatten the function (everything inside is fully desirable, everything outside is zero) and the optimisation becomes uninformative.
- Differential weighting is achieved by exponentiating each desirability before averaging: $d_i^{s_i}$ with $s_i > 1$ increases the influence of response i, $s_i < 1$

decreases it. The Derringer–Suich formulation has this exponent as a built-in parameter.
- For fitted response surfaces, predict each response over a fine grid (or use a numerical optimiser) and maximise the overall desirability; the maximum is rarely at a corner of the design region and benefits from explicit search.
- Desirability functions are a heuristic for combining responses; for high-stakes decisions, complement them with formal multi-objective Pareto analysis to expose the trade-off curve and let the decision-maker choose explicitly.
- Document the desirability-function specifications (limits, targets, exponents) in the protocol; reviewers and downstream users cannot reproduce the optimisation without them, and the choices materially affect the recommended factor settings.

16.10.9 R Packages Used

`desirability` for canonical Derringer–Suich functions (`dMax`, `dMin`, `dTarget`) and combined-desirability prediction (`dOverall`); `qualityTools::desirability()` for an alternative interface integrated with RSM workflow; `mco` for formal multi-objective Pareto-front analysis as a complement to desirability heuristics; `rsm` for fitting the response surfaces that feed into the desirability optimisation.

16.10.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.10.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.11 2^k Factorial Designs

16.11.1 Introduction

The 2^k factorial design studies k factors, each at exactly two levels conventionally coded as -1 (low) and $+1$ (high), and runs every one of the 2^k possible combinations of factor levels. Because the design is balanced and orthogonal, every main effect and every interaction up to the k-way is estimable independently of every other, with all estimates having the same standard error. This combination of compactness, orthogonality, and full-information estimation makes the 2^k factorial the workhorse of industrial design of experiments (DoE), the natural starting point for product development and process improvement, and the foundation on which fractional factorials and response-surface designs are built.

16.11.2 Prerequisites

A working understanding of ANOVA with multiple factors, the concept of an interaction effect, and the use of coded factor levels (the orthogonal ± 1 scheme that simplifies computation and interpretation).

16.11.3 Theory

With coded levels ± 1, every factor effect is an orthogonal contrast: the main effect of factor A is the average response at $A = +1$ minus the average response at $A = -1$, and analogously for any interaction (a product of ± 1 columns averaged across runs). Orthogonality means each effect is estimated independently of every other effect, that the effects are uncorrelated, and that the analysis can proceed by simple linear regression with the coded factors and their products as predictors.

For an unreplicated 2^k run, residual degrees of freedom are zero and conventional ANOVA inference is impossible; Daniel's normal-probability plot of effects provides a graphical alternative that flags large effects against the inert ones, which form a Normal-distributed reference.

16.11.4 Assumptions

Two-level factor effects suffice to capture variation in the response (no curvature within the design region requiring a third level), runs are independent, and the residual error is approximately Normal with constant variance. Replicated designs satisfy these assumptions more transparently and allow formal F-tests; unreplicated designs rely on the small number of large effects against many inert effects ("effect sparsity") for inference.

16.11.5 R Implementation

```
library(FrF2)

fac <- FrF2(nruns = 8, nfactors = 3, replications = 2,
            randomize = TRUE, seed = 2026)
head(fac)

fac$A <- as.numeric(as.character(fac$A))
fac$B <- as.numeric(as.character(fac$B))
fac$C <- as.numeric(as.character(fac$C))

set.seed(2026)
fac$y <- 10 + 2 * fac$A + 1.5 * fac$B + 0.8 * fac$A * fac$B +
         rnorm(nrow(fac), 0, 0.7)

fit <- lm(y ~ A * B * C, data = fac)
summary(fit)
```

16.11.6 Output & Results

`FrF2()` generates a randomised 2^k layout with replication. The fitted regression returns coefficients for all main effects and all interactions; in coded units these coefficients are exactly half the conventional "factor effect" magnitudes. With replication, residual degrees of freedom support standard t-tests and an ANOVA partitioning into main, interaction, and residual sums of squares.

16.11.7 Interpretation

A reporting sentence: "The 2^3 factorial design with two replicates estimated significant main effects of factor A (2.1, $p < 0.001$) and factor B (1.4, $p < 0.001$) and a significant AB interaction (0.8, $p = 0.01$); factor C and all higher-order interactions were not significantly different from zero. Combined with an interaction plot, the AB synergy suggests using both factors at their high levels yields the optimal response." Always pair the regression output with an interaction plot when AB-type interactions are significant.

16.11.8 Practical Tips

- Always randomise the run order to protect against time-trend confounding; running treatments in factor-level order produces designs in which a temporal drift could be mistaken for a factor effect.
- With replication, residual degrees of freedom allow formal t-tests and F-tests; without replication, Daniel's normal-probability plot of effects (or Lenth's pseudo-standard-error procedure) is the standard inferential tool, exploiting effect sparsity.

- Screen large factor counts ($k \geq 5$) first with fractional factorials of resolution III or IV, then follow up with a full factorial on the small subset of active effects identified — the classic two-stage industrial DoE workflow.
- Code factors as ± 1 rather than the natural units; coded coefficients are directly comparable, orthogonal, and interpretable as half-effects, while natural-unit coefficients depend on factor scale and confound location with effect.
- Always report effect size alongside the p-value; a statistically significant tiny effect rarely justifies a process change, while a large but underpowered effect may motivate follow-up experimentation regardless of significance.
- Pair main-effect plots and interaction plots with the regression table; visual diagnostics are far more informative than coefficient lists alone for engineers and scientists who must act on the result.

16.11.9 R Packages Used

`FrF2` for the canonical 2^k design generation, including replication and randomisation, and Daniel's plot via `DanielPlot()`; `DoE.base` for general orthogonal-array construction including 2^k as a special case; base R `lm()` and `aov()` for analysis; `unrepx` for unreplicated-design effect identification (Lenth, half-Normal, Box-Meyer); `daewr` for textbook companion datasets and analysis helpers.

16.11.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.11.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.12 3^k Factorial Designs

16.12.1 Introduction

The 3^k factorial design uses three coded levels per factor — low (-1), middle (0), and high ($+1$) — and runs all 3^k combinations. The third level is what distinguishes the 3^k family from the simpler 2^k family: with three levels the design supports estimation of pure quadratic terms in each factor, and therefore the second-order polynomial approximation to a smooth response surface. The 3^k factorial is the simplest full-factorial construction that captures curvature and is the conceptual ancestor of the modern response-surface designs (central composite, Box-Behnken) that economise on the run count by replacing the dense 3^k grid with sparser alternatives carrying the same second-order information.

16.12.2 Prerequisites

Familiarity with 2^k factorial designs, the concept of an interaction effect, and quadratic regression for capturing curvature in a response.

16.12.3 Theory

For k three-level factors a full factorial has 3^k runs. With coded levels $\{-1, 0, +1\}$ the standard analysis model is the second-order polynomial

$$y = \beta_0 + \sum_i \beta_i x_i + \sum_i \beta_{ii} x_i^2 + \sum_{i<j} \beta_{ij} x_i x_j + \varepsilon,$$

with linear, pure-quadratic, and two-factor-interaction terms. The design grows quickly: $3^3 = 27$, $3^4 = 81$, $3^5 = 243$. For screening purposes the 3^k becomes impractical at $k = 4$ or beyond, and fractional 3^k designs or central composite / Box-Behnken layouts are far more economical for the same second-order information.

16.12.4 Assumptions

The response is well approximated by a second-order polynomial within the design region (no strong cubic or higher curvature), residuals are approximately Normal with constant variance, and the three levels span a range over which the polynomial approximation is plausible.

16.12.5 R Implementation

```
library(rsm)

set.seed(2026)
df <- expand.grid(A = c(-1, 0, 1), B = c(-1, 0, 1))
```

```
df$y <- 10 + 2 * df$A + 1.5 * df$B - df$A^2 +
        0.8 * df$A * df$B + rnorm(nrow(df), 0, 0.5)

fit <- rsm(y ~ SO(A, B), data = df)
summary(fit)
```

16.12.6 Output & Results

`rsm()` with the second-order specification returns coefficients for linear, pure-quadratic, and two-factor-interaction terms together with a stationary-point analysis (eigenvalue classification of the quadratic-form matrix as maximum, minimum, or saddle) and a lack-of-fit test against centre replicates when present. Reporting all components of the canonical analysis is the standard summary.

16.12.7 Interpretation

A reporting sentence: "The 3^2 full factorial estimated linear effects of A (2.0, $p < 0.001$) and B (1.5, $p = 0.002$), a significant quadratic term in A (-0.9, $p = 0.004$), and a mild interaction (0.7, $p = 0.07$). The stationary point of the fitted surface lay inside the design region and was a maximum, consistent with the expected concave behaviour. Predicted optimum was 12.4 at $(A, B) = (0.9, 0.8)$." Always pair the coefficient table with the canonical-analysis classification.

16.12.8 Practical Tips

- For $k \geq 3$, prefer central composite or Box-Behnken designs over full 3^k factorials; both deliver the same second-order information with far fewer runs and are the modern industrial-DoE default for response-surface modelling.
- Replicate the centre point ($x = 0$ for all factors) several times in any three-level design; the centre replicates supply pure-error degrees of freedom and a lack-of-fit test that diagnoses inadequacy of the second-order model.
- Coded levels $\{-1, 0, +1\}$ keep the design balanced and orthogonal, ensuring that linear, quadratic, and interaction effects are estimated independently and on directly comparable scales.
- Quadratic effects matter only when the response shows genuine curvature within the design region; if a 2^k factorial with replicated centre points fails to detect curvature, a 3^k or RSM follow-up is unnecessary.
- A 3^k full factorial is often a precursor to formal response-surface methodology; once the second-order surface is mapped, sequential experimentation along the path of steepest ascent or descent locates the optimum.
- For mixed-level designs (some factors at three levels, others at two) use orthogonal-array layouts via `DoE.base`; full crossed factorials become quickly unwieldy in mixed-level cases.

16.12.9 R Packages Used

`rsm` for second-order polynomial fitting with stationary-point and canonical analysis; `FrF2.mix` for fractional three-level and mixed-level designs; `DoE.base` for general orthogonal-array construction including 3^k as a special case; `agricolae` for textbook three-level layouts; `Vdgraph` for variance-dispersion comparisons across 3^k, CCD, and BBD alternatives.

16.12.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.12.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.13 Fractional Factorial Designs

16.13.1 Introduction

Fractional factorial designs run only a carefully chosen subset of a full 2^k factorial, accepting controlled confounding (also called aliasing) among effects in exchange for dramatic reductions in run count. They are the standard screening tool in industrial DoE: when many candidate factors must be evaluated and only a few are expected to matter, a fractional factorial sifts the active factors from the inert ones at a fraction of the cost of a full factorial. The trade-off is that some pairs (or larger groups) of effects share their estimating contrast and cannot be disentangled from the data alone, so design selection requires careful attention to which aliasing patterns are scientifically tolerable.

16.13.2 Prerequisites

A working understanding of full 2^k factorial designs, the concept of confounding among effect contrasts, and the Box–Hunter–Hunter sparsity-of-effects principle.

16.13.3 Theory

A 2^{k-p} design uses 2^{k-p} runs — a $1/2^p$-fraction of the corresponding full 2^k. The choice of p generators (defining relations among the factor columns) determines which effects are aliased with which. The design's **resolution** is the length of the shortest word in the defining relation:

- **Resolution III**: main effects are aliased with two-factor interactions; suitable for early screening only, where interactions are assumed negligible.
- **Resolution IV**: main effects are clear of two-factor interactions, but two-factor interactions alias with each other.
- **Resolution V**: main effects and two-factor interactions are all clear of each other; only three-factor and higher interactions are aliased.

Higher resolution generally requires more runs; the design choice is a balance between run economy and inferential cleanliness.

16.13.4 Assumptions

The sparsity-of-effects principle holds: lower-order effects (main, two-factor) dominate higher-order interactions, so aliasing among higher-order effects is acceptable. Each run is independent, residual error is approximately Normal with constant variance, and the chosen resolution is appropriate to the scientific question.

16.13.5 R Implementation

```
library(FrF2)

frac <- FrF2(nruns = 16, nfactors = 5, resolution = 5, randomize = TRUE)
design.info(frac)$aliased

set.seed(2026)
for (v in LETTERS[1:5]) frac[[v]] <- as.numeric(as.character(frac[[v]]))
frac$y <- 10 + 2 * frac$A + 1.5 * frac$B - 0.8 * frac$C +
          0.6 * frac$A * frac$B + rnorm(16, 0, 0.5)

fit <- lm(y ~ (A + B + C + D + E)^2, data = frac)
summary(fit)
```

16.13.6 Output & Results

`FrF2()` constructs the fractional factorial layout with the requested resolution; `design.info()` documents the alias structure explicitly so users see which effects are confounded with which. The regression fit returns coefficients for all main effects and two-factor interactions (the latter representing the joint estimate of any aliased pair), supporting standard t-tests on each.

16.13.7 Interpretation

A reporting sentence: "The 2^{5-1} resolution-V design used 16 runs — half of the full 32-run factorial — with main effects clear of all two-factor interactions and only the highest-order alias chain $ABCDE = I$. Significant effects were A (2.0, $p < 0.001$), B (1.5, $p = 0.001$), C (-0.8, $p = 0.02$), and the AB interaction (0.6, $p = 0.04$); factors D and E were inactive. The active subset $\{A, B, C\}$ will be followed up in a full factorial with replicated centre points to characterise curvature." Always document the alias structure.

16.13.8 Practical Tips

- Use resolution IV or V for screening whenever enough runs are available; resolution III is acceptable only for very early screening when interactions are confidently negligible and clean main-effect estimates are not essential.
- Resolution III Plackett–Burman designs are more economical (any number of factors that is a multiple of 4) but confound main effects with two-factor interactions; follow them up with a fold-over fraction or a higher-resolution design before drawing causal conclusions.
- Fold-over designs convert a resolution III into a resolution IV by adding a mirror-image fraction with all factor signs reversed; this de-aliases main effects from two-factor interactions at the cost of doubling the run count.
- After the screening step, follow up with a full factorial or response-surface design on the small subset of active factors identified; the classical industrial DoE workflow is "screen → characterise → optimise" in three sequential stages.
- Include centre points in fractional factorials whenever feasible; they enable a curvature test against the average of the factorial corners and provide pure-error degrees of freedom for lack-of-fit testing.
- Always document the alias chain in the methods section; reviewers will check it, and an undocumented aliasing pattern is a recurring source of misinterpreted DoE results.

16.13.9 R Packages Used

`FrF2` for canonical fractional-factorial construction with resolution control, alias-structure documentation, and Daniel-plot diagnostics; `DoE.base` for general orthogonal-array construction including fractional factorials; `BsMD` for Bayesian

screening of fractional designs; `unrepx` for unreplicated-design effect identification (Lenth, half-Normal, Box-Meyer); `daewr` for textbook companion datasets.

16.13.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.13.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.14 Graeco-Latin Square Designs

16.14.1 Introduction

A Graeco-Latin square superimposes two orthogonal Latin squares to block simultaneously on three nuisance factors in a single $t \times t$ layout. Each of the t treatments appears exactly once in every row, every column, and every Greek-letter level, so the four classifications — rows, columns, Greek letters, and Latin treatment letters — are mutually orthogonal. The design is extraordinarily compact: t^2 runs accommodate t treatments with three layers of blocking, where a comparable randomised-block design would need t^3 runs to achieve the same blocking. This makes Graeco-Latin squares classical tools in industrial and agricultural experiments where each run is expensive and three nuisance factors (operator, machine, day; or row, column, fertiliser-batch) need to be controlled simultaneously.

16.14.2 Prerequisites

Familiarity with Latin square designs, the orthogonality of factor classifications, and the concept of degrees-of-freedom partitioning in block ANOVA models.

16.14.3 Theory

The standard model is

$$y_{ijkl} = \mu + \rho_i + \gamma_j + \alpha_k + \tau_l + \varepsilon_{ijkl},$$

with row effect ρ_i, column effect γ_j, Greek-letter effect α_k, treatment effect τ_l, and independent Normal error. Orthogonal Latin squares of order t — required to construct the design — exist for all t except $t = 2$ and $t = 6$, the famous exceptions from the work of Tarry and the Bose–Shrikhande–Parker theory of mutually orthogonal Latin squares (Fisher's argument originally excluded $t = 6$).

16.14.4 Assumptions

No interactions exist between any pair of the four classifications (treatment, row, column, Greek), the residual error has homogeneous variance, and the response is approximately Normal. Because residual degrees of freedom are very small for the smallest squares, the design is sensitive to violations of these assumptions.

16.14.5 R Implementation

```
library(agricolae)

set.seed(2026)
trts <- LETTERS[1:4]
greek <- letters[1:4]
gls <- design.graeco(trts, greek, seed = 2026)
print(gls$book)

gls$book$y <- rnorm(16, mean = c(10, 12, 14,
 ↪  11)[as.integer(factor(gls$book$trts))], sd = 1.3)
fit <- aov(y ~ row + col + greek + trts, data = gls$book)
summary(fit)
```

16.14.6 Output & Results

`design.graeco()` returns the experimental layout with row, column, Greek, and treatment indicators for each of the t^2 runs. The ANOVA partitions the total sum of squares into row, column, Greek, treatment, and residual components,

with the treatment F-test as the primary inference. Reporting all four block effects alongside the treatment effect gives a complete picture of how variability was partitioned.

16.14.7 Interpretation

A reporting sentence: "The Graeco-Latin square blocked simultaneously on row, column, and Greek-letter classifications. The treatment main effect was marginal ($F_{3,3} = 6.4$, $p = 0.08$); residual degrees of freedom were limited ($\mathrm{df}_{\mathrm{res}} = 3$) so the test had low power, and a replicated design or a Latin square with one fewer blocking layer would have given more stable inference. Block sums of squares were modest, indicating the nuisance factors did not dominate the response variability." Always note residual df explicitly.

16.14.8 Practical Tips

- Residual degrees of freedom in a $t \times t$ Graeco-Latin square are $(t-1)(t-3)$, which is dangerously small for $t \leq 5$; for $t = 4$ only 3 residual df remain, severely limiting power.
- Replicate the entire square if residual df are too few; replicated Graeco-Latin squares restore power without losing the orthogonal blocking structure.
- The design is not available for $t = 2$ (trivially) or $t = 6$ (the Bose–Shrikhande–Parker non-existence result); for $t = 6$ a Latin square or a different blocking scheme must be used.
- The analysis assumes additivity of all four classifications; substantial interactions among rows, columns, Greek levels, and treatment invalidate the F-test, and a Tukey one-degree-of-freedom test for non-additivity is a useful diagnostic.
- Graeco-Latin squares are most common in agricultural field trials and industrial process-screening; in clinical-biostatistics they are rare because the assumed lack of interactions is hard to justify.
- Higher-order extensions (hyper-Graeco-Latin squares with four or more orthogonal Latin squares superimposed) exist but consume residual df even more aggressively and are seldom used in practice.

16.14.9 R Packages Used

`agricolae::design.graeco()` for the canonical layout construction and randomisation; `crossdes` for systematic generation of mutually orthogonal Latin squares of higher order; `DoE.base` for general orthogonal-array construction encompassing Graeco-Latin layouts; `lme4` and `lmerTest` for mixed-model analysis when block effects are treated as random.

16.14.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.14.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.15 Latin Square Designs

16.15.1 Introduction

A Latin square design simultaneously blocks on two nuisance factors — the rows and columns of a $t \times t$ grid — with t treatments arranged so that every treatment appears exactly once in each row and exactly once in each column. The result is a remarkably economical design: t treatments are replicated t times in only t^2 experimental units while controlling for two layers of nuisance variation. Latin squares are classical workhorses of agricultural experimentation (where rows and columns might be soil-fertility gradients in two perpendicular directions), industrial DoE (where day and operator are blocked), and clinical-pharmacology crossover trials (where subject and period block the treatment sequence).

16.15.2 Prerequisites

A working understanding of completely randomised designs, the principle of blocking, and the partitioning of variability in two-way ANOVA models without replication.

16.15.3 Theory

The standard analysis model is

$$y_{ijk} = \mu + \rho_i + \gamma_j + \tau_k + \varepsilon_{ijk}, \qquad \varepsilon_{ijk} \sim N(0, \sigma^2),$$

with row effect ρ_i, column effect γ_j, treatment effect τ_k, and independent Normal error. The constraint that each treatment appears once per row and once per column defines a Latin square. The design has $(t-1)$ treatment df, $(t-1)$ row df, $(t-1)$ column df, and $(t-1)(t-2)$ residual df, requiring t^2 runs in total.

16.15.4 Assumptions

No interactions exist between any pair of the three classifications (treatment, row, column), residual variance is homogeneous, and the response is approximately Normal. The design uses one observation per cell, so departures from these assumptions cannot easily be diagnosed from replication; residual diagnostics and a Tukey one-degree-of-freedom test for non-additivity are essential checks.

16.15.5 R Implementation

```
library(agricolae)

set.seed(2026)
t <- 4
trts <- LETTERS[1:t]
ls_design <- design.lsd(trts, seed = 2026)
book <- ls_design$book
print(book)

book$y <- rnorm(t^2, mean = c(10, 12, 14, 11)[as.integer(book$trts)], sd
 ↪  = 1.5)
fit <- aov(y ~ row + col + trts, data = book)
summary(fit)
```

16.15.6 Output & Results

`design.lsd()` returns a randomised Latin-square layout with row, column, and treatment indicators. The ANOVA partitions the total sum of squares into row, column, treatment, and residual components, with the treatment F-test as the primary inference. Reporting the proportions of variability absorbed by row and column blocking alongside the treatment effect characterises the gain from the design.

16.15.7 Interpretation

A reporting sentence: "The 4×4 Latin square partitioned the 16 observations' variation into row, column, and treatment components. The treatment F-test was significant ($F_{3,6} = 7.2$, $p = 0.02$), with row blocking absorbing 20 % of total variability and column blocking 14 %; residual variance was correspondingly small. The design assumes no row-treatment or column-treatment interaction, supported by visual inspection of the residuals." Always report blocking variance components and verify the no-interaction assumption.

16.15.8 Practical Tips

- Latin squares have very few residual df for small t (only 6 for $t = 4$); consider replication or a larger square if effects of interest are modest in size.
- The analysis assumes no interactions between treatment and either blocking factor; substantive interactions invalidate the standard F-test, and the no-interaction assumption should always be defended from context (or tested via a Tukey one-df non-additivity check).
- For $t > 7$ a Latin square requires too many units ($t^2 \geq 49$); a balanced incomplete block design or a fractional-factorial blocked layout is usually more practical.
- Graeco-Latin squares extend to a third blocking factor by superimposing two orthogonal Latin squares; they are even more economical but with even fewer residual df and the same additive-effects assumption.
- Always randomly permute rows, columns, and treatment labels of a standard Latin square to obtain a valid realisation; using an unrandomised square confounds systematic patterns with treatment effects.
- In crossover clinical trials, "subject" plays the role of one block and "period" the role of the other; carryover effects, if present, violate the no-interaction assumption and must be modelled separately.

16.15.9 R Packages Used

`agricolae::design.lsd()` for the canonical layout and randomisation; base R `aov()` for the ANOVA analysis; `crossdes` for systematic Latin-square construction including replicated and balanced layouts; `daewr` for textbook examples; `emmeans` for marginal means and treatment contrasts after fitting.

16.15.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.15.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.16 Simplex Centroid Designs

16.16.1 Introduction

Simplex centroid designs are mixture experiments that include the centroids of every sub-simplex of the component simplex: each pure component (vertex), each binary midpoint (edge centroid), each ternary centroid (face centroid), and on up to the overall centroid of the full simplex. The result is a set of $2^q - 1$ design points for q components that spans all blend orders simultaneously, making the design well suited to detecting higher-order synergies — three-way and four-way blend interactions that simpler simplex-lattice designs cannot estimate. They are widely used in food science, pharmaceutical formulation, and chemical product development whenever the response depends on combinations of multiple ingredients and pure components alone do not capture the behaviour.

16.16.2 Prerequisites

Familiarity with mixture experiments, the simplex-lattice design, and the Scheffé canonical polynomial form for mixture models.

16.16.3 Theory

For q components a simplex centroid design has $2^q - 1$ points: $\binom{q}{1} = q$ vertices, $\binom{q}{2}$ edge midpoints, $\binom{q}{3}$ face centroids, and so on, up to the overall centroid. The Scheffé canonical polynomial including all blend orders is

$$y = \sum_i \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j + \sum_{i<j<k} \beta_{ijk} x_i x_j x_k + \cdots,$$

with no intercept (because $\sum_i x_i = 1$). The simplex centroid design supplies

exactly enough information to estimate every blend term up to the q-way interaction, in contrast to simplex-lattice designs that estimate only up to a chosen order $m \leq q$.

16.16.4 Assumptions

Components are non-negative and sum to one (so the design lies on the standard simplex), higher-order synergies are potentially meaningful (otherwise a lower-order simplex-lattice would suffice), and residual error is approximately Normal with constant variance across the simplex.

16.16.5 R Implementation

```
library(mixexp)

design <- SCD(3)
design

set.seed(2026)
design$y <- 10 * design$x1 + 12 * design$x2 + 8 * design$x3 +
            6 * design$x1 * design$x2 +
            4 * design$x1 * design$x3 +
            3 * design$x1 * design$x2 * design$x3 +
            rnorm(nrow(design), 0, 0.3)

fit <- MixModel(design, "y",
                mixcomps = c("x1", "x2", "x3"), model = 3)
summary(fit)
```

16.16.6 Output & Results

`SCD()` returns the seven design points for a three-component simplex centroid (three vertices, three edge midpoints, the centroid). Fitting the cubic Scheffé model yields coefficients for the linear blend, the three pairwise blend interactions, and the ternary blend. Reporting the ternary coefficient explicitly is the canonical way to demonstrate that the centroid design has captured higher-order synergy beyond pairwise interaction.

16.16.7 Interpretation

A reporting sentence: "The seven-run simplex centroid design with cubic Scheffé fit detected a meaningful ternary synergy ($\beta_{123} = 3.1$, $p = 0.02$) beyond the pairwise blend interactions ($\beta_{12} = 6.0$, $\beta_{13} = 3.9$). The ternary centroid (1/3, 1/3, 1/3) achieved a fitted response of 11.3, exceeding any binary edge centroid; the optimal blend therefore mixes all three components rather than just pairs."

Always report the higher-order coefficients explicitly when justifying a centroid design.

16.16.8 Practical Tips

- Simplex centroid designs require more runs than simplex-lattice designs of the same maximum interaction order; use the centroid form only when higher-order blends are scientifically plausible and worth estimating.
- Plot the fitted response surface on the ternary (three-component) or quaternary simplex; mixture-design intuition is overwhelmingly visual, and contour plots on the simplex reveal where the optimum lies far more clearly than coefficient lists.
- Augment the simplex centroid with axial points (positions partway between the centroid and each vertex) to improve coverage of non-centroid mixtures and stabilise prediction in the interior of the simplex.
- For bounded mixtures with explicit constraints on minimum or maximum component proportions, extreme-vertices designs are the appropriate alternative; the standard centroid form assumes the full unconstrained simplex.
- Mixture-amount designs cross the simplex with a total-amount factor and are appropriate when both formulation and total dose matter; they are common in pharmaceutical formulation and feed-mixture studies.
- The `mixexp` and `MixExpR` packages also support mixture-process designs that combine mixture components with non-mixture process variables (e.g., temperature, pressure); these are increasingly common in chemical-engineering DoE.

16.16.9 R Packages Used

`mixexp::SCD()` and `mixexp::MixModel()` for the canonical simplex centroid design and Scheffé-model fitting; `MixExpR` for an alternative tidyverse-friendly mixture-design ecosystem; `qualityTools::mixDesign()` for related mixture-design construction; `daewr` for textbook companion datasets and worked examples.

16.16.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.16.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials

- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.17 Simplex Lattice Mixture Designs

16.17.1 Introduction

Mixture experiments study how a response depends on the relative proportions of two or more components in a blend, where the proportions are constrained to sum to one. They arise routinely in food formulation (recipes), paint and coating chemistry, pharmaceutical excipient blending, fuel formulation, and any setting in which the absolute amount is fixed and only the relative composition can vary. Because the simplex constraint $x_1 + \cdots + x_q = 1$ removes one degree of freedom, standard polynomial regression models are not directly applicable, and Scheffé's canonical polynomial forms — which omit the intercept and pure quadratic terms — are the appropriate functional family. Simplex-lattice designs are the foundational mixture-design construction and the natural starting point for any mixture experiment.

16.17.2 Prerequisites

A working understanding of factorial designs, basic simplex geometry, and the Scheffé canonical polynomial form for mixture models.

16.17.3 Theory

A $\{q, m\}$ simplex-lattice design for q components samples all blends in which each component takes a proportion in the set $\{0, 1/m, 2/m, \dots, 1\}$, restricted to those proportions that sum to one. For $q = 3$ and $m = 2$, the design has six points: three pure components (vertices of the simplex), and three binary midpoints (edge centres at $(1/2, 1/2, 0)$, $(1/2, 0, 1/2)$, $(0, 1/2, 1/2)$). The Scheffé canonical polynomial of order m in canonical form is

$$y = \sum_i \beta_i x_i + \sum_{i<j} \beta_{ij} x_i x_j + \cdots,$$

with no intercept (because the linear terms already absorb a constant) and no

pure-quadratic terms (because $\sum x_i = 1$ implies $x_i^2 = x_i - \sum_{j \neq i} x_i x_j$, redistributing the squared terms into linear and interaction blends).

16.17.4 Assumptions

The components sum to one (so the design lies on the standard $(q-1)$-simplex), the response is a smooth function of the component proportions, the residual error is approximately Normal with constant variance, and the chosen Scheffé polynomial order m adequately captures the surface curvature.

16.17.5 R Implementation

```
library(mixexp)

design <- SLD(3, 2)
design

set.seed(2026)
design$y <- 10 * design$x1 + 12 * design$x2 + 8 * design$x3 +
            6 * design$x1 * design$x2 +
            rnorm(nrow(design), 0, 0.3)

fit <- MixModel(design, "y", mixcomps = c("x1", "x2", "x3"), model = 2)
summary(fit)
```

16.17.6 Output & Results

`SLD()` returns the simplex-lattice points, and `MixModel()` fits the Scheffé polynomial of the requested order. The output reports per-component linear blend coefficients and pairwise interaction coefficients; reading these together with a contour plot on the ternary simplex is the standard way to identify the optimal blend.

16.17.7 Interpretation

A reporting sentence: "The $\{3, 2\}$ simplex-lattice mixture design with six runs identified component 2 as the strongest pure component ($\beta_2 = 12.1$) and a positive synergy between components 1 and 2 ($\beta_{12} = 5.9$). The fitted optimum lies on the x_1-x_2 edge near $(0.5, 0.5, 0)$ with predicted response 12.5, exceeding any pure-component value. A follow-up centroid-augmented design will confirm whether the optimum truly lies on the binary edge or shifts toward the interior." Always interpret pure components alongside blend interactions.

16.17.8 Practical Tips

- Report coefficients only in the Scheffé canonical form; standard polynomial models with intercept and pure quadratic terms misrepresent mixture data because they ignore the simplex constraint.
- Augment simplex-lattice designs with the overall centroid (and optionally axial points partway between centroid and vertices) for better prediction of non-extreme blends; pure simplex-lattice designs over-sample the boundary and under-sample the interior.
- For experiments with binding lower or upper bounds on components, use constrained mixture designs with extreme-vertices construction or pseudo-component reparameterisation; unconstrained simplex designs would place runs in the infeasible region.
- Mixture-process variable designs cross the simplex with one or more factorial factors (e.g., mixing temperature, time, pressure); they are common in chemical-engineering DoE and require a combined Scheffé + standard-polynomial model.
- Contour plots on the ternary simplex (or generalisations to higher dimensions) are the standard visualisation; mixture intuition is overwhelmingly visual, and contour plots reveal optimal blends and constraint-binding edges far better than coefficient tables.
- For more than four or five components, full simplex-lattice designs become impractical; D-optimal mixture designs or fractional simplex constructions are the modern alternative.

16.17.9 R Packages Used

`mixexp::SLD()` and `mixexp::MixModel()` for the canonical simplex-lattice design and Scheffé-model fitting; `mixexp::SCD()` for the simplex-centroid extension; `MixExpR` for a tidyverse-friendly mixture-design ecosystem; `qualityTools::mixDesign()` for an alternative mixture-design toolkit; `AlgDesign` for D-optimal mixture designs under arbitrary constraints.

16.17.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.17.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials

- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.18 Optimal Designs: D, A, I

16.18.1 Introduction

Computer-generated optimal designs minimise a chosen criterion on the information matrix. When a standard design does not fit constraints (irregular region, mixed-level factors, restricted runs), optimal designs are the primary recourse.

16.18.2 Prerequisites

Linear regression; Fisher information matrix.

16.18.3 Theory

For a design matrix X under the assumed model, the information matrix is $M = X^T X$. Common criteria: - **D-optimal**: maximise $\det(M)$ – minimises the volume of the joint confidence ellipsoid for coefficients. - **A-optimal**: minimise $\text{trace}(M^{-1})$ – minimises the average variance of coefficient estimates. - **I-optimal**: minimise average prediction variance over the design region. - **G-optimal**: minimise the maximum prediction variance.

Computer algorithms (Fedorov exchange) search for the optimum.

16.18.4 Assumptions

Model is correctly specified; $n \geq p$ (parameters); design region is explicit.

16.18.5 R Implementation

```
library(AlgDesign)

set.seed(2026)
# Candidate set: 3^2 factorial grid
candidates <- expand.grid(x1 = c(-1, 0, 1), x2 = c(-1, 0, 1))
```

```
# D-optimal design for a second-order model, selecting 8 runs
d_opt <- optFederov(~ quad(x1, x2),
                    data = candidates,
                    nTrials = 8,
                    criterion = "D",
                    nRepeats = 20)
d_opt$design

# I-optimal
i_opt <- optFederov(~ quad(x1, x2),
                    data = candidates,
                    nTrials = 8,
                    criterion = "I",
                    nRepeats = 20)
i_opt$design
```

16.18.6 Output & Results

Two sets of 8 runs from the 9-point candidate set; D-optimal maximises determinant of $X^T X$; I-optimal minimises average prediction variance.

16.18.7 Interpretation

"D-optimal selected the 8 runs that most tightly constrain the polynomial coefficients; I-optimal picked a different subset that minimises prediction variance across the design region. Choose D for inference on coefficients, I for prediction."

16.18.8 Practical Tips

- Use D-optimal for parameter-precision goals; I-optimal for prediction-focused studies.
- Generate several random starts (`nRepeats`); optimal-design search is non-convex.
- Verify the final design's efficiency against standard designs if one is applicable.
- Use optimal designs when standard textbook designs don't fit constraints.
- Mixed-level optimal designs can handle categorical factors; pass them as factors to `optFederov`.

16.18.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a

manuscript's application of the method.

16.18.10 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.19 Orthogonal Arrays

16.19.1 Introduction

Orthogonal arrays (OAs) are tabulated fractional-factorial designs in which every pair of columns contains every combination of levels an equal number of times. Genichi Taguchi popularised a catalogue of specific OAs — L_4, L_8, L_9, L_{16}, L_{18}, L_{27}, and others — as economical screening designs in his quality-engineering framework, and the labels are now standard shorthand in industrial DoE. The orthogonality property guarantees that main effects of distinct factors are estimated independently and on equal footing, so OAs deliver the cleanest possible main-effect estimates from the smallest possible run count, given the assumed effect-sparsity context.

16.19.2 Prerequisites

Familiarity with fractional factorial designs, the Plackett–Burman family, and the concept of orthogonality among columns of a design matrix.

16.19.3 Theory

An $L_n(m^k)$ array has n rows (runs) and k columns (factors), each column at m equally-replicated levels. Orthogonality means that for any pair of columns each combination of levels appears an equal number of times, ensuring uncorrelated main-effect estimates.

The standard catalogue includes:

- **L4**: 4 runs, up to 3 two-level factors.

- **L8**: 8 runs, up to 7 two-level factors.
- **L9**: 9 runs, up to 4 three-level factors.
- **L16**: 16 runs, up to 15 two-level factors.
- **L18**: 18 runs, one two-level factor plus 7 three-level factors (mixed-level).
- **L27**: 27 runs, up to 13 three-level factors.

16.19.4 Assumptions

Main effects dominate the response (effect-sparsity principle); two-factor and higher-order interactions are either negligible or have been assigned to specific pre-determined columns by reading the array's linear-graph; runs are independent with constant residual variance.

16.19.5 R Implementation

```
library(qualityTools)

l9 <- taguchiDesign("L9_3^4")
summary(l9)

set.seed(2026)
l9$y <- rnorm(9, mean = 10 + 2 * as.numeric(l9$A) -
                      1.5 * as.numeric(l9$C), sd = 0.5)

effectPlot(l9, response = "y")
```

16.19.6 Output & Results

`taguchiDesign()` returns the requested standard array as a `qualityTools` design object; `effectPlot()` renders the main-effects plot showing the average response at each level of each factor. Reading the plot — the factor whose level-mean line shows the steepest slope is most active — is the canonical first interpretation step in a Taguchi-style screening analysis.

16.19.7 Interpretation

A reporting sentence: "The L_9 orthogonal array screened four three-level factors in nine runs. The main-effects plot identified factors A and C as active (level means ranging across $\sim$ 4 units of response), with optimal levels A_3 and C_1. Factors B and D showed nearly flat level-means and were inactive. The active subset will be carried forward into a full 3^2 factorial with replicated centre points to characterise curvature." Always justify "active" assignments by effect magnitude rather than p-values alone in OA screening.

16.19.8 Practical Tips

- Taguchi orthogonal arrays are a subset of the broader fractional-factorial framework; modern DoE practice often prefers `FrF2`, `DoE.base`, and `conf.design` constructions for greater flexibility, explicit alias documentation, and Lenth-style analysis tools.
- Interaction analysis from an OA requires consulting the array's linear-graph (a graphical representation of which column-pair maps to which interaction); without it, interactions are confounded silently with main effects.
- Mixed-level orthogonal arrays such as L_{18} and L_{36} accommodate combinations of two-level and three-level factors and are useful when one or two factors warrant a quadratic check while others are clearly two-level.
- Follow up an OA screening with a higher-resolution design on the small subset of active factors identified; the screen-then-characterise sequence is the canonical industrial-DoE workflow and applies to OAs as much as to standard fractional factorials.
- Custom-designed D-optimal designs from `AlgDesign::optFederov()` often beat off-the-shelf Taguchi arrays in efficiency, especially for unusual factor counts, mixed levels, or constrained design regions; consider them when the standard catalogue does not fit cleanly.
- The Taguchi philosophy distinguishes "control" factors from "noise" factors via inner and outer arrays; this is the basis of robust parameter design and is treated separately in dedicated robust-design tutorials.

16.19.9 R Packages Used

`qualityTools::taguchiDesign()` for standard Taguchi catalogue arrays and `effectPlot()` for main-effects visualisation; `DoE.base::oa.design()` for general orthogonal-array construction beyond the Taguchi catalogue; `FrF2` for fractional factorials including those equivalent to Taguchi arrays; `AlgDesign` for computer-generated D-optimal alternatives; `conf.design` for low-level construction of confounding patterns.

16.19.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.19.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials

- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.20 Plackett-Burman Designs

16.20.1 Introduction

Plackett–Burman (PB) designs, introduced by Plackett and Burman in 1946, are two-level orthogonal screening designs with n runs (where n is a multiple of 4) that can accommodate up to $n - 1$ factors. They are widely regarded as the most economical screening tool in industrial DoE: when many candidate factors must be sifted to identify the few that drive the response, a 12-run PB design can screen up to 11 factors, a 20-run PB up to 19, and so on. The price for this economy is a resolution-III aliasing structure in which main effects are confounded with two-factor interactions, so PB designs are appropriate strictly for screening — never for confirmatory or interaction-aware inference.

16.20.2 Prerequisites

Familiarity with fractional factorial designs, the concept of resolution, the effect-sparsity principle, and the distinction between regular and non-regular fractional factorial constructions.

16.20.3 Theory

PB designs are built from Hadamard matrices, square matrices of ± 1 entries with mutually orthogonal columns. With n runs the design supplies $n - 1$ orthogonal contrast columns, each of which can carry one factor. For $n = 12$, 11 factors fit; for $n = 20$, 19 factors.

Non-regular PB designs (those whose run count is a multiple of 4 but not a power of 2, like 12, 20, 24) have a famously complex partial-aliasing structure: each main effect is aliased with multiple two-factor interactions at fractional weights, rather than fully confounded with a single one. This complex partial aliasing is a property the experimentalist must understand to interpret active-effect candidates correctly. Regular PB designs (n a power of 2) coincide with standard resolution-III fractional factorials and have cleaner alias chains.

16.20.4 Assumptions

Two-factor and higher-order interactions are negligible compared with main effects (the sparsity-of-effects principle), and follow-up experiments will be conducted on the screened-in subset to characterise interactions properly. PB designs are a tool of two-stage industrial DoE: screen first, characterise second.

16.20.5 R Implementation

```r
library(FrF2)

pb <- pb(nruns = 12, nfactors = 8, randomize = TRUE, seed = 2026)
pb

for (v in paste0("A", 1:8)) {
  pb[[v]] <- as.numeric(as.character(pb[[v]]))
}
set.seed(2026)
pb$y <- 10 + 2.5 * pb$A1 + 1.8 * pb$A2 + rnorm(nrow(pb), 0, 0.6)

fit <- lm(y ~ ., data = pb)
summary(fit)
```

16.20.6 Output & Results

`pb()` constructs a randomised Plackett–Burman layout for the requested run count and factor count. The main-effects regression returns coefficients for each factor; combining the regression with a Lenth-style or half-Normal-plot identification of the active subset is the standard analytical approach for unreplicated PB designs.

16.20.7 Interpretation

A reporting sentence: "The 12-run Plackett–Burman design screened eight two-level factors, identifying A1 (effect 2.5, $p < 0.001$) and A2 (effect 1.8, $p < 0.001$) as the principal drivers of the response. The remaining six factors had effect estimates within Lenth's pseudo-standard-error envelope and were dropped from follow-up. A subsequent 2^2 full factorial with replicated centre points characterised curvature and the A1 $\times$ A2 interaction in the active subset." Always describe both active and inactive factors and the screening criterion used.

16.20.8 Practical Tips

- Plackett–Burman is a screening design; always follow it up with a higher-resolution fractional factorial, full factorial, or response-surface design on

the active factor subset. Drawing causal conclusions directly from a PB result without follow-up is methodologically unsound.
- The 12-run design is the most popular member of the family because it balances economy (one-third the runs of a comparable resolution-IV fractional factorial) with the orthogonality and ease-of-interpretation of regular Hadamard-based construction.
- Complex partial aliasing in non-regular PB designs can occasionally reveal active two-factor interactions indirectly through effect-heredity logic, but this should be regarded as exploratory; formal interaction inference still requires a higher-resolution follow-up design.
- Always randomise run order to protect against temporal confounding; running treatments in design order conflates time-trend effects with factor effects in any unreplicated screening design.
- Report transparently which factors were screened in and out of the follow-up; the pruning decision is itself a substantive analytical choice and reviewers will want to see the criterion used.
- For mixed-level screening, definitive screening designs (DSDs, Jones and Nachtsheim 2011) generalise PB ideas to three-level factors and offer cleaner analysis of curvature alongside main-effect screening.

16.20.9 R Packages Used

`FrF2::pb()` for Plackett–Burman construction with explicit randomisation; `DoE.base::oa.design()` for general Hadamard-matrix-based orthogonal arrays; `daewr` for textbook companion examples and analysis helpers; `unrepx` for Lenth-style identification of active effects in unreplicated PB designs; `dsd` and `daewr::DefScreen()` for definitive screening designs as a modern alternative.

16.20.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.20.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE

- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.21 Power Analysis for DOE

16.21.1 Introduction

Power analysis for designed experiments determines the run count required to detect specified main effects and interactions with adequate probability. It bridges experimental-design choice and statistical inference: a design is only useful if its run count gives sensible power for the smallest effect that would be scientifically actionable. Skimping on power means under-powered experiments that fail to detect real effects and waste resources on inconclusive results; over-powering an experiment beyond what is needed wastes resources differently and exposes the design to detecting irrelevantly small effects. A pre-specified power calculation is therefore now standard in any DoE protocol — industrial, agricultural, or clinical — and is increasingly required by regulators and journal reviewers.

16.21.2 Prerequisites

A working understanding of effect sizes (Cohen's d, f, f^2), the structure of ANOVA F-tests, the non-central F-distribution, and the concepts of significance level and statistical power.

16.21.3 Theory

For a 2^k factorial design, the main-effect F-test on a single factor has non-centrality parameter

$$\lambda = \frac{n\Delta^2}{4\sigma^2},$$

where n is the total number of runs, Δ is the effect (difference between high- and low-level means), and σ^2 is the residual variance. Power is computed as $P(F > F_{\alpha,1,\mathrm{df}} \mid \lambda)$ under the non-central F-distribution. Interactions typically have effect sizes 30–50 % of the corresponding main effects, so designs powered adequately for main effects routinely lack power for two-factor interactions and for higher-order terms.

16.21.4 Assumptions

The residual variance is known or pre-specified, the effect size of interest is fixed in advance, the design and analysis model are pre-specified, and the resid-

uals satisfy the standard ANOVA assumptions of independence, Normality, and constant variance.

16.21.5 R Implementation

```r
library(pwr)

pwr.t.test(d = 1, n = 8, sig.level = 0.05,
           type = "two.sample", alternative = "two.sided")

set.seed(2026)
n_sim <- 500
reps <- 2
power_sim <- mean(replicate(n_sim, {
  A <- rep(c(-1, 1), each = 4 * reps)
  B <- rep(rep(c(-1, 1), each = 2 * reps), 2)
  C <- rep(c(-1, 1), times = 4 * reps)
  y <- 1 * A + 0.5 * B + rnorm(8 * reps, 0, 1.5)
  pA <- summary(lm(y ~ A + B + C))$coefficients["A", "Pr(>|t|)"]
  pA < 0.05
}))
power_sim
```

16.21.6 Output & Results

The analytical power calculation for a two-sample t-test reports power for a single main effect under classical Cohen's d assumptions; the simulation loop reports empirical power under the actual analysis model and noise structure. The two should agree to within Monte Carlo error when assumptions match, and they often do not — the simulation approach generalises naturally to mixed-effects models, generalised linear models, and non-Normal residuals where no analytical formula exists.

16.21.7 Interpretation

A reporting sentence: "Pre-specified power analysis for the 2^3 factorial with two replicates ($n = 16$ total runs) achieved 82 % power to detect a 1.0 SD main effect at $\alpha = 0.05$, exceeding the 80 % minimum target. Power for a 0.5 SD two-factor interaction was only 28 %, so the experiment was not powered for interaction detection; if interactions are scientifically important, a third replicate ($n = 24$) raises interaction power to roughly 50 %, and a 2^4 design with one replicate would achieve 70 %." Always report power separately for main effects and interactions.

16.21.8 Practical Tips

- Always power the experiment for the smallest effect of scientific interest, not the largest expected effect; under-powering for small effects is the most common DoE error in published industrial work.
- Two-factor interactions typically have effect sizes 30–50 % of main effects; if interactions are part of the inferential question, scale up replication or run count accordingly. Designs powered for main effects routinely have very poor interaction power.
- Replicate the design (rather than expanding the factor space) to increase power for fixed factor structure; centre points add pure-error df without expanding the design region.
- Use simulation rather than analytical formulas when the analysis model deviates from classical ANOVA — mixed-effects models, generalised linear models, non-Normal residuals, and complex covariance structures are all best handled by Monte Carlo as above.
- Pre-specify the power calculation in the experimental protocol with explicit assumed values for σ, the smallest meaningful effect, and the analysis model; post-hoc power calculations after the experiment are statistically invalid and methodologically frowned upon.
- When prior data on σ are unavailable, conduct a pilot experiment first; treating an arbitrary or wishful σ as known invalidates the whole calculation.

16.21.9 R Packages Used

`pwr` for analytical power on t-tests, F-tests, ANOVA, and correlations; `Superpower` for ANOVA power simulation across complex factorial designs; `simr` for power analysis of mixed-effects models via simulation; `WebPower` for web-style power tools and complex designs; `daewr` for textbook DoE power examples.

16.21.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.21.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR

- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.22 Randomised Complete Block Design

16.22.1 Introduction

A randomised complete block design (RCBD) is one of the most useful experimental designs in applied research. When a known source of variation – day of measurement, animal cage, plate, subject – is likely to influence the outcome, RCBD groups experimental units into homogeneous blocks and randomises treatments *within* each block. The block effect is thereby separated from the treatment effect, and the residual error is the within-block variability, which is typically much smaller than the pooled unblocked error.

16.22.2 Prerequisites

The reader should have met the one-way ANOVA and the principle of randomisation. Familiarity with R's formula syntax is assumed.

16.22.3 Theory

Let y_{ij} be the response of the experimental unit receiving treatment i in block j, for $i = 1, \dots, t$ treatments and $j = 1, \dots, b$ blocks. The additive RCBD model is

$$y_{ij} = \mu + \tau_i + \beta_j + \varepsilon_{ij}, \qquad \varepsilon_{ij} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2),$$

where τ_i is the treatment effect (summing to zero) and β_j is the block effect. Each treatment appears exactly once in each block – that is the "complete" in RCBD.

The ANOVA decomposes the total sum of squares into treatment, block, and residual components. The F-test for treatments has $t-1$ numerator and $(t-1)(b-1)$ denominator degrees of freedom. If blocking has captured real variation, the residual mean square is smaller than it would have been in a completely randomised design, and the test has higher power.

The design assumes **no treatment-by-block interaction**. If an interaction exists (a treatment works better in some blocks than others), the additive model is

misspecified. The Tukey one-degree-of-freedom test for non-additivity provides a partial check.

16.22.4 Assumptions

1. **Additivity**: no treatment-by-block interaction.
2. **Independence**: residuals are mutually independent.
3. **Homoscedasticity**: residuals have constant variance across treatments and blocks.
4. **Normality**: residuals are approximately normal.

16.22.5 R Implementation

```
library(agricolae)
library(emmeans)

treatments <- LETTERS[1:4]
blocks <- paste0("B", 1:5)

set.seed(2026)
design <- design.rcbd(trt = treatments, r = length(blocks),
                      seed = 2026)
print(design$book)

design$book$y <- with(design$book,
                      5 +
                        as.numeric(factor(block)) * 0.8 +
                        c(0, 1.2, 1.8,
                          ↪  0.4)[as.numeric(factor(treatments))] +
                        rnorm(nrow(design$book), 0, 0.6))

fit <- aov(y ~ block + treatments, data = design$book)
summary(fit)

emm <- emmeans(fit, ~ treatments)
pairs(emm, adjust = "tukey")
```

`design.rcbd()` produces a randomised plan with each treatment appearing once per block. The simulated response includes a block effect and a treatment effect with the lowest for A, highest for C. `aov()` fits the additive RCBD model; `emmeans::pairs()` performs pairwise comparisons with Tukey adjustment.

16.22.6 Output & Results

The ANOVA table:

```
            Df Sum Sq Mean Sq F value  Pr(>F)
```

```
block        4 18.120   4.530  12.815 0.00015 ***
treatments   3  9.452   3.151   8.916 0.00142 **
Residuals   12  4.242   0.353
```

The block effect accounts for a substantial fraction of the variability – confirming that blocking was worthwhile – and the treatment effect is significant at $p = 0.0014$. Pairwise comparisons show C significantly higher than A ($p < 0.001$) and B ($p = 0.03$), and D not significantly different from A.

16.22.7 Interpretation

For a manuscript: "Treatments were compared in a randomised complete block design with four treatments and five blocks. An additive ANOVA showed a significant treatment effect, $F(3, 12) = 8.92$, $p = 0.001$, after accounting for block variation (which itself was substantial, $F(4, 12) = 12.82$, $p < 0.001$). Tukey-adjusted pairwise comparisons indicated that treatment C produced significantly higher responses than treatments A and B (see Table X)."

16.22.8 Practical Tips

- Block on variables that you expect to influence the outcome and that you cannot randomise out. Common choices: day of measurement, microtitre plate, cage, field plot, subject.
- Do not include the block as a "predictor of interest" – its role is to remove nuisance variation. Reporting its effect is informative but rarely the headline.
- If you suspect treatment-by-block interaction, switch to a design that models it: a randomised complete block with replicates within blocks, or a factorial design with block as a factor.
- When the number of treatments exceeds what fits in a block (e.g., 8 treatments but only 4 samples per plate), use an incomplete block design (BIBD).
- Always include randomisation within each block – the "R" in RCBD is essential for validity, even though the formula structure looks the same.

16.22.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.22.10 See also — labs in this chapter

- Observational designs and STROBE

- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.23 Randomisation in Design

16.23.1 Introduction

Randomisation is the protective heart of experimental design. It ensures that in expectation, unknown sources of variation are balanced across treatments; it validates the model's assumption of independent errors; and it prevents systematic bias from lurking variables.

16.23.2 Prerequisites

Experimental design basics.

16.23.3 Theory

Three Fisher principles of DoE: replication, randomisation, blocking. Randomisation: - Ensures **expected balance** of known and unknown covariates. - Justifies the random-error model used in ANOVA. - Validates inference (significance tests are approximate randomisation tests).

Randomisation procedures vary by design: CRD = unrestricted, RCBD = within-block, Latin square = row-column, split-plot = nested.

16.23.4 Assumptions

Randomisation is genuinely random (computer-generated sequence, sealed envelopes); allocation is concealed from the experimenter until assignment.

16.23.5 R Implementation

```
set.seed(2026)

# Randomise 12 units to 4 treatments for a CRD
```

```r
treatments <- rep(c("A", "B", "C", "D"), 3)
allocation <- sample(treatments, 12, replace = FALSE)
allocation

# For a RCBD, randomise within each of 3 blocks
blocks <- rep(1:3, each = 4)
alloc_rcbd <- do.call(rbind, lapply(split(1:12, blocks), function(idx) {
  data.frame(unit = idx, treatment = sample(c("A", "B", "C", "D")))
}))
alloc_rcbd
```

16.23.6 Output & Results

A randomised allocation list; randomisation is within block for RCBD to preserve block structure.

16.23.7 Interpretation

"Experimental units were randomly allocated to 4 treatments using a computer-generated sequence (seed 2026); in the RCBD, randomisation was within each block to preserve the local-control benefit."

16.23.8 Practical Tips

- Document the randomisation seed and procedure; reproducibility is part of responsible reporting.
- Conceal allocation from experimenters until assignment; otherwise bias can creep in.
- Randomise run order as well as treatment allocation to guard against temporal confounding.
- Restricted randomisation (within blocks, strata) is required for blocked designs; unrestricted randomisation can mismatch the analysis.
- Randomisation tests (permutation inference) are a rigorous alternative to parametric tests when distributional assumptions fail.

16.23.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.23.10 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.24 Repeated-Measures Designs

16.24.1 Introduction

Repeated-measures designs measure the same experimental unit at multiple time points or under multiple conditions. They gain statistical power relative to between-subject designs by removing the between-subject component of variance from the within-subject contrasts of interest, but they introduce a within-subject correlation structure that must be modelled correctly. The classical repeated-measures ANOVA requires sphericity and is restricted to balanced complete designs; the modern alternative is mixed-effects modelling with subject as a random effect, which accommodates unbalanced data, missing time points, and richer covariance structures while remaining valid under broader assumptions.

16.24.2 Prerequisites

A working understanding of one-way and factorial ANOVA, the concept of correlated data, and the essentials of mixed-effects modelling with random intercepts and (optionally) random slopes.

16.24.3 Theory

The standard mixed-effects model for repeated measures is

$$y_{ij} = \mu + \tau_j + s_i + \varepsilon_{ij}, \qquad s_i \sim N(0, \sigma_s^2), \quad \varepsilon_{ij} \sim N(0, \sigma^2),$$

with within-subject factor effect τ_j, subject random intercept s_i, and residual error ε_{ij}. Common covariance structures imposed on the residual vector for one subject include:

- **Compound symmetry**: equal residual variance and equal correlation across all time pairs (strict — implied by a random-intercept model).
- **AR(1)**: autoregressive of order one, with correlation decaying with lag.
- **Unstructured**: fully general $T \times T$ covariance with $T(T+1)/2$ parameters.

For classical repeated-measures ANOVA, sphericity is the relevant assumption, and Greenhouse–Geisser or Huynh–Feldt corrections are applied when Mauchly's test detects a violation.

16.24.4 Assumptions

The within-subject correlation structure is correctly specified, residuals are approximately Normal, and missing data are missing at random (MAR) — a standard assumption that mixed-effects models can accommodate without listwise deletion.

16.24.5 R Implementation

```r
library(lme4); library(lmerTest)

set.seed(2026)
n_sub <- 30; n_time <- 4
sub_re <- rnorm(n_sub, 0, 1)

df <- expand.grid(subject = 1:n_sub, time = 1:n_time)
df$y <- 10 + 0.5 * df$time + sub_re[df$subject] +
        rnorm(nrow(df), 0, 0.5)

fit_ri <- lmer(y ~ factor(time) + (1 | subject), data = df)
anova(fit_ri)

fit_rs <- lmer(y ~ time + (time | subject), data = df)
summary(fit_rs)$coefficients
```

16.24.6 Output & Results

`lmer()` with a random intercept gives the within-subject contrast for time, and `anova()` from `lmerTest` provides F-tests with Satterthwaite-adjusted denominator df. Adding a random slope (`(time|subject)`) lets the within-subject trend itself vary across subjects, which is appropriate when there is reason to believe individuals differ in their response trajectories.

16.24.7 Interpretation

A reporting sentence: "Repeated-measures analysis with subject as a random intercept showed a significant time effect ($F_{3,87} = 18.2$, $p < 0.001$). The subject random effect absorbed substantial between-subject variance ($\sigma_s^2 = 1.04$, $\sigma^2 = 0.27$, intra-class correlation 0.79), markedly improving precision relative to a between-subject analysis. A random-slope model that allowed subject-specific time trends did not improve fit ($\Delta\text{AIC} = +1.4$), so the random-intercept specification is reported as primary." Always report variance components.

16.24.8 Practical Tips

- Use mixed-effects models (`lmer` from `lme4`, `lme` from `nlme`, or generalised-mixed via `glmmTMB`) rather than classical repeated-measures ANOVA whenever possible; they handle unbalanced data, missing time points, irregular timing, and richer covariance structures naturally.
- For irregular or continuous timing, specify time as a continuous covariate with random slopes; this is far more efficient than treating each time point as a separate factor level when the response is genuinely a smooth function of time.
- If you must use classical repeated-measures ANOVA (e.g., for compatibility with a standard reporting requirement), check sphericity with Mauchly's test and apply Greenhouse–Geisser or Huynh–Feldt correction whenever it is violated.
- Missing time points are handled naturally by mixed-effects models under the MAR assumption; classical repeated-measures ANOVA requires complete cases and discards subjects with any missing time, which is wasteful and can introduce bias.
- Report both the fixed-effect estimates (the substantive scientific output) and the random-effect variance components (which describe how much within-subject correlation is present); the latter is essential for judging whether the design is well chosen.
- For binary or count outcomes measured repeatedly, use generalised mixed-effects models (`glmer`, `glmmTMB`) or generalised estimating equations (GEE via the `geepack` package) — the latter trade efficiency for robust marginal inference.

16.24.9 R Packages Used

`lme4::lmer()` and `lmerTest` for mixed-effects analysis with denominator-df adjustment; `nlme::lme()` for richer covariance-structure specification (AR1, unstructured) on the residuals; `glmmTMB` for generalised and complex-covariance mixed models; `geepack::geeglm()` for population-averaged GEE analyses; `afex::aov_car()` for classical repeated-measures ANOVA with built-in sphericity correction.

16.24.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.24.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.25 Resolution and Aliasing

16.25.1 Introduction

In any fractional factorial design, effects are aliased — two or more distinct effects produce the same contrast in the data and cannot be disentangled from the experimental results alone. The choice of fraction determines exactly which effects are aliased with which, and the design's **resolution** summarises the severity of this aliasing in a single Roman numeral. Higher resolution means lower-order effects are protected from aliasing with each other; lower resolution means smaller designs but messier confounding. Understanding resolution and the alias chain is essential for interpreting any fractional-factorial result, because an unrecognised alias is one of the most common sources of misleading DoE conclusions.

16.25.2 Prerequisites

Familiarity with fractional factorial designs, the algebra of defining relations and word lengths, and the effect-sparsity principle that motivates accepting higher-order aliasing.

16.25.3 Theory

A fractional factorial is defined by one or more **defining relations** of the form $I = ABCD\cdots$, where each word represents a product of factor letters. The defining relation determines the alias structure: multiplying any effect by the defining word (modulo squaring out $A^2 = I$) produces its alias. For example, with $I = ABCD$, we have $A = BCD$, $AB = CD$, and so on. The **resolution** of the design is the length of the shortest word in the defining relation:

- **Resolution III**: the shortest defining word has length 3, so main effects are aliased with two-factor interactions ($A = BC$). Suitable for screening only.
- **Resolution IV**: shortest word length 4; main effects are clear of two-factor interactions, but two-factor interactions alias with each other ($AB = CD$).
- **Resolution V**: shortest word length 5; main effects and all two-factor interactions are mutually clear; only three-factor and higher interactions are aliased with two-factor ones.

16.25.4 Assumptions

The effect-sparsity principle holds — higher-order interactions are negligible compared with lower-order effects, so accepting aliasing among higher-order terms is acceptable. The chosen resolution is appropriate for the scientific question, and the alias chain is documented in the analysis.

16.25.5 R Implementation

```r
library(FrF2)

res3 <- FrF2(nruns = 8, nfactors = 7)
design.info(res3)$aliased

res4 <- FrF2(nruns = 32, nfactors = 7, resolution = 4)
design.info(res4)$aliased

res6 <- FrF2(nruns = 32, nfactors = 6, resolution = 6)
design.info(res6)$aliased
```

16.25.6 Output & Results

`design.info()` reports the complete alias structure of the design — the defining relation, the resolution, and the chain of effects aliased together. Inspecting this output before analysis is essential: it tells the analyst exactly which estimates can be interpreted as pure main or interaction effects and which represent the joint estimate of an aliased pair.

16.25.7 Interpretation

A reporting sentence: "The 2^{7-4}_{III} design used 8 runs to screen seven two-level factors, with main effects aliased with two-factor interactions; the design is appropriate when only main effects are expected to matter and would not be appropriate if interaction effects were of interest. The 2^{7-2}_{IV} design used 32 runs to study the same factor set with main effects clear of two-factor interactions, at the cost of four-fold more experimentation. The complete alias chain is reported in Supplementary Table S1." Always document resolution and alias chain.

16.25.8 Practical Tips

- Resolution III designs (and Plackett–Burman designs at run counts not a power of 2) are appropriate for early screening when interactions are confidently negligible; never rely on a resolution-III design for inference about interactions or for confirmatory analysis.
- Resolution IV is the standard choice for initial investigation of many factors when the experimentalist wants clean main effects without paying the run-count cost of resolution V.
- Resolution V and above are appropriate for fitting models that include two-factor interactions; many industrial DoE protocols specify resolution V as the minimum for response-surface follow-up designs.
- Fold-over augmentation — adding a mirror-image fraction with all factor signs reversed — upgrades a resolution III design to resolution IV by de-aliasing main effects from two-factor interactions, at the cost of doubling the run count.
- Always document the alias structure explicitly in reports and protocols; reviewers and downstream analysts cannot interpret the results correctly without knowing which effects can be separated from which.
- For mixed-level fractional designs and irregular run counts, software-generated D-optimal designs may achieve cleaner partial aliasing than the standard regular fractions; consider them when the standard catalogue does not match your factor structure.

16.25.9 R Packages Used

`FrF2::FrF2()` and `FrF2::design.info()` for fractional factorial construction with explicit resolution control and alias-structure documentation; `DoE.base` for general orthogonal-array DoE with comprehensive alias reporting; `unrepx` for unreplicated-design effect identification under aliased structures; `BsMD` for Bayesian screening of fractional designs; `daewr` for textbook companion datasets.

16.25.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.25.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.26 Response Surface Methodology

16.26.1 Introduction

Response surface methodology (RSM), introduced by Box and Wilson in 1951, is a sequential experimental strategy for finding and characterising the optimum of a process response in the space of multiple controllable factors. RSM combines fractional or full factorial screening, steepest-ascent search, and second-order polynomial fitting near the optimum into a coherent workflow that has become standard practice in chemical engineering, food science, pharmaceutical formulation, manufacturing process improvement, and any setting where a smooth response surface must be navigated efficiently. Its hallmark is the staged, sequential character: each stage of experimentation informs the next, and only the final stage commits resources to the dense second-order designs needed for full characterisation.

16.26.2 Prerequisites

A working understanding of factorial designs, polynomial regression for response-surface modelling, gradient-based optimisation concepts, and the canonical form of a quadratic surface.

16.26.3 Theory

RSM operates in two main phases:

1. **First-order phase**: a fractional or full factorial with replicated centre points fits a linear model in a restricted operating region. The path of steepest ascent (or descent) along the fitted gradient guides further experimentation toward the optimum, with confirmation runs along the path.
2. **Second-order phase**: once experimentation is near the optimum, a central composite or Box-Behnken design fits a full second-order polynomial. Canonical analysis decomposes the quadratic-form matrix into eigenvalues, classifying the stationary point as a maximum, minimum, saddle, or rising/stationary ridge.

The canonical form is what makes RSM more than just curve-fitting: it characterises the geometry of the response surface and reveals when the optimum is genuinely a peak, when it lies on a ridge requiring constrained search, and when further experimentation is needed.

16.26.4 Assumptions

The response is a smooth, well-defined function of the factors over the design region; a second-order polynomial is an adequate local approximation in the neighbourhood of the optimum; residual error is Normal with constant variance; runs are independent.

16.26.5 R Implementation

```
library(rsm)

set.seed(2026)
ccd <- ccd(~ x1 + x2, n0 = c(3, 3), alpha = "rotatable",
           randomize = FALSE)
ccd$y <- 10 + 2 * ccd$x1 + 1.5 * ccd$x2 -
         1.2 * ccd$x1^2 - 0.9 * ccd$x2^2 +
         0.6 * ccd$x1 * ccd$x2 +
         rnorm(nrow(ccd), 0, 0.4)

fit <- rsm(y ~ SO(x1, x2), data = ccd)
summary(fit)

contour(fit, ~ x1 + x2, at = c(0, 0), image = TRUE,
        main = "RSM response surface")

canonical(fit)
```

16.26.6 Output & Results

The `rsm()` summary returns linear, quadratic, and interaction coefficients with significance tests, plus the canonical-analysis classification (maximum, mini-

mum, saddle, or ridge) based on the eigenvalues of the quadratic-form matrix. `contour()` and `persp()` render the surface visually, and `canonical()` reports the stationary point in original and canonical coordinates.

16.26.7 Interpretation

A reporting sentence: "Response-surface methodology with a 14-run rotatable CCD identified a concave surface with stationary point at $(x_1^*, x_2^*) = (0.85, 0.72)$. Canonical analysis confirmed a true maximum (both eigenvalues negative: $\lambda_1 = -1.31$, $\lambda_2 = -0.79$); predicted peak response 12.1, with a 95 % prediction interval of [11.5, 12.7]. Confirmation runs at the predicted optimum yielded 12.0 (mean of 3 replicates), within the interval and supporting the model." Always validate the predicted optimum with confirmation runs.

16.26.8 Practical Tips

- The first-order steepest-ascent phase can save many runs; don't jump straight to a second-order fit if the operating region is far from the optimum, because a quadratic fit far from the peak yields meaningless coefficients and wasted runs.
- Always inspect contour and 3D surface plots in addition to the coefficient table; numerical summaries can hide ridges, saddles, and other surface features that are obvious visually.
- Canonical analysis reveals the nature of the stationary point through the signs of the eigenvalues: all negative = maximum, all positive = minimum, mixed signs = saddle, near-zero = ridge requiring constrained or directional search.
- When the predicted optimum lies outside the experimental design region, use ridge analysis (constrained optimisation along the path of steepest ascent within the cube) rather than naive extrapolation, which is dangerous and statistically unsupported.
- Follow RSM with confirmation runs at the predicted optimum and inside the prediction interval; falsified predictions reveal model inadequacy and motivate a higher-order fit or a re-positioned design region.
- For multiple responses (a common situation in industrial work — yield, purity, cost), use desirability functions to combine the surfaces into a single optimisation criterion rather than optimising each response independently.

16.26.9 R Packages Used

`rsm` for canonical RSM construction and analysis (CCD, BBD, second-order fits, canonical analysis, steepest-ascent paths, contour and 3D plots); `desirability` for multi-response desirability-function optimisation; `qualityTools` for an alternative RSM toolkit; `Vdgraph` for variance-dispersion graphs comparing RSM design alternatives; `daewr` for textbook companion examples and `DoE.wrapper` for higher-level design-workflow integration.

16.26.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.26.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.27 Robust Parameter Design

16.27.1 Introduction

Robust parameter design (Taguchi, 1950s-60s) seeks combinations of controllable design factors that minimise response variability across noise factors (environmental, batch-to-batch, use-conditions). The goal is a product or process that performs consistently regardless of noise.

16.27.2 Prerequisites

Factorial designs; variance decomposition.

16.27.3 Theory

Inner array of control factors crossed with **outer array** of noise factors. At each control setting, compute a summary – mean, standard deviation, or signal-to-noise ratio. Choose the control setting that optimises the summary.

Modern approach: fit a response model with control, noise, and control-x-noise interactions; find control levels that minimise variance while meeting the target mean.

16.27.4 Assumptions

Noise factors represent realistic variation at use time; interactions with control factors are the primary lever for variance reduction.

16.27.5 R Implementation

```
library(FrF2)

set.seed(2026)
# Inner array: 2^3 controllable factors
ctrl <- FrF2(nruns = 8, nfactors = 3, randomize = FALSE)
ctrl$A <- as.numeric(as.character(ctrl$A))
ctrl$B <- as.numeric(as.character(ctrl$B))
ctrl$C <- as.numeric(as.character(ctrl$C))

# Outer array: 2^2 noise factors
noise <- expand.grid(N1 = c(-1, 1), N2 = c(-1, 1))

# Simulate: noise x control interaction
results <- do.call(rbind, lapply(1:nrow(ctrl), function(i) {
  vals <- sapply(1:nrow(noise), function(j) {
    10 + 2 * ctrl$A[i] + 1.5 * ctrl$B[i] +
      0.5 * noise$N1[j] + 0.8 * ctrl$A[i] * noise$N1[j] +
      rnorm(1, 0, 0.3)
  })
  data.frame(run = i, mean = mean(vals), sd = sd(vals),
             snr = -10 * log10(var(vals)))
}))
results

# Choose control level minimising variance
results$A <- ctrl$A
cbind(ctrl, results)[order(results$sd), ]
```

16.27.6 Output & Results

Per-control-run mean, SD, and signal-to-noise ratio; the control combination with the smallest SD is the robust-optimal.

16.27.7 Interpretation

“Robust parameter design identified setting (A = -1, B = +1, C = +1) as minimising response SD across noise factors. Control factor A interacted with noise factor N1, explaining the variance reduction at A = -1.”

16.27.8 Practical Tips

- The inner-outer array approach can be computationally expensive; use a combined model with control-noise interactions when possible.
- Report both mean and variance; robust designs often accept a small mean-response penalty for large variance reduction.
- Consider dual-response surface methods for joint mean-variance optimisation.
- Pay attention to control-noise interactions: that is where robustness lives.
- Combining Taguchi's philosophy with modern RSM is more flexible than pure Taguchi methods.

16.27.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.27.10 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.28 Signal-to-Noise Ratios

16.28.1 Introduction

Signal-to-noise (SNR) ratios in Taguchi's framework jointly summarise the mean and variance of a response. Three variants handle different quality targets: smaller-is-better, larger-is-better, and nominal-is-best. Maximising SNR selects factor combinations that produce both a desirable mean and low variance.

16.28.2 Prerequisites

Taguchi methods; variance and mean summaries.

16.28.3 Theory

- **Smaller-is-better** (minimisation): $\text{SNR} = -10 \log_{10}\left(\frac{1}{n}\sum y_i^2\right)$.
- **Larger-is-better** (maximisation): $\text{SNR} = -10 \log_{10}\left(\frac{1}{n}\sum 1/y_i^2\right)$.
- **Nominal-is-best** (hit target τ): $\text{SNR} = 10 \log_{10}(\bar{y}^2/s^2)$.

Higher SNR is always better. Bias: SNRs conflate mean and variance; variance reduction can increase SNR even when mean drifts.

16.28.4 Assumptions

Outcomes are non-negative (for smaller/larger-is-better); target is appropriate; replicates exist to estimate variance.

16.28.5 R Implementation

```
set.seed(2026)

# Smaller-is-better: defects per batch
defects <- matrix(rpois(20 * 3, lambda = 4), 20, 3)
snr_sb <- -10 * log10(rowMeans(defects^2))

# Larger-is-better: process yield
yield <- matrix(rnorm(20 * 3, 80, 5), 20, 3)
snr_lb <- -10 * log10(rowMeans(1 / yield^2))

# Nominal-is-best: target 50
values <- matrix(rnorm(20 * 3, 50, 3), 20, 3)
means <- rowMeans(values); sds <- apply(values, 1, sd)
snr_nom <- 10 * log10(means^2 / sds^2)

head(data.frame(snr_sb, snr_lb, snr_nom))
```

16.28.6 Output & Results

Per-batch SNR values under the three variants; the batch maximising each SNR has the desirable combination of mean and variance.

16.28.7 Interpretation

"Smaller-is-better SNR was -12.5 (batch 3) vs -14.7 (batch 10); batch 3 achieved lower mean and lower variance of defects. The larger-is-better SNR identified a different optimal batch, suggesting SNRs depend on target."

16.28.8 Practical Tips

- SNR combines mean and variance – convenient but conflates them. Modern practice reports mean and variance separately.
- Dual-response surface methods (Myers-Carter, Vining-Myers) extend SNR thinking by fitting separate mean and variance models.
- For nominal-is-best, ensure $\bar{y}$ hits target before interpreting SNR.
- SNR is sensitive to sample size; fewer replicates inflate or deflate the metric.
- Report SNR in dB (decibels, i.e. using $10\log_{10}$), to follow Taguchi conventions.

16.28.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.28.10 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.29 Split-Plot Designs

16.29.1 Introduction

Split-plot designs arise whenever some experimental factors are hard to change between runs while others are easy to change within a run. Oven temperature, pasture treatment, baking time, batch-level chemical additive, or instrument calibration are typically hard to change — they can be set once per batch but not freely between samples. By contrast, additive level within a batch, fertiliser sub-plot within a paddock, or sample-level annotation can be randomised within

the larger group. Restricted randomisation of the hard-to-change factor at the whole-plot level and unrestricted randomisation of the easy-to-change factor within whole plots produces a design with two error strata, each requiring its own appropriate F-test denominator. Recognising the split-plot structure rather than ignoring it is essential to valid inference.

16.29.2 Prerequisites

A working understanding of factorial designs, mixed-effects models with crossed and nested random effects, and the concept of multiple error strata in restricted-randomisation experiments.

16.29.3 Theory

The standard split-plot model is

$$y_{ijk} = \mu + \alpha_i + \delta_{ij} + \beta_k + (\alpha\beta)_{ik} + \varepsilon_{ijk},$$

with whole-plot effect α_i, whole-plot error $\delta_{ij} \sim N(0, \sigma^2_W)$, sub-plot effect β_k, interaction $(\alpha\beta)_{ik}$, and sub-plot error $\varepsilon_{ijk} \sim N(0, \sigma^2_S)$. The whole-plot factor α is tested against the whole-plot error, while the sub-plot factor β and the interaction $\alpha\beta$ are tested against the (typically smaller) sub-plot error. As a result, the sub-plot factor and the interaction usually have more power than the whole-plot factor — a structural asymmetry that should inform which factor is assigned to which stratum.

16.29.4 Assumptions

Whole-plot errors are independent across whole plots; sub-plot errors are independent within and across whole plots; both error components are Normal with constant variance; the random-effects structure is correctly specified in the analysis.

16.29.5 R Implementation

```
library(lme4); library(lmerTest)

set.seed(2026)
n_rep <- 2; n_wp <- 3; n_sp <- 4
df <- expand.grid(rep = 1:n_rep, temp = factor(1:n_wp),
                  form = factor(1:n_sp))
df$batch <- interaction(df$rep, df$temp)
batch_re <- rnorm(nlevels(df$batch), 0, 0.6)[df$batch]

df$y <- 10 + 0.5 * as.numeric(df$temp) + 0.7 * as.numeric(df$form) +
```

```
        batch_re + rnorm(nrow(df), 0, 0.5)

fit <- lmer(y ~ temp * form + (1 | batch), data = df)
anova(fit)
```

16.29.6 Output & Results

`lmer()` with a `(1|batch)` random effect captures the whole-plot stratum (each batch corresponds to one whole-plot setting of temperature), and the residual captures the sub-plot stratum. `anova()` from `lmerTest` returns F-tests with Satterthwaite or Kenward-Roger denominator df, automatically routing each effect against its appropriate stratum.

16.29.7 Interpretation

A reporting sentence: "The split-plot mixed model showed a significant whole-plot effect of oven temperature ($F_{2,4} = 5.2$, $p = 0.04$, against batch-level error) and a highly significant sub-plot effect of formulation ($F_{3,18} = 14.8$, $p < 0.001$, against residual error). The temperature $\times$ formulation interaction was non-significant ($F_{6,18} = 1.1$, $p = 0.40$). Sub-plot effects were tested with substantially more power because the sub-plot error was much smaller than the whole-plot error ($\sigma^2_S = 0.25$ vs $\sigma^2_W = 0.36$)." Always identify which stratum each F-test uses.

16.29.8 Practical Tips

- Always match the error term to the effect's level in the design; analysing a split-plot as if it were a CRD treats whole-plot and sub-plot effects with the wrong denominator and inflates or deflates significance for the wrong factor.
- Mixed-effects models (`lmer` from `lme4`) give correct standard errors and F-tests when the random-effect structure is specified correctly; explicit attention to the random-effect formula is the single most common source of split-plot analysis errors.
- Replicate whole plots whenever possible — whole-plot F-tests have very few df otherwise. With three whole-plot levels and a single replicate, the whole-plot test has essentially no power; a second replicate at minimum is recommended.
- Split-plot structure arises naturally in agricultural field experiments (treatments applied to whole strips, sub-treatments within), manufacturing (batch-level process settings, within-batch additives), and instrumentation studies (calibration once per session, sample-level treatments within session).
- Strip-plot designs extend the split-plot idea to two hard-to-change factors applied in perpendicular directions; they have three error strata and

require correspondingly more careful random-effects specification.
- Assigning a factor to the whole-plot stratum reduces its detection power; if a factor is scientifically critical, assign it to the sub-plot stratum if possible — even at the cost of some operational inconvenience.

16.29.9 R Packages Used

`lme4::lmer()` and `lmerTest` for mixed-effects analysis with denominator-df adjustment; `agricolae::design.split()` for canonical split-plot layout construction; `daewr` for textbook companion datasets and analysis helpers; `emmeans` for marginal means and contrasts at each stratum; `nlme::lme()` for an alternative interface with explicit fixed/random specification.

16.29.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.29.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.30 Method of Steepest Ascent

16.30.1 Introduction

The method of steepest ascent is the workhorse of the first phase of response-surface methodology. After fitting a linear (first-order) model to a small factorial experiment in the current operating region, the gradient vector points in the direction of fastest predicted improvement; subsequent confirmation runs are

taken sequentially along this gradient, moving the experimental region toward the neighbourhood of the optimum. The strategy is efficient because it replaces an exhaustive grid search of the factor space with a directed walk informed by the local linear approximation, dramatically reducing the number of runs required to locate the region near a process optimum.

16.30.2 Prerequisites

A working understanding of first-order linear regression, the concept of the gradient vector, and the sequential staging of experimentation in response-surface methodology.

16.30.3 Theory

Fit a first-order linear model

$$\hat{y} = \beta_0 + \sum_i \beta_i x_i$$

in the current operating region using a small two-level factorial with replicated centre points. The steepest-ascent direction is the gradient $\nabla \hat{y} = (\beta_1, \dots, \beta_k)$, normalised to unit length. Subsequent runs are taken at successive multiples of a chosen step size along this direction; experimentally observed responses guide whether to continue, slow down, or stop. When the response stops improving along the path, a new first-order design is fit at the new operating region and the procedure iterates.

16.30.4 Assumptions

The local first-order linear approximation is adequate in the current region (no strong curvature would invalidate the gradient direction), the response surface is smooth, and the step size along the path is appropriate — too small and the search wastes runs, too large and it overshoots the optimum. Centre-point replicates allow detection of curvature via lack-of-fit, signalling when to switch to a second-order fit.

16.30.5 R Implementation

```
library(rsm)

set.seed(2026)
region1 <- data.frame(x1 = c(-1, 1, -1, 1, 0),
                      x2 = c(-1, -1, 1, 1, 0))
region1$y <- 10 + 2 * region1$x1 + 1.5 * region1$x2 +
             rnorm(nrow(region1), 0, 0.3)
```

```
fit1 <- rsm(y ~ FO(x1, x2), data = region1)
summary(fit1)

direction <- steepest(fit1, dist = c(0.5, 1, 2, 3, 5, 7))
direction
```

16.30.6 Output & Results

`rsm()` with the first-order specification returns the gradient coefficients with significance tests and any lack-of-fit information from centre replicates. `steepest()` traces the path of steepest ascent at requested distances from the design centre, producing predicted responses along the path that guide the experimentalist's choice of follow-up runs.

16.30.7 Interpretation

A reporting sentence: "The first-order fit in the initial operating region gave coefficients $\hat{\beta}_1 = 2.0$ and $\hat{\beta}_2 = 1.5$, both highly significant ($p < 0.001$). The steepest-ascent path moved in the direction $(0.80, 0.60)$ with step ratio approximately $2 : 1.5$. Confirmation runs at distances 1, 3, 5, and 7 from the centre showed monotone increase to a plateau at distance 5; a new first-order design was then fit centred there, revealing significant curvature ($p = 0.004$ from centre-point lack-of-fit), and a second-order CCD was used for final characterisation." Always describe the path and the stopping criterion.

16.30.8 Practical Tips

- Take the first few steps along the path experimentally rather than relying purely on numerical extrapolation; the linear approximation is local, and only experimental confirmation reveals when it begins to break down.
- Stop moving along the path when the response plateaus or decreases, then fit a new first-order design at the new operating region and repeat the process; this iterative restart is the key to efficient convergence on the optimum.
- When the gradient coefficients are roughly equal in magnitude, the ascent direction is diagonal in the factor space; when one dominates, the path is primarily along that axis. Inspecting the path on a contour plot makes the direction concrete and informs step-size choice.
- If the first-order fit shows significant curvature (a centre-point lack-of-fit test rejects), the linear approximation is inadequate and steepest ascent should be replaced by a second-order fit (CCD, BBD) for final characterisation.
- Steepest ascent is unconstrained; if factor levels have hard physical or operational constraints, use ridge analysis (constrained optimisation along

the direction of steepest improvement, restricted to the feasible region) instead.
- Pre-specify the step-size schedule and the stopping criterion in the experimental protocol; ad-hoc decisions during a steepest-ascent run are a recurring source of irreproducible RSM results.

16.30.9 R Packages Used

`rsm::FO()` for first-order model specification, `rsm::steepest()` for the canonical path-of-steepest-ascent computation, `rsm::ridge()` for constrained ridge analysis when factor bounds bind; `qualityTools` for an alternative RSM toolkit; `desirability` for multi-response steepest-ascent equivalents under desirability-function optimisation; `daewr` for textbook companion datasets.

16.30.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.30.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.31 Strip-Plot Designs

16.31.1 Introduction

Strip-plot designs — also known as split-block designs — apply two hard-to-change factors in perpendicular strips across a block, with the resulting factor

combinations realised in the cells of the cross. They arise naturally in agricultural field experiments where one factor (e.g., irrigation level) is applied along rows and another (e.g., fertiliser type) along columns, in industrial manufacturing where two stages of processing each cover an entire pallet, and in food and pharmaceutical settings where two restrictions on randomisation operate in different directions. The defining feature is three independent error strata — one for the row factor, one for the column factor, and one for their interaction — each requiring its own appropriate F-test denominator.

16.31.2 Prerequisites

A working understanding of split-plot designs, the concept of multiple error strata, and mixed-effects model specification with nested random effects.

16.31.3 Theory

The strip-plot model has three error strata corresponding to the three sources of restricted randomisation:

- **Row factor**: applied to entire rows; tested against row-stratum error (variation among rows within blocks).
- **Column factor**: applied to entire columns; tested against column-stratum error.
- **Row × column interaction**: tested against the cell-level (smallest) error.

Each main effect therefore has lower power than it would in a completely randomised design, because its denominator is built from a coarser stratum, but the design is operationally feasible when full randomisation of the two hard-to-change factors is impractical or prohibitively expensive.

16.31.4 Assumptions

The three error strata are independent, errors are Normal and homoscedastic at each stratum, and the random-effects structure is correctly specified — typically random effects for row-within-block and column-within-block, with the cell-level residual capturing the interaction-level error.

16.31.5 R Implementation

```
library(lme4); library(lmerTest)

set.seed(2026)
n_rep <- 3
rows <- factor(rep(1:3, each = 4, times = n_rep))
cols <- factor(rep(rep(1:4, 3), n_rep))
rep_f <- factor(rep(1:n_rep, each = 12))
```

```
df <- data.frame(rep = rep_f, rows, cols)

df$row_batch <- interaction(df$rep, df$rows)
df$col_batch <- interaction(df$rep, df$cols)

row_re <- rnorm(nlevels(df$row_batch), 0, 0.5)[df$row_batch]
col_re <- rnorm(nlevels(df$col_batch), 0, 0.4)[df$col_batch]

df$y <- 10 + 0.6 * as.numeric(df$rows) + 0.4 * as.numeric(df$cols) +
        row_re + col_re + rnorm(nrow(df), 0, 0.3)

fit <- lmer(y ~ rows * cols + (1 | row_batch) + (1 | col_batch),
            data = df)
anova(fit)
```

16.31.6 Output & Results

`lmer()` with two crossed random-effect terms (row-within-replicate and column-within-replicate) and the cell-level residual recovers the three error strata. `anova()` returns the appropriate F-tests for the row and column main effects against their respective stratum errors and for the interaction against the cell-level error. Reporting all three F-tests with their stratum-specific denominator df is the standard summary.

16.31.7 Interpretation

A reporting sentence: "The strip-plot ANOVA showed the row factor was marginally significant ($F_{2,4} = 3.8$, $p = 0.05$ against row-stratum error), the column factor was significant ($F_{3,6} = 6.4$, $p = 0.01$ against column-stratum error), and the row $\times$ column interaction was non-significant when tested against the cell-level error ($F_{6,24} = 1.2$, $p = 0.34$). Random-effect variance components for row-within-replicate and column-within-replicate were modest, indicating the strip structure absorbed limited but non-trivial nuisance variation." Always report each effect against its appropriate stratum.

16.31.8 Practical Tips

- Strip-plot designs require replication across blocks (multiple replicates of the cross structure) for valid inference; a single block leaves no error degrees of freedom for the row or column main effects.
- Strip-plot structure is most natural in agricultural field experiments where two perpendicular operations (e.g., ploughing direction and spraying direction) are applied independently, and in industrial manufacturing where two upstream processes cover whole batches in different ways.

- Correct random-effects specification is critical; an incorrect model treats restricted randomisation as if it were complete, yielding F-tests with the wrong denominator and inflated false-positive rates.
- Use `lmerTest::ranova()` to test the significance of the random-effect variance components themselves; this also helps diagnose whether the strip structure was substantively different from a CRD analysis would have implied.
- In industrial settings, strip-plot structure arises whenever two manufacturing stages each apply to a whole pallet, batch, or run before disaggregating; recognising the structure rather than ignoring it is essential to valid inference.
- For analysis software-checks, fit the same dataset with the wrong (CRD-like) model and the right (strip-plot) model; comparing the two highlights the inferential cost of ignoring the design and is a useful teaching diagnostic.

16.31.9 R Packages Used

`lme4::lmer()` and `lmerTest` for mixed-effects analysis with denominator-df adjustment; `agricolae::design.strip()` for the canonical strip-plot layout construction; `daewr` for textbook-companion strip-plot datasets and analysis helpers; `emmeans` for marginal means and contrasts at each stratum.

16.31.10 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.31.11 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve
- Pre-registration and statistical analysis plans

16.32 Taguchi Methods

16.32.1 Introduction

Taguchi methods (1950s-60s) apply orthogonal-array experiments to optimise product and process design for minimum variability. Central ideas: loss functions defining quality as deviation from target, signal-to-noise (SNR) ratios summarising mean and variance jointly, and compact orthogonal arrays for efficient screening.

16.32.2 Prerequisites

Fractional factorial designs; signal-to-noise.

16.32.3 Theory

Quality loss $L(y) = k(y - \tau)^2$ for target τ – quality decreases quadratically with deviation.

SNR: - Smaller-is-better: $\text{SNR} = -10 \log_{10}(\sum y_i^2/n)$. - Larger-is-better: $\text{SNR} = -10 \log_{10}(\sum 1/y_i^2/n)$. - Nominal-is-best: $\text{SNR} = 10 \log_{10}(\bar{y}^2/s^2)$.

Taguchi arrays (L4, L8, L9, L16) are Plackett-Burman-style orthogonal arrays with Japanese quality-engineering conventions.

16.32.4 Assumptions

Main effects dominate (sparsity); variance reduction via control factor choice (robust design).

16.32.5 R Implementation

```
library(qualityTools)

set.seed(2026)
# L8 array (2-level, 7 factors)
ta <- taguchiDesign("L8_2^7")
summary(ta)

# Simulate responses
ta$y <- rnorm(8, mean = 10 + 2 * as.numeric(ta$A) +
                        1.5 * as.numeric(ta$B), sd = 0.5)
effectPlot(ta, response = "y",
           main = "Taguchi main-effects plot (L8)")
```

16.32.6 Output & Results

L8 orthogonal array and main-effects plot showing the average response at each factor's two levels.

16.32.7 Interpretation

"The L8 design screened 7 factors in 8 runs; the main-effects plot identified factors A and B as dominant, with the optimal settings maximising (or minimising) the target response."

16.32.8 Practical Tips

- Taguchi arrays are essentially Plackett-Burman designs; modern DoE uses statistical fractional factorials with aliasing awareness.
- SNR metrics conflate mean and variance; modern practice separates them via dual-response surface methods.
- Taguchi's three-stage design (system, parameter, tolerance) is a useful framework; the specific SNR tools are debated.
- Orthogonal arrays are efficient for screening; interactions must be explicitly assigned to specific columns.
- For manufacturing and quality engineering, Taguchi remains a widely-used vocabulary; modern methods supersede the specifics.

16.32.9 For Reviewers

Use the shared reviewer checklist, together with this method's assumptions and practical tips.

Method-specific reviewer notes have not yet been developed for this entry. The links above support a structured review; they do not certify the example or a manuscript's application of the method.

16.32.10 See also — labs in this chapter

- Observational designs and STROBE
- Randomised controlled trials
- Adaptive, non-inferiority, equivalence trials
- Bench and translational design
- MCAR, MAR, MNAR
- Multiple imputation with mice
- Linear mixed models with lme4
- GLMMs and GEE
- DAGs with dagitty and ggdag
- Propensity scores and IPTW
- G-methods, IV, DiD, RDD; heterogeneity of treatment effect
- SIR and SEIR models with deSolve

- Pre-registration and statistical analysis plans

16.33 Week 1, Session 3 — Adaptive, non-inferiority, equivalence trials

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.33.1 Learning objectives

- Distinguish superiority, non-inferiority, and equivalence trials by their null hypotheses.
- Choose a non-inferiority margin and say what it means clinically.
- Run and plot a non-inferiority test with a confidence interval versus the margin.

16.33.2 Prerequisites

Inferential statistics and confidence intervals.

16.33.3 Background

A superiority trial asks whether a new treatment beats the comparator. A non-inferiority (NI) trial asks whether the new treatment is *not worse than* the comparator by more than a pre-specified margin (often called Δ). An equivalence trial asks whether the new treatment is neither better nor worse than the comparator by more than Δ on either side. These are distinct statistical questions and they require different designs, analyses, and sample sizes.

The NI margin is the hardest thing about an NI trial. It has to be small enough that clinicians and patients would trade it for the benefits of the new treatment (lower cost, fewer side effects, easier administration), but not so small that the trial is infeasible. Regulators will often ask for a margin no larger than a defined fraction of the historical treatment effect of the comparator versus placebo.

Adaptive trials modify some aspect of the design (sample size, arm allocation, the decision rule) during the trial based on interim results, using pre-specified rules. Group sequential designs, for example, stop early for overwhelming efficacy or for futility, while spending Type I error according to a chosen alpha-spending function.

A one-sided 97.5% confidence interval that stays on the correct side of the margin is the operational test in most NI trials. Graphically, this is a forest plot where you check that the interval does not cross the margin line.

16.33.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.33.5 1. Hypothesis

H0: the new treatment is worse than the comparator by more than $\Delta = 3$ units on a continuous outcome. H1: the new treatment is not worse by more than Δ.

16.33.6 2. Visualise

```
n <- 150
trial <- tibble(
  arm = rep(c("comparator", "new"), each = n),
  y   = c(rnorm(n, 70, 10), rnorm(n, 69, 10))
)

trial |>
  ggplot(aes(arm, y, fill = arm)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "Outcome") +
  theme(legend.position = "none")
```

16.33.7 3. Assumptions

Independent observations, roughly normal within-arm residuals, and a pre-specified margin declared before looking at the data. The *direction* of the test matters: we test whether the lower bound of the mean difference (new − comparator) is above $-\Delta$.

16.33.8 4. Conduct

```
fit <- t.test(y ~ arm, data = trial)
fit
```

```
delta <- 3
ci <- fit$conf.int
est <- diff(rev(fit$estimate))  # new - comparator
ni_pass <- ci[1] > -delta
tibble(estimate = est, low = ci[1], high = ci[2], margin = -delta,
 ↪  ni_pass)
```

```
tibble(estimate = est, low = ci[1], high = ci[2]) |>
  ggplot(aes(x = estimate, y = 1)) +
  geom_point(size = 3) +
  geom_errorbarh(aes(xmin = low, xmax = high), height = 0.1) +
  geom_vline(xintercept = 0, linetype = "dashed", colour = "grey40") +
  geom_vline(xintercept = -delta, colour = "firebrick") +
  labs(x = "Mean difference (new - comparator)", y = NULL) +
  theme(axis.text.y = element_blank())
```

16.33.9 5. Concluding statement

> The new treatment had a mean difference of `round(est,2)` versus comparator (95% CI: `round(ci[1],2)` to `round(ci[2],2)`). With a pre-specified non-inferiority margin of $\Delta = 3$, the new treatment `if(ni_pass)"met"else"didnotmeet"` the non-inferiority criterion because the lower bound `if(ni_pass)"exceeded"else"fellbelow"` $-\Delta$.

16.33.9.1 A word on adaptive designs

Adaptive designs work when the adaptation rules and the alpha spent at each look are pre-specified. Running an interim analysis with the hope of extending the trial if it looks promising — without a pre- specified rule — inflates Type I error and destroys the trial's inferential warranty.

16.33.10 Common pitfalls

- Picking the NI margin after seeing the data.
- Using a two-sided superiority test and claiming non-inferiority.
- Running an "interim look" without a pre-specified spending rule.
- Treating equivalence and non-inferiority as interchangeable.

16.33.11 Further reading

- Piaggio G et al. (2012), *CONSORT extension for non-inferiority and equivalence trials*.
- Jennison C, Turnbull BW. *Group Sequential Methods with Applications to Clinical Trials*.
- FDA (2016), *Non-Inferiority Clinical Trials to Establish Effectiveness*.

16.33.12 Session info

```
sessionInfo()
```

16.33.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.34 Week 1, Session 4 — Bench and translational design

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.34.1 Learning objectives

- Recognise blocking, factorial, and split-plot designs in a bench setting.
- Simulate a 2×2 factorial experiment and test the interaction.
- Spot and correct a pseudoreplication error in a cell-culture example.

16.34.2 Prerequisites

Linear regression and ANOVA; familiarity with the formula interface.

16.34.3 Background

Bench and translational biology lives on small, carefully structured experiments whose designs are often misread as simple one-way comparisons. Blocking groups observations that share a nuisance variable (a batch, a day, a plate) so that comparisons are made within blocks. Factorial designs manipulate two or more factors at once and estimate main effects and interactions from a single experiment. Split-plot designs randomise two factors at different scales — a hard-to-change factor (oven temperature) at the whole- plot level, an easier one (position on the tray) at the sub-plot level — and the analysis must respect that structure.

Pseudoreplication is what happens when the experimental unit and the observational unit are confused. If one animal's blood is pipetted into ten wells, the ten wells are not ten independent samples; they are ten technical replicates of one biological replicate. Analysing them as independent inflates the apparent sample size, shrinks the standard error, and produces spuriously small p-values. The fix is to aggregate within the experimental unit or use a mixed model that knows about the nested structure.

A useful question to ask before every analysis: *what is the unit I could have independently randomised?* That is the unit of analysis. If an intervention was applied to a cage but measurements were taken on each mouse, the cage is the unit.

16.34.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.34.5 1. Hypothesis

A drug and a diet each affect a continuous biomarker, and their combination may have a non-additive effect (interaction).

16.34.6 2. Visualise

```
n_per_cell <- 10
fact <- expand_grid(
  drug = c("no", "yes"),
  diet = c("standard", "modified"),
  rep  = seq_len(n_per_cell)
) |>
  mutate(
    y = 10 +
      (drug == "yes")  * 1.5 +
      (diet == "modified") * 1.0 +
      (drug == "yes" & diet == "modified") * 2.0 +
      rnorm(n(), 0, 1.5)
  )

fact |>
  ggplot(aes(diet, y, fill = drug)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "Biomarker", fill = "Drug")
```

16.34.7 3. Assumptions

Independent observations *within each cell* (every replicate is a separate biological unit), approximately normal residuals, and equal variance across cells. For the pseudoreplication demonstration we deliberately break the independence assumption.

```
fact |>
  group_by(drug, diet) |>
  summarise(mean = mean(y), sd = sd(y), n = n(), .groups = "drop")
```

16.34.8 4. Conduct

```
fit <- lm(y ~ drug * diet, data = fact)
anova(fit)
```

Now a pseudoreplication error. Suppose each biological replicate was measured three times (technical replicates) and we naively treated the 3 × rows as independent.

```
pseudo <- fact |>
  slice(rep(seq_len(n()), each = 3)) |>
  mutate(tech_noise = rnorm(n(), 0, 0.3),
         y = y + tech_noise)

fit_bad  <- lm(y ~ drug * diet, data = pseudo)
fit_good <- lm(y ~ drug * diet, data = fact)  # aggregated (original)

bind_rows(
  bad  = broom::tidy(fit_bad)  |> filter(term == "drugyes:dietmodified"),
  good = broom::tidy(fit_good) |> filter(term == "drugyes:dietmodified"),
  .id = "analysis"
)
```

The point estimate barely moves; the standard error in the bad analysis is smaller because the apparent n is three times larger.

16.34.9 5. Concluding statement

> In a simulated 2 × 2 factorial experiment (10 biological replicates per cell), there was evidence of an interaction between drug and diet on the biomarker (F and p from the ANOVA table above). Naive analysis of technical replicates as independent observations shrinks the standard error of the interaction estimate without changing the point estimate — a textbook pseudoreplication artefact.

16.34.10 Common pitfalls

- Counting wells, sections, or reads as independent samples.
- Forgetting to include the block in the model when the design was blocked.
- Running a one-way ANOVA on a factorial design and losing the interaction.
- Split-plot designs analysed with a single-error-term model.

16.34.11 Further reading

- Montgomery DC. *Design and Analysis of Experiments.*

- Lazic SE (2010), *The problem of pseudoreplication in neuroscientific studies.*
- Festing MFW (2006), *Design and statistical methods in studies using animal models.*

16.34.12 Session info

```
sessionInfo()
```

16.34.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.35 Week 3, Session 3 — DAGs with dagitty and ggdag

Workflow lab using the variant template: Goal → Approach → Execution → Check → Report.

16.35.1 Learning objectives

- Express a causal scenario as a directed acyclic graph (DAG).
- Identify confounders, mediators, and colliders from a DAG.
- Use `dagitty::adjustmentSets()` to find sufficient adjustment sets.

16.35.2 Prerequisites

Conceptual knowledge of confounding.

16.35.3 Background

A DAG is a set of variables joined by arrows that encode assumed causal directions. The graph distinguishes three kinds of third variable on a path between exposure and outcome: a *confounder* is a common cause; a *mediator* lies on the causal path from exposure to outcome; a *collider* is a common effect of two variables on the path. The practical upshot is that adjusting for a confounder reduces bias, adjusting for a mediator removes part of the effect you are trying to estimate, and adjusting for a collider *creates* bias where none existed.

`dagitty` and `ggdag` let you declare a DAG in text, visualise it, and then query it for adjustment sets — the minimal variable sets that block all back-door paths from exposure to outcome without opening new ones. The right adjustment set depends on the question. For the total effect of X on Y, you want to close all

back-door paths but leave mediators alone; for the direct effect, you also block mediators.

M-bias is the classic example of collider adjustment: conditioning on a variable that is a common effect of an unmeasured cause of X and an unmeasured cause of Y opens a path and biases the estimate. This is why "adjust for everything" is bad advice.

16.35.4 Setup

```
library(tidyverse)
library(dagitty)
library(ggdag)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.35.5 1. Goal

Write a confounder, mediator, and collider scenario as DAGs; ask `dagitty` which variables to adjust for.

16.35.6 2. Approach

```
dag1 <- dagitty('dag {
  X -> Y
  C -> X
  C -> Y
  X [exposure]
  Y [outcome]
}')

dag2 <- dagitty('dag {
  X -> M -> Y
  X -> Y
  X [exposure]
  Y [outcome]
}')

dag3 <- dagitty('dag {
  X -> Z
  Y -> Z
  X -> Y
  X [exposure]
  Y [outcome]
}')
```

```
ggdag(dag1) + theme_dag()
ggdag(dag2) + theme_dag()
ggdag(dag3) + theme_dag()
```

16.35.7 3. Execution

```
adjustmentSets(dag1, exposure = "X", outcome = "Y",
               effect = "total")
```

```
# Total vs direct effect when a mediator exists
adjustmentSets(dag2, exposure = "X", outcome = "Y",
               effect = "total")
adjustmentSets(dag2, exposure = "X", outcome = "Y",
               effect = "direct")
```

```
# Collider: do not adjust for Z
adjustmentSets(dag3, exposure = "X", outcome = "Y")
```

16.35.8 4. Check

Simulate dag1 and verify that adjusting for C recovers the true effect while failing to adjust biases it.

```
n  <- 2000
c_ <- rnorm(n)
x  <- 0.6 * c_ + rnorm(n)
y  <- 0.4 * x + 0.7 * c_ + rnorm(n)
df <- tibble(c_, x, y)

coef(lm(y ~ x,        data = df))["x"]
coef(lm(y ~ x + c_,   data = df))["x"]
```

The unadjusted coefficient is inflated; the adjusted one is close to 0.4 as simulated.

16.35.9 5. Report

> A directed acyclic graph is a compact, testable statement of causal assumptions. Dagitty identified the confounder C as the required adjustment set in the first DAG; the collider Z in the third DAG is explicitly *not* in any adjustment set, and conditioning on it would introduce selection bias.

16.35.10 Common pitfalls

- Listing "every variable we measured" as the adjustment set.
- Adjusting for a mediator and calling the result a total effect.
- Using automated variable selection on observational data without a DAG.
- Drawing the DAG after the analysis.

16.35.11 Further reading

- Textor J et al. (2016), *Robust causal inference using directed acyclic graphs: the R package 'dagitty'*.
- Hernán MA, Robins JM. *Causal Inference: What If*.
- Greenland S, Pearl J, Robins JM (1999), *Causal diagrams for epidemiologic research*.

16.35.12 Session info

```
sessionInfo()
```

16.35.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.36 Week 3, Session 5 — G-methods, IV, DiD, RDD; heterogeneity of treatment effect

Workflow lab: Goal → Approach → Execution → Check → Report.

16.36.1 Learning objectives

- Use inverse-probability weighting as a g-method for time-varying confounding.
- Recognise the data-generating situations that justify an instrumental-variable (IV), difference-in-differences (DiD), or regression-discontinuity (RDD) design.
- Estimate and interpret heterogeneity of treatment effect (HTE) with a subgroup-agnostic method.

16.36.2 Prerequisites

Propensity-score and IPTW basics (W3 S4).

16.36.3 Background

Traditional regression adjustment fails for time-varying confounders that are affected by prior treatment — adjusting for them blocks part of the treatment effect, failing to adjust leaves residual confounding. Robins's g-methods solve this by reweighting or standardising in a way that respects the temporal order. IV methods exploit a variable that affects treatment but not outcome directly; DiD compares changes over time between groups sharing a counterfactual trend; RDD exploits a sharp assignment rule around a threshold.

Heterogeneity of treatment effect is the shift from "is the average effect non-zero?" to "for whom is it largest?" Model-based HTE uses causal forests, meta-learners (S-, T-, X-learner), or Bayesian hierarchical models to estimate conditional average treatment effects (CATEs) while controlling for multiple testing.

16.36.4 Setup

```
library(tidyverse)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.36.5 1. Goal

Simulate a time-varying-confounding scenario, estimate the causal effect naively and then with IPTW (a marginal structural model), and finish with a simple HTE estimate.

16.36.6 2. Approach

```
n <- 2000
df <- tibble(
  l0 = rnorm(n),
  a0 = rbinom(n, 1, plogis(0.2 * l0)),
  l1 = rnorm(n, mean = 0.5 * a0 + 0.3 * l0),
  a1 = rbinom(n, 1, plogis(0.3 * l1 + 0.2 * a0)),
  y  = 1.0 * a0 + 0.8 * a1 + 0.5 * l0 + 0.5 * l1 + rnorm(n)
)
```

The true effect of each treatment is positive and additive. `l1` is a time-varying confounder because it is affected by `a0` and predicts both `a1` and `y`.

16.36.7 3. Execution

Naive regression adjusting for `l1` blocks part of the `a0` effect.

```
naive <- lm(y ~ a0 + a1 + l0 + l1, data = df)
tidy(naive, conf.int = TRUE) |>
  filter(term %in% c("a0", "a1"))
```

IPTW with stabilised weights:

```
num0 <- glm(a0 ~ 1, data = df, family = binomial)
den0 <- glm(a0 ~ l0, data = df, family = binomial)
num1 <- glm(a1 ~ a0, data = df, family = binomial)
den1 <- glm(a1 ~ a0 + l0 + l1, data = df, family = binomial)

w0 <- ifelse(df$a0 == 1, fitted(num0), 1 - fitted(num0)) /
      ifelse(df$a0 == 1, fitted(den0), 1 - fitted(den0))
w1 <- ifelse(df$a1 == 1, fitted(num1), 1 - fitted(num1)) /
      ifelse(df$a1 == 1, fitted(den1), 1 - fitted(den1))
df$sw <- w0 * w1

msm <- lm(y ~ a0 + a1, data = df, weights = sw)
tidy(msm, conf.int = TRUE) |>
  filter(term %in% c("a0", "a1"))
```

16.36.8 4. Check

Balance and weight sanity:

```
summary(df$sw)
```

Stabilised weights should centre on 1 with no extreme tail. If they do not, a term in the treatment model is likely misspecified.

16.36.9 5. Report

> In a simulated cohort with time-varying confounding ($n = 2000$), naive regression underestimated the effect of initial treatment. An inverse-probability-weighted marginal structural model recovered effects close to the true data-generating values. Instrumental variable, difference-in-differences, and regression-discontinuity designs exploit alternative identification strategies, each with its own assumptions. Heterogeneity of treatment effect can be explored with causal forests or meta-learners.

16.36.9.1 Brief sketches

- **IV:** regress `y` on an instrument `z` to recover β`=cov(y,z)/cov(a,z)`.
- **DiD:** fit `lm(y~time*group)` on repeated measures.
- **RDD:** fit local linear regression on each side of the assignment cutoff.

16.36.10 Common pitfalls

- Using ordinary regression adjustment for time-varying confounders.
- Trimming extreme weights silently, hiding model misspecification.
- Reporting subgroup p-values without multiplicity control as HTE evidence.

16.36.11 Further reading

- Hernán MA, Robins JM. *Causal Inference: What If.*
- Athey S, Imbens GW. (2019). *Machine Learning Methods That Economists Should Know About.*

16.36.12 Session info

```
sessionInfo()
```

16.36.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.37 Week 2, Session 4 — GLMMs and GEE

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.37.1 Learning objectives

- Fit a binary GLMM for clustered data with `lme4::glmer` and with `glmmTMB`.
- Fit a GEE with `geepack::geeglm` and compare coefficient interpretations.
- Say why GLMM and GEE estimate different things (subject-specific vs population-average).

16.37.2 Prerequisites

Linear mixed models and logistic regression.

16.37.3 Background

For binary outcomes in clustered data, the two standard approaches are generalised linear mixed models (GLMM) and generalised estimating equations (GEE). A GLMM is a random-effects model on the link scale; its fixed-effect coefficients are conditional on the random effect and describe subject-specific

effects. A GEE fits a marginal model using a working correlation structure; its coefficients describe population-averaged effects. For logistic models, the two will not in general agree even when both are correctly specified.

`lme4::glmer` is the classic random-effects fitter; `glmmTMB` handles larger and more complex random structures including correlated random effects and zero-inflated distributions. `geepack` fits GEEs with several working correlation choices; the standard errors are robust to misspecification of the correlation structure but the point estimate can shift.

The choice between GLMM and GEE is not a statistical technicality; it is a choice of estimand. If you care about what happens inside a person over time, you want a GLMM. If you care about a population- average effect for policy or for a marginal treatment effect, you want a GEE (or a marginalised GLMM).

16.37.4 Setup

```
library(tidyverse)
library(lme4)
library(glmmTMB)
library(geepack)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.37.5 1. Hypothesis

Treatment reduces the probability of a repeated binary event in a clustered study (clusters = clinics). The true log-odds fixed effect is -0.5.

16.37.6 2. Visualise

```
J  <- 30
nj <- 20
dat <- tibble(
  clinic = factor(rep(seq_len(J), each = nj)),
  trt    = rbinom(J * nj, 1, 0.5),
  u      = rep(rnorm(J, 0, 0.8), each = nj)
) |>
  mutate(lp = -0.5 * trt + u,
         y  = rbinom(n(), 1, plogis(lp)))

dat |>
  group_by(clinic, trt) |>
  summarise(p = mean(y), .groups = "drop") |>
  ggplot(aes(factor(trt), p)) +
  geom_boxplot(alpha = 0.6) +
```

```
  labs(x = "Treatment", y = "Event probability per clinic")
```

16.37.7 3. Assumptions

GLMM: normal random intercepts for clinic, correct link. GEE: correct mean structure; standard errors robust to working correlation.

16.37.8 4. Conduct

```
fit_glmer <- glmer(y ~ trt + (1 | clinic),
                   data = dat, family = binomial)
fit_tmb   <- glmmTMB(y ~ trt + (1 | clinic),
                     data = dat, family = binomial)
fit_gee   <- geeglm(y ~ trt, id = clinic, data = dat,
                    family = binomial, corstr = "exchangeable")

summary(fit_glmer)$coefficients
summary(fit_tmb)$coefficients$cond
summary(fit_gee)$coefficients
```

```
bind_rows(
  glmer   = broom.mixed::tidy(fit_glmer, effects = "fixed"),
  glmmTMB = broom.mixed::tidy(fit_tmb,   effects = "fixed"),
  gee     = broom::tidy(fit_gee),
  .id = "model"
) |>
  filter(term == "trt") |>
  select(model, estimate, std.error)
```

16.37.9 5. Concluding statement

> The GLMM gave a subject-specific treatment log-odds of about `round(fixef(fit_glmer)["trt"],2)`; the GEE gave a population- averaged log-odds of about `round(coef(fit_gee)["trt"],2)`. The GEE estimate is typically attenuated toward zero relative to the GLMM; both are correct answers to different questions.

16.37.10 Common pitfalls

- Comparing GLMM and GEE coefficients as if they estimate the same thing.
- Using a working independence structure in GEE when clusters are large and treatment is heterogeneous within them.
- Ignoring overdispersion in a count GLMM by assuming Poisson when negative binomial fits better.

- Fitting a GLMM with too few clusters (< 10) and trusting the variance component.

16.37.11 Further reading

- Liang KY, Zeger SL (1986), *Longitudinal data analysis using generalized linear models*.
- Zeger SL, Liang KY, Albert PS (1988), *Models for longitudinal data: a generalized estimating equation approach*.
- Brooks ME et al. (2017), *glmmTMB balances speed and flexibility*.

16.37.12 Session info

```
sessionInfo()
```

16.37.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.38 Week 2, Session 3 — Linear mixed models with lme4

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.38.1 Learning objectives

- Fit random-intercept and random-slope linear mixed models to longitudinal data.
- Test whether adding a random slope improves fit (likelihood ratio test with ML).
- Interpret REML estimates of variance components.

16.38.2 Prerequisites

Linear regression; familiarity with `lm()`.

16.38.3 Background

Longitudinal data have two sources of variation: differences between people and differences over time within a person. A random-intercept model lets each person have their own average level; a random-slope model lets each person also have their own rate of change. When the slopes genuinely vary, omitting the

random slope underestimates the standard error of the fixed-effect slope and produces p-values that are too small.

REML (restricted maximum likelihood) is the default because it gives unbiased estimates of the variance components by integrating out the fixed effects. It is the right choice for reporting variance components. For comparing models that differ in fixed effects, you must refit with ML (likelihood ratio tests on REML fits are invalid); for comparing models that differ only in random effects, REML is fine.

A common misstep is to interpret the variance of the random intercept as the "within-subject variance". It is the between- subject variance of the intercept. The within-subject variance is the residual variance, σ^2, once the random effects are accounted for.

16.38.4 Setup

```
library(tidyverse)
library(lme4)
library(lmerTest)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.38.5 1. Hypothesis

In the `sleepstudy` data, reaction time increases with days of sleep deprivation, and the *rate* of increase varies by subject.

16.38.6 2. Visualise

```
data(sleepstudy, package = "lme4")

sleepstudy |>
  ggplot(aes(Days, Reaction, group = Subject)) +
  geom_line(alpha = 0.3) +
  geom_smooth(aes(group = 1), method = "lm", se = FALSE,
              colour = "firebrick") +
  labs(x = "Days of deprivation", y = "Reaction time (ms)")
```

16.38.7 3. Assumptions

Linear relationship between Days and Reaction for each subject; normally distributed random intercepts and slopes (if included); residuals independent within and across subjects after random effects account for clustering; approximately constant residual variance.

16.38.8 4. Conduct

```
ri <- lmer(Reaction ~ Days + (1 | Subject),
           data = sleepstudy, REML = TRUE)
rs <- lmer(Reaction ~ Days + (Days | Subject),
           data = sleepstudy, REML = TRUE)
summary(rs)
```

```
# LRT on REML OK here because only random-effects structure differs,
# but we use REFIT = TRUE in anova() to get correct comparison.
anova(ri, rs, refit = FALSE)
```

```
# Fit with ML if the fixed effects changed
fm_ml <- lmer(Reaction ~ Days + (Days | Subject),
              data = sleepstudy, REML = FALSE)
fm_ml
```

```
sleepstudy |>
  mutate(fit_rs = predict(rs)) |>
  ggplot(aes(Days, Reaction, group = Subject)) +
  geom_point(alpha = 0.4) +
  geom_line(aes(y = fit_rs), colour = "steelblue") +
  facet_wrap(~ Subject, ncol = 6) +
  theme(strip.text = element_text(size = 7))
```

16.38.9 5. Concluding statement

> In the sleepstudy data, reaction time increased by about `round(fixef(rs)["Days"],1)` ms per day of deprivation, with substantial subject-to-subject variation in slope (SD $\approx$ `round(sqrt(VarCorr(rs)$Subject[2,2]),1)` ms/day). Adding a random slope significantly improved fit over a random-intercept model (likelihood-ratio test $p < 0.001$).

16.38.10 Common pitfalls

- Testing random-effects with REML-based LRTs without refitting.
- Ignoring the random slope when slopes clearly vary.
- Treating the `Intercept` variance as the within-subject error.
- Using p-values from the Wald z on small samples.

16.38.11 Further reading

- Bates D et al. (2015), *Fitting linear mixed-effects models using lme4*.
- Pinheiro JC, Bates DM. *Mixed-Effects Models in S and S-PLUS*.

- Gelman A, Hill J. *Data Analysis Using Regression and Multilevel/Hierarchical Models*.

16.38.12 Session info

```
sessionInfo()
```

16.38.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.39 Week 2, Session 1 — MCAR, MAR, MNAR

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.39.1 Learning objectives

- Define MCAR, MAR, and MNAR and give an example of each.
- Simulate the three mechanisms and measure the bias in a complete-case analysis.
- Decide when a missingness assumption is defensible from the data alone and when it is not.

16.39.2 Prerequisites

Regression basics.

16.39.3 Background

The language of missingness is Rubin's taxonomy. MCAR (missing completely at random) means the probability of being missing does not depend on anything — observed or unobserved. MAR (missing at random) means the probability depends only on observed variables. MNAR (missing not at random) means the probability depends on the missing value itself, even after conditioning on everything observed. Each is a property of the process, not the data, and you can rarely decide between MAR and MNAR from the data alone.

Complete-case analysis drops any row with a missing value. It is unbiased under MCAR, usually biased under MAR, and always suspect under MNAR. Multiple imputation handles MAR correctly if the imputation model is compatible with the analysis model, but cannot rescue MNAR without explicit modelling of the missingness mechanism.

Auxiliary variables — things that correlate with both the outcome and the missingness but are not in the analysis model — are the currency of MAR. If you have them, put them in the imputation model. If you do not, your MAR claim is on thinner ice than it appears.

16.39.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.39.5 1. Hypothesis

We estimate the mean of Y from each of three datasets with the same true mean, where Y has been made missing by MCAR, MAR, and MNAR processes.

16.39.6 2. Visualise

```
n  <- 1000
df <- tibble(
  x = rnorm(n, 50, 10),
  y = 2 + 0.5 * x + rnorm(n, 0, 5)
)

mcar <- df |> mutate(y_obs = if_else(runif(n) < 0.30, NA_real_, y),
                     mech  = "MCAR")
mar  <- df |> mutate(p = plogis(-2 + 0.05 * (x - 50)),
                     y_obs = if_else(runif(n) < p, NA_real_, y),
                     mech  = "MAR")
mnar <- df |> mutate(p = plogis(-2 + 0.05 * (y - mean(y))),
                     y_obs = if_else(runif(n) < p, NA_real_, y),
                     mech  = "MNAR")

all <- bind_rows(
  mcar |> select(mech, x, y, y_obs),
  mar  |> select(mech, x, y, y_obs),
  mnar |> select(mech, x, y, y_obs)
)

all |>
  mutate(missing = is.na(y_obs)) |>
  ggplot(aes(x, y, colour = missing)) +
  geom_point(alpha = 0.5) +
  facet_wrap(~ mech) +
  labs(colour = "Y missing?")
```

16.39.7 3. Assumptions

Complete-case analysis is the bluntest possible method. It is justified if missingness is MCAR. We will compute the mean of Y on observed rows for each mechanism and compare with the true mean.

16.39.8 4. Conduct

```
results <- all |>
  group_by(mech) |>
  summarise(
    truth = mean(y),
    cc    = mean(y_obs, na.rm = TRUE),
    bias  = cc - truth,
    pct_missing = mean(is.na(y_obs)) * 100
  )
results
```

```
results |>
  ggplot(aes(mech, bias)) +
  geom_col(alpha = 0.7, fill = "steelblue") +
  geom_hline(yintercept = 0, linetype = "dashed") +
  labs(x = NULL, y = "Bias of complete-case mean")
```

16.39.9 5. Concluding statement

> Complete-case estimation of the mean of Y was unbiased under MCAR (bias ≈ `round(results$bias[results$mech=="MCAR"],2)`) and meaningfully biased under MAR (`round(results$bias[results$mech=="MAR"],2)`) and MNAR (`round(results$bias[results$mech=="MNAR"],2)`), despite the same true mean and similar missingness proportions.

16.39.10 Common pitfalls

- Using Little's test as evidence *for* MCAR.
- Imputing with a model that omits the analysis outcome.
- Reporting a sample size inflated by a complete-case filter that hides 40% dropout.
- Using "the missingness was not significantly related to X" as proof of MAR.

16.39.11 Further reading

- Little RJA, Rubin DB. *Statistical Analysis with Missing Data.*
- van Buuren S (2018), *Flexible Imputation of Missing Data.*

- Sterne JA et al. (2009), *Multiple imputation for missing data in epidemiological and clinical research.*

16.39.12 Session info

```
sessionInfo()
```

16.39.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.40 Week 2, Session 2 — Multiple imputation with mice

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.40.1 Learning objectives

- Run multiple imputation by chained equations with `mice`.
- Diagnose convergence with trace and density plots.
- Pool regression coefficients across imputations using Rubin's rules.

16.40.2 Prerequisites

The MCAR/MAR/MNAR lab.

16.40.3 Background

Multiple imputation (MI) replaces each missing value with several plausible values drawn from a predictive distribution, producing m completed datasets. Each completed dataset is analysed separately; the results are pooled using Rubin's rules so that the standard errors reflect both the within-imputation uncertainty (ordinary sampling variance) and the between-imputation uncertainty (the variance of the imputations themselves).

`mice` implements MI by chained equations: for each incomplete variable, it fits a conditional model given the others and draws imputations from the posterior predictive distribution of that model. The procedure iterates until the imputations stabilise. The standard diagnostics are trace plots (should mix and not drift) and density plots comparing observed and imputed values (should overlap but need not match exactly).

The inclusion of the outcome in the imputation model is not optional. Omitting it biases the imputations toward the null and attenuates the estimated effect in the pooled analysis.

16.40.4 Setup

```
library(tidyverse)
library(mice)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.40.5 1. Hypothesis

In the `mice::nhanes` teaching dataset, the cholesterol outcome (`chl`) is related to age and BMI (`bmi`). We will estimate that relationship under MI.

16.40.6 2. Visualise

```
data(nhanes, package = "mice")
md.pattern(nhanes, plot = FALSE)
```

```
nhanes |>
  mutate(missing_chl = is.na(chl)) |>
  ggplot(aes(bmi, chl, colour = missing_chl)) +
  geom_point(size = 2) +
  labs(x = "BMI", y = "Cholesterol", colour = "chl missing?")
```

16.40.7 3. Assumptions

MAR conditional on the variables in the imputation model; a compatible imputation model (predictive mean matching by default); enough imputations (here $m = 10$) to stabilise the pooled standard errors.

16.40.8 4. Conduct

```
imp <- mice(nhanes, m = 10, method = "pmm",
            printFlag = FALSE, seed = 42)
imp
```

```
plot(imp)
```

```
densityplot(imp)
```

```
fit   <- with(imp, lm(chl ~ age + bmi))
pooled <- pool(fit)
summary(pooled, conf.int = TRUE)
```

16.40.9 5. Concluding statement

> Multiple imputation of the `nhanes` dataset ($m = 10$, predictive mean matching) gave a pooled coefficient for BMI on cholesterol of `round(summary(pooled)$estimate[summary(pooled)$term=="bmi"],2)` (see pooled table above), with standard errors reflecting both within- and between-imputation uncertainty through Rubin's rules.

16.40.10 Common pitfalls

- Imputing the outcome separately from the covariates with incompatible models.
- Using $m = 5$ when the fraction of missing information is high.
- Pooling means or SDs by hand instead of `pool()`.
- Dropping the outcome from the imputation model because it "feels like cheating".

16.40.11 Further reading

- van Buuren S, Groothuis-Oudshoorn K (2011), *mice: Multivariate Imputation by Chained Equations in R*.
- van Buuren S (2018), *Flexible Imputation of Missing Data*, ch. 4–6.
- White IR, Royston P, Wood AM (2011), *Multiple imputation using chained equations: issues and guidance for practice*.

16.40.12 Session info

```
sessionInfo()
```

16.40.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.41 Week 1, Session 1 — Observational designs and STROBE

Workflow lab using the variant template: Goal → Approach → Execution → Check → Report.

16.41.1 Learning objectives

- Distinguish cohort, case-control, and cross-sectional designs and say when each is the right tool.
- Simulate a source population and draw both a cohort sample and a nested case-control sample from it.
- Map a study report to the STROBE checklist.

16.41.2 Prerequisites

Inferential statistics and regression. Familiarity with tibbles and `glm()`.

16.41.3 Background

Observational designs differ primarily in how participants enter the study and when the outcome is measured relative to the exposure. In a cohort we sample on exposure and wait; in a case-control we sample on the outcome and look back; in a cross-sectional we sample once and measure everything together. The choice is driven by how common the outcome is, how long it takes to develop, and how easy the exposure is to measure retrospectively.

STROBE (Strengthening the Reporting of Observational Studies in Epidemiology) is the reporting checklist that keeps observational papers honest. It has 22 items covering title and abstract, introduction, methods, results, and discussion, with specific branches for the three main designs. You do not need to memorise it; you need to know that reviewers will check it.

A nested case-control sampled from a defined cohort behaves statistically like the full cohort for odds ratios that approximate risk ratios when the outcome is rare. This is the trick behind most large pharmacoepidemiology studies: a full-cohort analysis is wasteful when measurement on every unexposed person is expensive, so analysts sample controls from the risk set instead.

16.41.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.41.5 1. Goal

Simulate a source population, carve a cohort study and a nested case-control study out of it, and compare the exposure–outcome association recovered by each.

16.41.6 2. Approach

We generate a population of 5000 people with a binary exposure and a binary outcome whose probability depends on exposure and a confounder (age). We then (a) analyse the full cohort and (b) draw a case-control sample with four controls per case.

```
n_pop <- 5000
pop <- tibble(
  id       = seq_len(n_pop),
  age      = rnorm(n_pop, 55, 10),
  exposure = rbinom(n_pop, 1, 0.30)
)
# outcome depends on exposure (log OR ~ 0.8) and age
lp <- -4 + 0.8 * pop$exposure + 0.04 * (pop$age - 55)
pop <- pop |>
  mutate(prob = plogis(lp),
         outcome = rbinom(n_pop, 1, prob))
```

```
pop |>
  count(exposure, outcome) |>
  mutate(exposure = factor(exposure, labels = c("unexposed", "exposed")),
         outcome  = factor(outcome,  labels = c("no", "yes"))) |>
  ggplot(aes(exposure, n, fill = outcome)) +
  geom_col(position = "dodge", alpha = 0.8) +
  labs(x = NULL, y = "Count", fill = "Outcome")
```

16.41.7 3. Execution

```
# Cohort analysis on full population
fit_cohort <- glm(outcome ~ exposure + age, data = pop, family = binomial)
coef_cohort <- broom::tidy(fit_cohort, conf.int = TRUE, exponentiate =
↪  TRUE)
coef_cohort
```

```
# Nested case-control: all cases, 4 controls per case
cases    <- pop |> filter(outcome == 1)
controls <- pop |>
  filter(outcome == 0) |>
  slice_sample(n = 4 * nrow(cases))
ncc <- bind_rows(cases, controls)
fit_ncc <- glm(outcome ~ exposure + age, data = ncc, family = binomial)
coef_ncc <- broom::tidy(fit_ncc, conf.int = TRUE, exponentiate = TRUE)
coef_ncc
```

16.41.8 4. Check

The exposure odds ratio from the nested case-control should be close to that from the cohort, because the outcome is uncommon. Precision is worse in the case-control sample but the point estimate is not biased by the sampling scheme.

```
bind_rows(
  cohort = coef_cohort |> filter(term == "exposure"),
  ncc    = coef_ncc    |> filter(term == "exposure"),
  .id = "design"
) |>
  select(design, estimate, conf.low, conf.high)
```

16.41.9 5. Report

> In a simulated source population of 5000 adults, exposure was associated with the outcome with an adjusted odds ratio of `round(coef_cohort$estimate[2],2)` (95% CI: `round(coef_cohort$conf.low[2],2)` to `round(coef_cohort$conf.high[2],2)`) in a full-cohort analysis, and `round(coef_ncc$estimate[2],2)` (95% CI: `round(coef_ncc$conf.low[2],2)` to `round(coef_ncc$conf.high[2],2)`) in a 1:4 nested case-control sample.

16.41.9.1 STROBE in one paragraph

Report the sampling frame, the eligibility criteria, the exposure and outcome definitions, how confounders were chosen and handled, whatever you did about missing data, and a flow diagram of who was included, excluded, and lost. If you can do those six things clearly, you have answered most of the STROBE items for free.

16.41.10 Common pitfalls

- Treating a cross-sectional association as evidence of a causal effect when temporality is unknown.
- Reporting an odds ratio as a risk ratio when the outcome is common.
- Drawing controls from a different population than the cases.
- Skipping a flow diagram in an observational paper.

16.41.11 Further reading

- von Elm E et al. (2007), *The STROBE Statement*.
- Rothman KJ, Greenland S, Lash TL. *Modern Epidemiology*, ch. 6–8.
- Pearce N (2016), *Analysis of matched case-control studies*, BMJ.

16.41.12 Session info

```
sessionInfo()
```

16.41.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.42 Week 4, Session 5 — Pre-registration and statistical analysis plans

Workflow lab: Goal → Approach → Execution → Check → Report.

16.42.1 Learning objectives

- Distinguish a pre-registration from a statistical analysis plan.
- Draft the core sections of each for a realistic study.
- Recognise the forms of post-hoc flexibility that pre-registration is designed to prevent.

16.42.2 Prerequisites

Earlier design and causal-inference material.

16.42.3 Background

A pre-registration is a timestamped public record of a study's hypotheses, design, and primary analysis. A statistical analysis plan (SAP) is a more detailed, often confidential companion document that specifies exactly how the data will be handled — variable definitions, handling of missing data, subgroup analyses, sensitivity analyses, and reporting conventions. Between them, these two documents make the difference between "we planned this" and "we planned this, we can prove it, and here is the file to show when it was signed."

Flexibility during analysis — *researcher degrees of freedom* — is not fraud. It is usually well-meaning curiosity. But aggregated across a field it produces a literature of spurious findings, and for any one study it produces an analysis that will not replicate. A pre-registration does not forbid curiosity. It separates *confirmatory* analyses (specified in advance) from *exploratory* ones, and requires each to be labelled when reported.

16.42.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.42.5 1. Goal

Produce a minimal-but-complete pre-registration template for a fictional two-arm randomised trial, plus a SAP skeleton.

16.42.6 2. Approach

A pre-registration has four mandatory elements:

1. **Question and hypotheses** — exact form of H_0 and H_1 for each primary outcome.
2. **Design** — eligibility, allocation, blinding, intervention, follow-up.
3. **Primary analysis** — the single specification whose p-value will be quoted as the headline result.
4. **Sample size justification** — the calculation behind the target N.

A SAP adds:

5. **Secondary analyses** — ordered and pre-specified.
6. **Handling of missing data** — mechanism assumed, method used.
7. **Subgroup analyses** — pre-specified, limited in number.
8. **Sensitivity analyses** — especially for primary outcomes.

16.42.7 3. Execution — template

```
# Pre-registration — Trial X

Protocol version 1.0 | Date 2026-04-18 | PI [NAME]

## 1. Research question
Primary: Does [intervention] reduce [outcome] relative to [control]
in [population] over [time frame]?

## 2. Hypotheses
H0: difference in [outcome] = 0.
H1: difference in [outcome] != 0. Two-sided, alpha = 0.05.

## 3. Design
Two-arm, parallel-group, double-blind, placebo-controlled trial.
1:1 allocation, block randomisation (block size 4), stratified by site.
```

```
## 4. Primary analysis
ITT. Linear regression of outcome at follow-up on arm, adjusted for
baseline value and stratification factors. Primary estimand:
adjusted mean difference with 95% CI.

## 5. Sample size
Target n = 200 per arm; power = 0.80 for d = 0.3, alpha = 0.05.

## 6. Data handling
- Missing data: multiple imputation under MAR (m = 20).
- Outliers: retained in primary analysis.
- Adherence: per-protocol analysis as sensitivity.
```

16.42.8 4. Check

Three questions to ask of any pre-registration before posting:

1. Could a reader reconstruct the primary analysis from the text without contacting you?
2. Are exploratory analyses labelled as such?
3. Is the target N justified by a calculation a reader can redo?

16.42.9 5. Report

> A pre-registration for Trial X was deposited on the OSF on [date] (DOI [doi]). The primary analysis, statistical model, and sample-size justification are specified therein. Any deviation from the pre-registered plan is reported in the Deviations section of the manuscript with its rationale.

16.42.10 Common pitfalls

- Treating pre-registration as a checkbox without a concrete analysis plan.
- Pre-registering two primary outcomes and quoting the winner.
- Failing to report deviations from plan.

16.42.11 Further reading

- Nosek BA et al. (2018). *The preregistration revolution.*
- AMRC / NIHR Statistical Analysis Plan templates.

16.42.12 Session info

```
sessionInfo()
```

16.42.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.43 Week 3, Session 4 — Propensity scores and IPTW

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.43.1 Learning objectives

- Estimate a propensity score and match nearest-neighbour with `MatchIt`.
- Construct inverse-probability-of-treatment weights (IPTW) and fit a weighted outcome model.
- Diagnose covariate balance with `cobalt::love.plot`.

16.43.2 Prerequisites

DAGs and logistic regression.

16.43.3 Background

The propensity score is the probability of being treated given covariates. Under the assumptions of conditional exchangeability and positivity, conditioning on the propensity score yields an unbiased estimate of the average treatment effect. Matching and inverse-probability-of-treatment weighting are two ways to do that conditioning.

Matching pairs each treated unit with one (or more) untreated units with similar propensity scores, then analyses the matched sample. IPTW gives each unit a weight of $1/e(X)$ if treated and $1/(1 - e(X))$ if untreated, creating a pseudo-population where treatment is independent of X. Both estimators require the propensity model to be correctly specified; neither handles unmeasured confounding.

Balance diagnostics assess whether the matched or weighted sample has covariate distributions that are similar across treatment groups. The standardised mean difference (SMD) is the workhorse metric; anything below 0.1 is usually considered balanced.

Large weights (extreme propensities near 0 or 1) are a sign of poor positivity. Stabilised weights and truncation can help, but the underlying problem — a region of covariate space where one treatment is never seen — cannot be fixed statistically.

16.43.4 Setup

```
library(tidyverse)
library(MatchIt)
library(cobalt)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.43.5 1. Hypothesis

Treatment reduces the outcome by a known amount. We will estimate the effect with naive regression, matching, and IPTW, and compare with the simulated truth.

16.43.6 2. Visualise

```
n <- 1000
dat <- tibble(
  age  = rnorm(n, 60, 10),
  sev  = rnorm(n, 0,   1),
  sex  = rbinom(n, 1, 0.5)
) |>
  mutate(ps = plogis(-1 + 0.04 * (age - 60) + 0.8 * sev + 0.3 * sex),
         trt = rbinom(n, 1, ps),
         y   = 2 - 1.5 * trt + 0.05 * (age - 60) +
                 0.8 * sev + 0.3 * sex + rnorm(n, 0, 1))

ggplot(dat, aes(ps, fill = factor(trt))) +
  geom_density(alpha = 0.5) +
  labs(x = "Propensity score", y = "Density", fill = "Treated?")
```

16.43.7 3. Assumptions

No unmeasured confounding (conditional exchangeability); positivity (everyone has a non-zero chance of each treatment); correct specification of the propensity model.

16.43.8 4. Conduct

```
# Naive regression
fit_naive <- lm(y ~ trt + age + sev + sex, data = dat)
coef(fit_naive)["trt"]
```

```r
# Nearest-neighbour matching
m <- matchit(trt ~ age + sev + sex, data = dat,
             method = "nearest", ratio = 1)
m
matched <- match.data(m)

fit_match <- lm(y ~ trt, data = matched,
                weights = matched$weights)
coef(fit_match)["trt"]
```

```r
# IPTW
dat <- dat |>
  mutate(w = if_else(trt == 1, 1 / ps, 1 / (1 - ps)))
fit_iptw <- lm(y ~ trt, data = dat, weights = w)
coef(fit_iptw)["trt"]
```

```r
love.plot(m, thresholds = c(m = 0.1), abs = TRUE)
```

16.43.9 5. Concluding statement

> The simulated treatment effect was -1.5. The naive regression gave `round(coef(fit_naive)["trt"],2)`; nearest-neighbour matching gave `round(coef(fit_match)["trt"],2)`; IPTW gave `round(coef(fit_iptw)["trt"],2)`. Balance plots showed that matching reduced all SMDs below 0.1.

16.43.10 Common pitfalls

- Matching with replacement without adjusting inference for the repeated use of controls.
- Reporting only the p-value for the treatment effect in the weighted model without a sandwich or bootstrap standard error.
- Ignoring extreme weights.
- Picking the matching method that gives the effect you wanted.

16.43.11 Further reading

- Austin PC (2011), *An introduction to propensity score methods for reducing the effects of confounding in observational studies*.
- Stuart EA (2010), *Matching methods for causal inference*.
- Hernán MA, Robins JM. *Causal Inference: What If*, ch. 12.

16.43.12 Session info

```
sessionInfo()
```

16.43.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.44 Week 1, Session 2 — Randomised controlled trials

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

16.44.1 Learning objectives

- Contrast parallel-group, crossover, cluster, and factorial RCT designs.
- Simulate a parallel-group RCT with a realistic allocation procedure.
- Interpret intention-to-treat (ITT) and per-protocol (PP) analyses and say why they can differ.

16.44.2 Prerequisites

Inferential statistics; `t.test()`, `lm()`, and basic tidyverse.

16.44.3 Background

The randomised controlled trial (RCT) is the reference design for causal inference because randomisation balances both measured and unmeasured confounders in expectation. Parallel-group RCTs assign each participant to one arm for the duration of the study. Crossover trials give each participant every treatment in a randomised order and rely on a washout period. Cluster trials randomise groups (clinics, villages) rather than individuals. Factorial trials vary more than one treatment at once and can answer more than one question per patient.

Every RCT has two analyses that will often disagree. Intention-to- treat analyses everyone in the arm they were randomised to, regardless of what they actually received. Per-protocol analyses only those who received the assigned treatment as planned. ITT is conservative for superiority and preserves the randomisation; PP is informative for efficacy in compliant patients but can be biased. Report both, and say which is primary in the protocol.

Allocation concealment — the process that keeps the next assignment unknown until the patient has been enrolled — is not the same as blinding. You can have

one without the other, and you need both for the trial to protect itself from selection bias and differential outcome ascertainment.

16.44.4 Setup

```
library(tidyverse)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.44.5 1. Hypothesis

Treatment reduces the outcome (a continuous symptom score) relative to placebo. We will also see what happens when 15% of participants cross over from treatment to placebo after randomisation.

16.44.6 2. Visualise

```
n <- 200
trial <- tibble(
  id    = seq_len(n),
  arm   = sample(rep(c("placebo", "treatment"), each = n / 2)),
  # 15% of treatment arm never take the drug (will be placebo in reality)
  crossed = if_else(arm == "treatment" & runif(n) < 0.15, TRUE, FALSE),
  received = if_else(crossed, "placebo", arm),
  y = rnorm(n, mean = 50, sd = 8) +
      if_else(received == "treatment", -5, 0)
)

trial |>
  ggplot(aes(arm, y, fill = arm)) +
  geom_boxplot(alpha = 0.6, colour = "grey30") +
  labs(x = NULL, y = "Symptom score") +
  theme(legend.position = "none")
```

16.44.7 3. Assumptions

A two-sample t-test on the ITT population assumes independent observations within arms and roughly normal residuals within each arm. We do not assume equal variances; `t.test()` uses Welch by default. The more important assumption is that randomisation worked — that the allocation is independent of every baseline covariate.

```
trial |>
  group_by(arm) |>
  summarise(mean = mean(y), sd = sd(y), n = n())
```

16.44.8 4. Conduct

```
itt <- t.test(y ~ arm, data = trial)
pp  <- t.test(y ~ received, data = trial)
itt
pp
```

The ITT estimate is closer to zero than the PP estimate because the crossed-over patients pull the treatment arm mean toward placebo.

16.44.9 5. Concluding statement

> In a simulated parallel-group RCT (n = 200), the ITT analysis showed a mean difference of `round(diff(itt$estimate)*-1,1)` (95% CI: `round(-itt$conf.int[2],1)` to `round(-itt$conf.int[1],1)`) on the symptom score; a per-protocol analysis gave a larger estimated benefit of `round(diff(pp$estimate)*-1,1)`, illustrating the usual direction of disagreement when non-adherence is informative.

16.44.10 Common pitfalls

- Calling a non-random allocation "randomised" because a coin was flipped after enrolment.
- Reporting only PP and calling it primary.
- Ignoring stratification variables in the analysis when they were used at randomisation.
- Cluster trials analysed as if individuals were independent.

16.44.11 Further reading

- Schulz KF, Altman DG, Moher D (2010), *CONSORT 2010 Statement*.
- Pocock SJ. *Clinical Trials: A Practical Approach*.
- Hernán MA, Hernández-Díaz S (2012), *Beyond the intention-to-treat in comparative effectiveness research*.

16.44.12 Session info

```
sessionInfo()
```

16.44.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

16.45 Week 4, Session 4 — SIR and SEIR models with deSolve

Workflow lab: Goal → Approach → Execution → Check → Report.

16.45.1 Learning objectives

- Write a compartmental SIR / SEIR model as a system of ODEs.
- Solve it with `deSolve::ode` and plot the trajectories.
- Compute the basic reproduction number R_0 from model parameters and illustrate its effect.

16.45.2 Prerequisites

Basic calculus intuition and comfort with differential-equation notation.

16.45.3 Background

Compartmental models partition a population into states — *Susceptible*, *Exposed*, *Infectious*, *Recovered* — and describe flows between them with ordinary differential equations. The SIR model has three compartments and two parameters (transmission rate β, recovery rate γ); the SEIR model adds an exposed-but-not-yet-infectious class and a progression rate σ. These models are caricatures, but they capture the shape of an epidemic — exponential rise, peak, decline — surprisingly well, and they make the role of $R_0 = \beta/\gamma$ visible.

The real power of compartmental models in practice is not prediction but counterfactual reasoning: *what if* we had vaccinated half the population on day zero, or *what if* the infectious period were a day shorter? Tuning β and γ to reflect interventions is the bread and butter of public-health modelling.

16.45.4 Setup

```
library(tidyverse)
library(deSolve)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

16.45.5 1. Goal

Simulate an SIR outbreak in a population of 1 million, for R_0 values 1.5, 2.5, and 4.

16.45.6 2. Approach

```
sir <- function(time, state, params) {
  with(as.list(c(state, params)), {
    N  <- S + I + R
    dS <- -beta * S * I / N
    dI <-  beta * S * I / N - gamma * I
    dR <-  gamma * I
    list(c(dS, dI, dR))
  })
}
```

16.45.7 3. Execution

```
run_sir <- function(R0, gamma = 1/7, days = 180, N = 1e6, I0 = 10) {
  params <- c(beta = R0 * gamma, gamma = gamma)
  state  <- c(S = N - I0, I = I0, R = 0)
  times  <- seq(0, days, by = 1)
  out <- as.data.frame(ode(state, times, sir, params))
  out$R0 <- R0
  out
}

trajectories <- bind_rows(
  run_sir(1.5), run_sir(2.5), run_sir(4.0)
) |>
  pivot_longer(c(S, I, R), names_to = "compartment", values_to = "n")
```

```
ggplot(trajectories, aes(time, n, colour = compartment)) +
  geom_line(linewidth = 0.8) +
  facet_wrap(~ R0, labeller = labeller(R0 = \(x) paste0("R0 = ", x))) +
  labs(x = "Day", y = "People")
```

16.45.8 4. Check

Peak prevalence and attack rate should rise with R_0.

```
trajectories |>
  filter(compartment == "I") |>
  group_by(R0) |>
  summarise(peak_I = max(n), peak_day = time[which.max(n)]) |>
  arrange(R0)
```

16.45.9 5. Report

> An SIR compartmental model was simulated for a population of one million with a 7-day mean infectious period across three basic reproduction numbers (R_0 = 1.5, 2.5, 4). Peak prevalence and the final attack rate increased sharply with R_0, illustrating how small changes in transmission translate into large changes in epidemic size. Calibrated analogues of this model underpin much real-world outbreak forecasting.

16.45.10 Common pitfalls

- Interpreting SIR trajectories as forecasts rather than scenarios.
- Forgetting that a constant β across time ignores interventions.
- Comparing peak prevalence across parameter settings without holding N and I_0 constant.

16.45.11 Further reading

- Keeling MJ, Rohani P. *Modeling Infectious Diseases in Humans and Animals.*
- Soetaert K et al. *Solving Differential Equations in R.*

16.45.12 Session info

```
sessionInfo()
```

16.45.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

Chapter 17

Reporting Templates

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

Copy-paste Methods + Results paragraphs, organised by reporting guideline. Pages here are deliberately formulaic: fill the bracketed *italics* with your numbers, replace the placeholder package versions with what you actually used, and check the **Reviewer checklist** link for each method to confirm you have reported every diagnostic a reviewer expects to see.

Templates are organised by guideline:

- **STROBE — observational studies** — 186 method templates
- **CONSORT — randomised trials** — 141 method templates
- **TRIPOD-AI — prediction models** — 163 method templates
- **PRISMA — systematic reviews & meta-analyses** — 25 method templates
- **ARRIVE — animal research** — 50 method templates

Every page is currently a **skeleton**: placeholders, no narrative text. The prose is written by hand into the slots marked `<!--TODO:reporting-template-->`.

Chapter 18

STROBE — observational studies

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

STROBE — *Strengthening the Reporting of Observational Studies in Epidemiology* — covers cohort, case-control, and cross-sectional studies. The 22-item checklist is at https://www.strobe-statement.org/. The templates below cover the analytic methods you are most likely to use in a STROBE-aligned paper.

This page contains **186 method-specific templates**. Each is a skeleton with placeholders in italics; the prose is written so you can paste it into a manuscript and replace only the bracketed values.

18.0.1 2^k Factorial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **2^k Factorial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> 2^k Factorial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `factorial-2k` — For Reviewers

18.0.2 3^k Factorial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **3^k Factorial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> 3^k Factorial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `factorial-3k` — For Reviewers

18.0.3 Accelerated Failure Time Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Accelerated Failure Time Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Accelerated Failure Time Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `accelerated-failure-time` — For Reviewers

18.0.4 Added-Variable Plots

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Added-Variable Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Added-Variable Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `added-variable-plots` — For Reviewers

18.0.5 Anderson-Darling Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Anderson-Darling Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Anderson-Darling Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `anderson-darling-test` — For Reviewers

18.0.6 Balanced Incomplete Block Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Balanced Incomplete Block Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Balanced Incomplete Block Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `balanced-incomplete-block` — For Reviewers

18.0.7 Bartlett's Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Bartlett's Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Bartlett's Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bartlett-test` — For Reviewers

18.0.8 Benjamini-Hochberg FDR

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Benjamini-Hochberg FDR** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Benjamini-Hochberg FDR yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `benjamini-hochberg-fdr` — For Reviewers

18.0.9 Benjamini-Yekutieli FDR

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Benjamini-Yekutieli FDR** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Benjamini-Yekutieli FDR yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `benjamini-yekutieli` — For Reviewers

18.0.10 Best Subset Selection

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Best Subset Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Best Subset Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `best-subset-selection` — For Reviewers

18.0.11 Beta Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Beta Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Beta Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `beta-regression` — For Reviewers

18.0.12 Bonferroni Correction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bonferroni Correction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Bonferroni Correction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bonferroni-correction` — For Reviewers

18.0.13 Bootstrap Confidence Intervals

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bootstrap Confidence Intervals** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Bootstrap Confidence Intervals yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bootstrap-confidence-intervals` — For Reviewers

18.0.14 Box-Behnken Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Box-Behnken Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Box-Behnken Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `box-behnken-design` — For Reviewers

18.0.15 Censoring Types

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Censoring Types** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Censoring Types yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `censoring-types` — For Reviewers

18.0.16 Central Composite Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Central Composite Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Central Composite Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `central-composite-design` — For Reviewers

18.0.17 Centring and Scaling Predictors

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Centring and Scaling Predictors** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Centring and Scaling Predictors yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `centered-scaled-predictors` — For Reviewers

18.0.18 Checking Cox Assumptions

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Checking Cox Assumptions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Checking Cox Assumptions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cox-assumptions-check` — For Reviewers

18.0.19 Chi-Squared Goodness-of-Fit Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Chi-Squared Goodness-of-Fit Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Chi-Squared Goodness-of-Fit Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `chi-squared-goodness-of-fit` — For Reviewers

18.0.20 Chi-Squared Test of Independence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Chi-Squared Test of Independence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Chi-Squared Test of Independence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `chi-squared-independence` — For Reviewers

18.0.21 Cochran-Mantel-Haenszel Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cochran-Mantel-Haenszel Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Cochran-Mantel-Haenszel Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cochran-mantel-haenszel` — For Reviewers

18.0.22 Competing Risks: Cumulative Incidence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Competing Risks: Cumulative Incidence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Competing Risks: Cumulative Incidence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `competing-risks-cif` — For Reviewers

18.0.23 Completely Randomised Design

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Completely Randomised Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Completely Randomised Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `completely-randomized-design` — For Reviewers

18.0.24 Concordance and the C-Index

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Concordance and the C-Index** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Concordance and the C-Index yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `concordance-c-index` — For Reviewers

18.0.25 Conditional Survival

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Conditional Survival** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Conditional Survival yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `conditional-survival` — For Reviewers

18.0.26 Constrained Mixture Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Constrained Mixture Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Constrained Mixture Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `constrained-mixture` — For Reviewers

18.0.27 Contrasts in R

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Contrasts in R** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Contrasts in R yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `contrasts-in-r` — For Reviewers

18.0.28 Cook's Distance

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cook's Distance** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Cook's Distance yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cooks-distance` — For Reviewers

18.0.29 Cox Proportional Hazards Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Cox Proportional Hazards Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Cox Proportional Hazards Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cox-proportional-hazards` — For Reviewers

18.0.30 Cox Regression as Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Cox Regression as Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Cox Regression as Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `survival-regression-cox` — For Reviewers

18.0.31 Crossover Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Crossover Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions

were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Crossover Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `crossover-designs` — For Reviewers

18.0.32 Desirability Functions

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Desirability Functions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Desirability Functions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `desirability-function` — For Reviewers

18.0.33 Dirichlet Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Dirichlet Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Dirichlet Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `dirichlet-regression` — For Reviewers

18.0.34 Dummy Coding

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Dummy Coding** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Dummy Coding yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `dummy-coding` — For Reviewers

18.0.35 Effect (Deviation) Coding

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect (Deviation) Coding** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect (Deviation) Coding yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `effect-coding` — For Reviewers

18.0.36 Effect Size: Cohen's d

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Size: Cohen's d** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect Size: Cohen's d yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `effect-size-cohens-d` — For Reviewers

18.0.37 Effect Size: Cohen's h

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Size: Cohen's h** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect Size: Cohen's h yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `effect-size-cohens-h` — For Reviewers

18.0.38 Effect Size: Eta-Squared

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Size: Eta-Squared** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect Size: Eta-Squared yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `effect-size-eta-squared` — For Reviewers

18.0.39 Effect Sizes: Overview

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Sizes: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect Sizes: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `effect-sizes-overview` — For Reviewers

18.0.40 Elastic Net

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Elastic Net** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Elastic Net yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `elastic-net` — For Reviewers

18.0.41 Equivalence Testing with TOST

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Equivalence Testing with TOST** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Equivalence Testing with TOST yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `equivalence-tost` — For Reviewers

18.0.42 Fine-Gray Subdistribution Hazards

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Fine-Gray Subdistribution Hazards** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect*

size with 95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Fine-Gray Subdistribution Hazards yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `fine-gray-regression` — For Reviewers

18.0.43 Fisher's Exact Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Fisher's Exact Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Fisher's Exact Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `fisher-exact-test` — For Reviewers

18.0.44 Fractional Factorial Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Fractional Factorial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Fractional Factorial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `fractional-factorial-design` — For Reviewers

18.0.45 Frailty Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Frailty Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Frailty Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `frailty-models` — For Reviewers

18.0.46 Friedman Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Friedman Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Friedman Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `friedman-test` — For Reviewers

18.0.47 GAMMs: Additive Mixed Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **GAMMs: Additive Mixed Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> GAMMs: Additive Mixed Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `gamm-additive-mixed` — For Reviewers

18.0.48 GAMs: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **GAMs: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> GAMs: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `gam-introduction` — For Reviewers

18.0.49 Generalised Estimating Equations

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Generalised Estimating Equations** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Generalised Estimating Equations yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `generalized-estimating-equations` — For Reviewers

18.0.50 glmer: Generalised Mixed Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **glmer: Generalised Mixed Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> glmer: Generalised Mixed Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `glmer-generalized-mixed` — For Reviewers

18.0.51 Graeco-Latin Square Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Graeco-Latin Square Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Graeco-Latin Square Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `graeco-latin-square` — For Reviewers

18.0.52 Hazard, Survival, and Cumulative Hazard

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Hazard, Survival, and Cumulative Hazard** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Hazard, Survival, and Cumulative Hazard yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `hazard-survival-cumulative` — For Reviewers

18.0.53 Holm's Step-Down Correction

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Holm's Step-Down Correction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Holm's Step-Down Correction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `holm-correction` — For Reviewers

18.0.54 Hurdle Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Hurdle Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Hurdle Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `hurdle-models` — For Reviewers

18.0.55 Interactions in Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Interactions in Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Interactions in Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `interactions-in-regression` — For Reviewers

18.0.56 Interval-Censored Data

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Interval-Censored Data** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Interval-Censored Data yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `interval-censored-data` — For Reviewers

18.0.57 Joint Longitudinal-Survival Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Joint Longitudinal-Survival Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Joint Longitudinal-Survival Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `joint-longitudinal-survival` — For Reviewers

18.0.58 Kaplan-Meier Estimation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Kaplan-Meier Estimation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Kaplan-Meier Estimation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `kaplan-meier` — For Reviewers

18.0.59 Kendall's Tau

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Kendall's Tau** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Kendall's Tau yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `kendall-tau` — For Reviewers

18.0.60 Kolmogorov-Smirnov Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Kolmogorov-Smirnov Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Kolmogorov-Smirnov Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `kolmogorov-smirnov-test` — For Reviewers

18.0.61 Kruskal-Wallis Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Kruskal-Wallis Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].*

Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Kruskal-Wallis Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `kruskal-wallis` — For Reviewers

18.0.62 Landmark Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Landmark Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Landmark Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `landmark-analysis` — For Reviewers

18.0.63 Lasso Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Lasso Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Lasso Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `lasso-regression` — For Reviewers

18.0.64 Latin Square Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Latin Square Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Latin Square Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `latin-square` — For Reviewers

18.0.65 Left Truncation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Left Truncation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Left Truncation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `left-truncation` — For Reviewers

18.0.66 Levene's Test of Variances

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Levene's Test of Variances** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Levene's Test of Variances yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `levene-test` — For Reviewers

18.0.67 Leverage and Influence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Leverage and Influence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Leverage and Influence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `leverage-influence` — For Reviewers

18.0.68 Linear Regression Assumptions

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Linear Regression Assumptions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Linear Regression Assumptions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `regression-assumptions` — For Reviewers

18.0.69 Log-Rank Test: Details

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Log-Rank Test: Details** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Log-Rank Test: Details yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `log-rank-test-details` — For Reviewers

18.0.70 Logistic Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `logistic-regression` — For Reviewers

18.0.71 Logistic Regression Diagnostics

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Logistic Regression Diagnostics** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Logistic Regression Diagnostics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `logistic-regression-diagnostics` — For Reviewers

18.0.72 Mann-Whitney U Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Mann-Whitney U Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Mann-Whitney U Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mann-whitney-u` — For Reviewers

18.0.73 McNemar's Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **McNemar's Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

McNemar's Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mcnemar-test` — For Reviewers

18.0.74 Method of Steepest Ascent

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Method of Steepest Ascent** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Method of Steepest Ascent yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `steepest-ascent` — For Reviewers

18.0.75 Minimum Detectable Effect

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Minimum Detectable Effect** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Minimum Detectable Effect yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `minimum-detectable-effect` — For Reviewers

18.0.76 Mixed ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Mixed ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Mixed ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `mixed-anova` — For Reviewers

18.0.77 Mixed-Effects Models: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Mixed-Effects Models: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Mixed-Effects Models: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mixed-models-intro` — For Reviewers

18.0.78 Model Selection with AIC and BIC

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Model Selection with AIC and BIC** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Model Selection with AIC and BIC yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `model-selection-aic-bic` — For Reviewers

18.0.79 Multi-State Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multi-State Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Multi-State Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `multi-state-models` — For Reviewers

18.0.80 Multicollinearity and VIF

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Multicollinearity and VIF** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Multicollinearity and VIF yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multicollinearity-vif` — For Reviewers

18.0.81 Multinomial Logistic Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Multinomial Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Multinomial Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multinomial-logistic` — For Reviewers

18.0.82 Multiple Comparisons: Overview

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Multiple Comparisons: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Multiple Comparisons: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multiple-comparisons-overview` — For Reviewers

18.0.83 Multiple Linear Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multiple Linear Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multiple Linear Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multiple-linear-regression` — For Reviewers

18.0.84 Negative Binomial Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Negative Binomial Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Negative Binomial Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `negative-binomial-regression` — For Reviewers

18.0.85 Nested vs Crossed Random Effects

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Nested vs Crossed Random Effects** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Nested vs Crossed Random Effects yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `nested-crossed-random-effects` — For Reviewers

18.0.86 Null and Alternative Hypotheses

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Null and Alternative Hypotheses** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Null and Alternative Hypotheses yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `null-alternative-hypotheses` — For Reviewers

18.0.87 Offsets in Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Offsets in Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Offsets in Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `offsets-in-regression` — For Reviewers

18.0.88 One-Proportion Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **One-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

One-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-proportion-test` — For Reviewers

18.0.89 One-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **One-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

One-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-sample-t-test` — For Reviewers

18.0.90 One-Sample z-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **One-Sample z-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

One-Sample z-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-sample-z-test` — For Reviewers

18.0.91 One-Way ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **One-Way ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> One-Way ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-way-anova` — For Reviewers

18.0.92 Optimal Designs: D, A, I

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Optimal Designs: D, A, I** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Optimal Designs: D, A, I yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `optimal-design-d-a-i` — For Reviewers

18.0.93 Ordinal Logistic Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Ordinal Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Ordinal Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `ordinal-logistic` — For Reviewers

18.0.94 Orthogonal Arrays

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Orthogonal Arrays** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Orthogonal Arrays yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `orthogonal-arrays` — For Reviewers

18.0.95 P-Values Explained

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **P-Values Explained** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

P-Values Explained yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `p-values-explained` — For Reviewers

18.0.96 Paired t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Paired t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence*

interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Paired t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `paired-t-test` — For Reviewers

18.0.97 Parametric Survival: Log-Normal

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Parametric Survival: Log-Normal** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Parametric Survival: Log-Normal yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `parametric-survival-lognormal` — For Reviewers

18.0.98 Parametric Survival: Weibull

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Parametric Survival: Weibull** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Parametric Survival: Weibull yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `parametric-survival-weibull` — For Reviewers

18.0.99 Partial Correlation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Partial Correlation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Partial Correlation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `partial-correlation` — For Reviewers

18.0.100 Pearson Correlation Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Pearson Correlation Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Pearson Correlation Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `pearson-correlation-test` — For Reviewers

18.0.101 Permutation Tests

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Permutation Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Permutation Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `permutation-tests` — For Reviewers

18.0.102 Plackett-Burman Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Plackett-Burman Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Plackett-Burman Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `plackett-burman` — For Reviewers

18.0.103 Poisson Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Poisson Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Poisson Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `poisson-regression` — For Reviewers

18.0.104 Polynomial Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Polynomial Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Polynomial Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `polynomial-regression` — For Reviewers

18.0.105 Post-Hoc Power: A Controversy

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Post-Hoc Power: A Controversy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Post-Hoc Power: A Controversy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `post-hoc-power-controversy` — For Reviewers

18.0.106 Post-Hoc Tests with Tukey HSD

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Post-Hoc Tests with Tukey HSD** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Post-Hoc Tests with Tukey HSD yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `post-hoc-tests-tukey` — For Reviewers

18.0.107 Power Analysis for DOE

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power Analysis for DOE** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power Analysis for DOE yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-in-doe` — For Reviewers

18.0.108 Power Analysis: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power Analysis: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power Analysis: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-analysis-introduction` — For Reviewers

18.0.109 Power for Agreement (Kappa)

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Agreement (Kappa)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Agreement (Kappa) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-agreement-kappa` — For Reviewers

18.0.110 Power for Bland-Altman Studies

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Bland-Altman Studies** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Bland-Altman Studies yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-bland-altman` — For Reviewers

18.0.111 Power for Chi-Squared Tests

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Chi-Squared Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Chi-Squared Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-chi-squared` — For Reviewers

18.0.112 Power for Cluster-RCT

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Cluster-RCT** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Cluster-RCT yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-cluster-rct` — For Reviewers

18.0.113 Power for Correlation Tests

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Correlation Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Correlation Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-correlation` — For Reviewers

18.0.114 Power for Cox Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Cox Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Cox Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-cox-regression` — For Reviewers

18.0.115 Power for Crossover Trials

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Crossover Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Crossover Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-crossover` — For Reviewers

18.0.116 Power for Diagnostic Accuracy

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Diagnostic Accuracy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Diagnostic Accuracy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-diagnostic-accuracy` — For Reviewers

18.0.117 Power for Equivalence (TOST)

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Equivalence (TOST)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Equivalence (TOST) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-equivalence-tost` — For Reviewers

18.0.118 Power for ICC

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for ICC** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for ICC yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-icc` — For Reviewers

18.0.119 Power for Linear Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Linear Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for Linear Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-linear-regression` — For Reviewers

18.0.120 Power for Logistic Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*.

Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-logistic-regression` — For Reviewers

18.0.121 Power for McNemar's Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for McNemar's Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for McNemar's Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-mcnemar` — For Reviewers

18.0.122 Power for Non-Inferiority Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Non-Inferiority Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Non-Inferiority Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-non-inferiority` — For Reviewers

18.0.123 Power for One-Proportion Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for One-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for One-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-one-proportion` — For Reviewers

18.0.124 Power for One-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for One-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for One-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-one-sample-t` — For Reviewers

18.0.125 Power for One-Way ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for One-Way ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for One-Way ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-anova` — For Reviewers

18.0.126 Power for Paired t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Paired t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for Paired t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-paired-t` — For Reviewers

18.0.127 Power for Repeated-Measures ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Repeated-Measures ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for Repeated-Measures ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-repeated-measures` — For Reviewers

18.0.128 Power for Stepped-Wedge Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Stepped-Wedge Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Stepped-Wedge Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-stepped-wedge` — For Reviewers

18.0.129 Power for the Log-Rank Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for the Log-Rank Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for the Log-Rank Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-logrank-test` — For Reviewers

18.0.130 Power for Two-Proportion Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Two-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Two-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-two-proportions` — For Reviewers

18.0.131 Principles of Blocking

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Principles of Blocking** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Principles of Blocking yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `blocking-principles` — For Reviewers

18.0.132 Probit Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Probit Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Probit Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `probit-regression` — For Reviewers

18.0.133 Quantile Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Quantile Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Quantile Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `quantile-regression` — For Reviewers

18.0.134 Quasi-Poisson Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Quasi-Poisson Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Quasi-Poisson Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `quasi-poisson` — For Reviewers

18.0.135 R-Squared and Adjusted R-Squared

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **R-Squared and Adjusted R-Squared** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> R-Squared and Adjusted R-Squared yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `r-squared-adjusted` — For Reviewers

18.0.136 Random-Intercept Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Random-Intercept Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Random-Intercept Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `random-intercepts` — For Reviewers

18.0.137 Random-Slope Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Random-Slope Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Random-Slope Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `random-slopes` — For Reviewers

18.0.138 Randomisation in Design

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Randomisation in Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Randomisation in Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `randomization-in-design` — For Reviewers

18.0.139 Randomised Complete Block Design

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Randomised Complete Block Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Randomised Complete Block Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `randomised-complete-block` — For Reviewers

18.0.140 Recurrent Events

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Recurrent Events** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Recurrent Events yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `recurrent-events` — For Reviewers

18.0.141 Repeated-Measures ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Repeated-Measures ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Repeated-Measures ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `repeated-measures-anova` — For Reviewers

18.0.142 Repeated-Measures Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Repeated-Measures Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Repeated-Measures Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `repeated-measures-design` — For Reviewers

18.0.143 Residual Diagnostics

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Residual Diagnostics** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Residual Diagnostics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `residual-diagnostics` — For Reviewers

18.0.144 Resolution and Aliasing

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Resolution and Aliasing** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Resolution and Aliasing yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `resolution-aliasing` — For Reviewers

18.0.145 Response Surface Methodology

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Response Surface Methodology** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Response Surface Methodology yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `response-surface-methodology` — For Reviewers

18.0.146 Restricted Mean Survival Time (RMST)

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Restricted Mean Survival Time (RMST)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Restricted Mean Survival Time (RMST) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `restricted-mean-survival-time` — For Reviewers

18.0.147 Ridge Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Ridge Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Ridge Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `ridge-regression` — For Reviewers

18.0.148 Robust Parameter Design

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Robust Parameter Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Robust Parameter Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `robust-parameter-design` — For Reviewers

18.0.149 Robust Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Robust Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Robust Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `robust-regression` — For Reviewers

18.0.150 ROC and AUC for Logistic Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **ROC and AUC for Logistic Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> ROC and AUC for Logistic Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `roc-auc-for-logistic` — For Reviewers

18.0.151 Royston-Parmar Flexible Parametric Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Royston-Parmar Flexible Parametric Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Royston-Parmar Flexible Parametric Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `flexible-parametric-royston-parmar` — For Reviewers

18.0.152 Sample Size for a Two-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Sample Size for a Two-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Sample Size for a Two-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-two-sample-t` — For Reviewers

18.0.153 Sample Size Sensitivity Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Sample Size Sensitivity Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Sample Size Sensitivity Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `sensitivity-analysis-sample-size` — For Reviewers

18.0.154 Schoenfeld Residuals

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Schoenfeld Residuals** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Schoenfeld Residuals yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `schoenfeld-residuals` — For Reviewers

18.0.155 Shapiro-Wilk Normality Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Shapiro-Wilk Normality Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Shapiro-Wilk Normality Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `shapiro-wilk-test` — For Reviewers

18.0.156 Signal-to-Noise Ratios

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Signal-to-Noise Ratios** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Signal-to-Noise Ratios yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `signal-to-noise-ratio` — For Reviewers

18.0.157 Simple Linear Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Simple Linear Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Simple Linear Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `simple-linear-regression` — For Reviewers

18.0.158 Simplex Centroid Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Simplex Centroid Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Simplex Centroid Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mixture-centroid` — For Reviewers

18.0.159 Simplex Lattice Mixture Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Simplex Lattice Mixture Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Simplex Lattice Mixture Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mixture-designs-simplex` — For Reviewers

18.0.160 Simulating Survival Data

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Simulating Survival Data** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Simulating Survival Data yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `survival-simulation` — For Reviewers

18.0.161 Spearman Rank Correlation Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Spearman Rank Correlation Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Spearman Rank Correlation Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `spearman-correlation-test` — For Reviewers

18.0.162 Spline Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Spline Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Spline Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `spline-regression` — For Reviewers

18.0.163 Split-Plot Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Split-Plot Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Split-Plot Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `split-plot-design` — For Reviewers

18.0.164 Stepwise Regression: Pitfalls

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Stepwise Regression: Pitfalls** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Stepwise Regression: Pitfalls yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `stepwise-regression-pitfalls` — For Reviewers

18.0.165 Stratified Cox Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Stratified Cox Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Stratified Cox Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `stratified-cox` — For Reviewers

18.0.166 Strip-Plot Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Strip-Plot Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Strip-Plot Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `strip-plot-design` — For Reviewers

18.0.167 Taguchi Methods

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Taguchi Methods** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Taguchi Methods yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `taguchi-methods` — For Reviewers

18.0.168 The Bootstrap: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Bootstrap: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> The Bootstrap: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bootstrap-introduction` — For Reviewers

18.0.169 The Brier Score for Survival

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Brier Score for Survival** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> The Brier Score for Survival yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `brier-score-survival` — For Reviewers

18.0.170 The Jackknife

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Jackknife** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> The Jackknife yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `jackknife` — For Reviewers

18.0.171 The Nelson-Aalen Estimator

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Nelson-Aalen Estimator** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> The Nelson-Aalen Estimator yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `nelson-aalen-estimator` — For Reviewers

18.0.172 The Runs Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Runs Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> The Runs Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `runs-test` — For Reviewers

18.0.173 The Sign Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Sign Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> The Sign Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `sign-test` — For Reviewers

18.0.174 Time-Dependent ROC

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Time-Dependent ROC** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Time-Dependent ROC yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `time-dependent-roc` — For Reviewers

18.0.175 Time-Varying Covariates

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Time-Varying Covariates** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Time-Varying Covariates yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `time-varying-covariates` — For Reviewers

18.0.176 Tobit Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Tobit Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Tobit Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `tobit-regression` — For Reviewers

18.0.177 Truncated Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Truncated Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Truncated Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `truncated-regression` — For Reviewers

18.0.178 Two-Proportion Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Two-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Two-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `two-proportion-test` — For Reviewers

18.0.179 Two-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Two-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Two-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `two-sample-t-test` — For Reviewers

18.0.180 Two-Way ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Two-Way ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Two-Way ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `two-way-anova` — For Reviewers

18.0.181 Type I and Type II Errors

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Type I and Type II Errors** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Type I and Type II Errors yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `type-i-type-ii-errors` — For Reviewers

18.0.182 Weighted Log-Rank Tests

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Weighted Log-Rank Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Weighted Log-Rank Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `weighted-log-rank` — For Reviewers

18.0.183 Welch's t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Welch's t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Welch's t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `welch-t-test` — For Reviewers

18.0.184 Wilcoxon Signed-Rank Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Wilcoxon Signed-Rank Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size*

with 95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Wilcoxon Signed-Rank Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `wilcoxon-signed-rank` — For Reviewers

18.0.185 Worked lme4::lmer Examples

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Worked lme4::lmer Examples** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Worked lme4::lmer Examples yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `lme4-lmer-examples` — For Reviewers

18.0.186 Zero-Inflated Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Zero-Inflated Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Zero-Inflated Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `zero-inflated-models` — For Reviewers

Chapter 19

CONSORT — randomised trials

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

CONSORT — *Consolidated Standards of Reporting Trials* — covers randomised controlled trials. The 25-item checklist and flow diagram are at https://www.consort-statement.org/. The templates below cover the analytic methods most often used for primary and secondary endpoints in a trial.

This page contains **141 method-specific templates**. Each is a skeleton with placeholders in italics; the prose is written so you can paste it into a manuscript and replace only the bracketed values.

19.0.1 2^k Factorial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **2^k Factorial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> 2^k Factorial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `factorial-2k` — For Reviewers

19.0.2 3^k Factorial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **3^k Factorial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> 3^k Factorial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `factorial-3k` — For Reviewers

19.0.3 Adaptive Trial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Adaptive Trial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Adaptive Trial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `adaptive-design` — For Reviewers

19.0.4 Alpha-Spending Functions

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Alpha-Spending Functions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Alpha-Spending Functions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `alpha-spending` — For Reviewers

19.0.5 Anderson-Darling Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Anderson-Darling Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Anderson-Darling Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `anderson-darling-test` — For Reviewers

19.0.6 Balanced Incomplete Block Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Balanced Incomplete Block Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Balanced Incomplete Block Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `balanced-incomplete-block` — For Reviewers

19.0.7 Bartlett's Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Bartlett's Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Bartlett's Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bartlett-test` — For Reviewers

19.0.8 Baseline Adjustment with ANCOVA

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Baseline Adjustment with ANCOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Baseline Adjustment with ANCOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `baseline-adjustment-ancova` — For Reviewers

19.0.9 Benjamini-Hochberg FDR

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Benjamini-Hochberg FDR** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Benjamini-Hochberg FDR yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `benjamini-hochberg-fdr` — For Reviewers

19.0.10 Benjamini-Yekutieli FDR

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Benjamini-Yekutieli FDR** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Benjamini-Yekutieli FDR yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `benjamini-yekutieli` — For Reviewers

19.0.11 Bland-Altman Limits of Agreement

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bland-Altman Limits of Agreement** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Bland-Altman Limits of Agreement yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bland-altman-analysis` — For Reviewers

19.0.12 Blinding Procedures

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Blinding Procedures** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95%*

confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Blinding Procedures yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `blinding-procedures` — For Reviewers

19.0.13 Block Randomisation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Block Randomisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Block Randomisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `block-randomization` — For Reviewers

19.0.14 Bonferroni Correction

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Bonferroni Correction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Bonferroni Correction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bonferroni-correction` — For Reviewers

19.0.15 Bootstrap Confidence Intervals

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bootstrap Confidence Intervals** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Bootstrap Confidence Intervals yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bootstrap-confidence-intervals` — For Reviewers

19.0.16 Box-Behnken Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Box-Behnken Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Box-Behnken Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `box-behnken-design` — For Reviewers

19.0.17 Central Composite Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Central Composite Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Central Composite Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `central-composite-design` — For Reviewers

19.0.18 Chi-Squared Goodness-of-Fit Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Chi-Squared Goodness-of-Fit Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Chi-Squared Goodness-of-Fit Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `chi-squared-goodness-of-fit` — For Reviewers

19.0.19 Chi-Squared Test of Independence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Chi-Squared Test of Independence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Chi-Squared Test of Independence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `chi-squared-independence` — For Reviewers

19.0.20 Clinical Equivalence Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Clinical Equivalence Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Clinical Equivalence Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `equivalence-clinical` — For Reviewers

19.0.21 Cluster-Randomised Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Cluster-Randomised Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Cluster-Randomised Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cluster-rct` — For Reviewers

19.0.22 Cochran-Mantel-Haenszel Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Cochran-Mantel-Haenszel Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Cochran-Mantel-Haenszel Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cochran-mantel-haenszel` — For Reviewers

19.0.23 Cohen's Kappa

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cohen's Kappa** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Cohen's Kappa yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cohens-kappa` — For Reviewers

19.0.24 Completely Randomised Design

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Completely Randomised Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Completely Randomised Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `completely-randomized-design` — For Reviewers

19.0.25 Conditional Power

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Conditional Power** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Conditional Power yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `conditional-power` — For Reviewers

19.0.26 Constrained Mixture Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Constrained Mixture Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Constrained Mixture Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `constrained-mixture` — For Reviewers

19.0.27 Crossover Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Crossover Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Crossover Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `crossover-designs` — For Reviewers

19.0.28 Crossover RCT Design

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Crossover RCT Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Crossover RCT Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `rct-design-crossover` — For Reviewers

19.0.29 Cutpoint Selection

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Cutpoint Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Cutpoint Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cutpoint-selection` — For Reviewers

19.0.30 Desirability Functions

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Desirability Functions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Desirability Functions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `desirability-function` — For Reviewers

19.0.31 Diagnostic Test Accuracy

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Diagnostic Test Accuracy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Diagnostic Test Accuracy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `diagnostic-accuracy` — For Reviewers

19.0.32 Effect Size: Cohen's d

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Size: Cohen's d** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect Size: Cohen's d yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `effect-size-cohens-d` — For Reviewers

19.0.33 Effect Size: Cohen's h

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Size: Cohen's h** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Effect Size: Cohen's h yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `effect-size-cohens-h` — For Reviewers

19.0.34 Effect Size: Eta-Squared

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Size: Eta-Squared** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Effect Size: Eta-Squared yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `effect-size-eta-squared` — For Reviewers

19.0.35 Effect Sizes: Overview

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Effect Sizes: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Effect Sizes: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `effect-sizes-overview` — For Reviewers

19.0.36 Equivalence Testing with TOST

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Equivalence Testing with TOST** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Equivalence Testing with TOST yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `equivalence-tost` — For Reviewers

19.0.37 Factorial Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Factorial Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Factorial Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `factorial-trial` — For Reviewers

19.0.38 Fisher's Exact Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Fisher's Exact Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Fisher's Exact Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `fisher-exact-test` — For Reviewers

19.0.39 Fractional Factorial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Fractional Factorial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Fractional Factorial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `fractional-factorial-design` — For Reviewers

19.0.40 Friedman Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Friedman Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Friedman Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `friedman-test` — For Reviewers

19.0.41 Graeco-Latin Square Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Graeco-Latin Square Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Graeco-Latin Square Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `graeco-latin-square` — For Reviewers

19.0.42 Holm's Step-Down Correction

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Holm's Step-Down Correction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Holm's Step-Down Correction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `holm-correction` — For Reviewers

19.0.43 Interim Analyses and Group Sequential Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Interim Analyses and Group Sequential Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Interim Analyses and Group Sequential Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `interim-analysis-group-sequential` — For Reviewers

19.0.44 Intraclass Correlation Coefficient (ICC)

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Intraclass Correlation Coefficient (ICC)** as implemented in *[package vX.Y]*

in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Intraclass Correlation Coefficient (ICC) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `icc-continuous-agreement` — For Reviewers

19.0.45 ITT vs Per-Protocol Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **ITT vs Per-Protocol Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

ITT vs Per-Protocol Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `itt-vs-pp-analysis` — For Reviewers

19.0.46 Kendall's Tau

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Kendall's Tau** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Kendall's Tau yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `kendall-tau` — For Reviewers

19.0.47 Kolmogorov-Smirnov Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Kolmogorov-Smirnov Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Kolmogorov-Smirnov Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `kolmogorov-smirnov-test` — For Reviewers

19.0.48 Kruskal-Wallis Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Kruskal-Wallis Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Kruskal-Wallis Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `kruskal-wallis` — For Reviewers

19.0.49 Latin Square Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Latin Square Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Latin Square Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `latin-square` — For Reviewers

19.0.50 Levene's Test of Variances

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Levene's Test of Variances** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Levene's Test of Variances yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `levene-test` — For Reviewers

19.0.51 Likelihood Ratios

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Likelihood Ratios** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Likelihood Ratios yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `likelihood-ratios` — For Reviewers

19.0.52 Mann-Whitney U Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Mann-Whitney U Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].*

Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Mann-Whitney U Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mann-whitney-u` — For Reviewers

19.0.53 McNemar's Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **McNemar's Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

McNemar's Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mcnemar-test` — For Reviewers

19.0.54 Method of Steepest Ascent

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Method of Steepest Ascent** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Method of Steepest Ascent yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `steepest-ascent` — For Reviewers

19.0.55 Minimisation Algorithm

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Minimisation Algorithm** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Minimisation Algorithm yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `minimization-algorithm` — For Reviewers

19.0.56 Minimum Detectable Effect

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Minimum Detectable Effect** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Minimum Detectable Effect yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `minimum-detectable-effect` — For Reviewers

19.0.57 Missing Data in RCTs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Missing Data in RCTs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Missing Data in RCTs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `missing-data-rct` — For Reviewers

19.0.58 Mixed ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Mixed ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Mixed ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `mixed-anova` — For Reviewers

19.0.59 Multiple Comparisons: Overview

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multiple Comparisons: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multiple Comparisons: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multiple-comparisons-overview` — For Reviewers

19.0.60 Multiple Imputation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multiple Imputation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multiple Imputation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `multiple-imputation` — For Reviewers

19.0.61 Non-Inferiority Margin Selection

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Non-Inferiority Margin Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Non-Inferiority Margin Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `non-inferiority-margin` — For Reviewers

19.0.62 Null and Alternative Hypotheses

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Null and Alternative Hypotheses** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Null and Alternative Hypotheses yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `null-alternative-hypotheses` — For Reviewers

19.0.63 O'Brien-Fleming Boundary

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **O'Brien-Fleming Boundary** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

O'Brien-Fleming Boundary yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `obrien-fleming-boundary` — For Reviewers

19.0.64 One-Proportion Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **One-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

One-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-proportion-test` — For Reviewers

19.0.65 One-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **One-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

One-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-sample-t-test` — For Reviewers

19.0.66 One-Sample z-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **One-Sample z-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> One-Sample z-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-sample-z-test` — For Reviewers

19.0.67 One-Way ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **One-Way ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> One-Way ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `one-way-anova` — For Reviewers

19.0.68 Optimal Designs: D, A, I

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Optimal Designs: D, A, I** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Optimal Designs: D, A, I yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `optimal-design-d-a-i` — For Reviewers

19.0.69 Orthogonal Arrays

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Orthogonal Arrays** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Orthogonal Arrays yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `orthogonal-arrays` — For Reviewers

19.0.70 P-Values Explained

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **P-Values Explained** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

P-Values Explained yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `p-values-explained` — For Reviewers

19.0.71 Paired t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Paired t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were

checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Paired t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `paired-t-test` — For Reviewers

19.0.72 Parallel-Group RCT Design

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Parallel-Group RCT Design** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Parallel-Group RCT Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `rct-design-parallel` — For Reviewers

19.0.73 Pearson Correlation Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Pearson Correlation Test** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Pearson Correlation Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `pearson-correlation-test` — For Reviewers

19.0.74 Permutation Tests

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Permutation Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Permutation Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `permutation-tests` — For Reviewers

19.0.75 Plackett-Burman Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Plackett-Burman Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Plackett-Burman Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `plackett-burman` — For Reviewers

19.0.76 Pocock Boundary

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Pocock Boundary** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Pocock Boundary yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `pocock-boundary` — For Reviewers

19.0.77 Post-Hoc Power: A Controversy

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Post-Hoc Power: A Controversy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Post-Hoc Power: A Controversy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `post-hoc-power-controversy` — For Reviewers

19.0.78 Post-Hoc Tests with Tukey HSD

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Post-Hoc Tests with Tukey HSD** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Post-Hoc Tests with Tukey HSD yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `post-hoc-tests-tukey` — For Reviewers

19.0.79 Power Analysis for DOE

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power Analysis for DOE** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power Analysis for DOE yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-in-doe` — For Reviewers

19.0.80 Power Analysis: Introduction

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power Analysis: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power Analysis: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-analysis-introduction` — For Reviewers

19.0.81 Power for Agreement (Kappa)

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Agreement (Kappa)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Agreement (Kappa) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-agreement-kappa` — For Reviewers

19.0.82 Power for Bland-Altman Studies

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Bland-Altman Studies** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Bland-Altman Studies yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-bland-altman` — For Reviewers

19.0.83 Power for Chi-Squared Tests

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Chi-Squared Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Chi-Squared Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-chi-squared` — For Reviewers

19.0.84 Power for Cluster-RCT

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Cluster-RCT** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size*

with 95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Cluster-RCT yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-cluster-rct` — For Reviewers

19.0.85 Power for Correlation Tests

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Correlation Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Correlation Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-correlation` — For Reviewers

19.0.86 Power for Cox Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Cox Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Cox Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-cox-regression` — For Reviewers

19.0.87 Power for Crossover Trials

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Crossover Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Crossover Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-crossover` — For Reviewers

19.0.88 Power for Diagnostic Accuracy

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Diagnostic Accuracy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Diagnostic Accuracy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-diagnostic-accuracy` — For Reviewers

19.0.89 Power for Equivalence (TOST)

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Equivalence (TOST)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for Equivalence (TOST) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-equivalence-tost` — For Reviewers

19.0.90 Power for ICC

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for ICC** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for ICC yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-icc` — For Reviewers

19.0.91 Power for Linear Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Linear Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for Linear Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-linear-regression` — For Reviewers

19.0.92 Power for Logistic Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with*

95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-logistic-regression` — For Reviewers

19.0.93 Power for McNemar's Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for McNemar's Test** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for McNemar's Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-mcnemar` — For Reviewers

19.0.94 Power for Non-Inferiority Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Non-Inferiority Trials** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Power for Non-Inferiority Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-non-inferiority` — For Reviewers

19.0.95 Power for One-Proportion Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for One-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for One-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-one-proportion` — For Reviewers

19.0.96 Power for One-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for One-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for One-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-one-sample-t` — For Reviewers

19.0.97 Power for One-Way ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for One-Way ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Power for One-Way ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-anova` — For Reviewers

19.0.98 Power for Paired t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Paired t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for Paired t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-paired-t` — For Reviewers

19.0.99 Power for Repeated-Measures ANOVA

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Power for Repeated-Measures ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Power for Repeated-Measures ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `power-repeated-measures` — For Reviewers

19.0.100 Power for Stepped-Wedge Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Stepped-Wedge Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Stepped-Wedge Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-stepped-wedge` — For Reviewers

19.0.101 Power for the Log-Rank Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for the Log-Rank Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for the Log-Rank Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-logrank-test` — For Reviewers

19.0.102 Power for Two-Proportion Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Power for Two-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Power for Two-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-two-proportions` — For Reviewers

19.0.103 Predictive Values and Prevalence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Predictive Values and Prevalence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Predictive Values and Prevalence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `predictive-values-prevalence` — For Reviewers

19.0.104 Principles of Blocking

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Principles of Blocking** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Principles of Blocking yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `blocking-principles` — For Reviewers

19.0.105 Randomisation in Design

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Randomisation in Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Randomisation in Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `randomization-in-design` — For Reviewers

19.0.106 Randomisation Methods

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Randomisation Methods** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Randomisation Methods yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `randomization-methods` — For Reviewers

19.0.107 Randomised Complete Block Design

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Randomised Complete Block Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Randomised Complete Block Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `randomised-complete-block` — For Reviewers

19.0.108 Reliability and Cronbach's Alpha

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Reliability and Cronbach's Alpha** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Reliability and Cronbach's Alpha yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `reliability-cronbach-alpha` — For Reviewers

19.0.109 Repeated-Measures ANOVA

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Repeated-Measures ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Repeated-Measures ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `repeated-measures-anova` — For Reviewers

19.0.110 Repeated-Measures Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Repeated-Measures Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Repeated-Measures Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `repeated-measures-design` — For Reviewers

19.0.111 Resolution and Aliasing

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Resolution and Aliasing** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Resolution and Aliasing yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `resolution-aliasing` — For Reviewers

19.0.112 Response Surface Methodology

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Response Surface Methodology** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Response Surface Methodology yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `response-surface-methodology` — For Reviewers

19.0.113 Robust Parameter Design

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Robust Parameter Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Robust Parameter Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `robust-parameter-design` — For Reviewers

19.0.114 ROC Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **ROC Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

ROC Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `roc-analysis` — For Reviewers

19.0.115 Sample Size for a Two-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Sample Size for a Two-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Sample Size for a Two-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `power-two-sample-t` — For Reviewers

19.0.116 Sample Size Re-Estimation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Sample Size Re-Estimation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Sample Size Re-Estimation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `sample-size-reestimation` — For Reviewers

19.0.117 Sample Size Sensitivity Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Sample Size Sensitivity Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Sample Size Sensitivity Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `sensitivity-analysis-sample-size` — For Reviewers

19.0.118 Sensitivity Analyses in Clinical Trials

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Sensitivity Analyses in Clinical Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Sensitivity Analyses in Clinical Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `sensitivity-analysis-clinical` — For Reviewers

19.0.119 Shapiro-Wilk Normality Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Shapiro-Wilk Normality Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Shapiro-Wilk Normality Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `shapiro-wilk-test` — For Reviewers

19.0.120 Signal-to-Noise Ratios

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Signal-to-Noise Ratios** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Signal-to-Noise Ratios yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `signal-to-noise-ratio` — For Reviewers

19.0.121 Simplex Centroid Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Simplex Centroid Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Simplex Centroid Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `mixture-centroid` — For Reviewers

19.0.122 Simplex Lattice Mixture Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Simplex Lattice Mixture Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Simplex Lattice Mixture Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `mixture-designs-simplex` — For Reviewers

19.0.123 Spearman Rank Correlation Test

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Spearman Rank Correlation Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Spearman Rank Correlation Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `spearman-correlation-test` — For Reviewers

19.0.124 Split-Plot Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Split-Plot Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Split-Plot Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `split-plot-design` — For Reviewers

19.0.125 Stepped-Wedge Trial

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Stepped-Wedge Trial** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Stepped-Wedge Trial yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `stepped-wedge-trial` — For Reviewers

19.0.126 Stratified Randomisation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Stratified Randomisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*.

Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Stratified Randomisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `stratified-randomization` — For Reviewers

19.0.127 Strip-Plot Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Strip-Plot Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Strip-Plot Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `strip-plot-design` — For Reviewers

19.0.128 Subgroup Analyses

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Subgroup Analyses** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Subgroup Analyses yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `subgroup-analysis` — For Reviewers

19.0.129 Subgroup Forest Plots

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Subgroup Forest Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Subgroup Forest Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `forest-plot-subgroup` — For Reviewers

19.0.130 Taguchi Methods

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Taguchi Methods** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Taguchi Methods yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `taguchi-methods` — For Reviewers

19.0.131 The Bootstrap: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Bootstrap: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> The Bootstrap: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bootstrap-introduction` — For Reviewers

19.0.132 The Jackknife

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Jackknife** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> The Jackknife yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `jackknife` — For Reviewers

19.0.133 The Runs Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Runs Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> The Runs Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `runs-test` — For Reviewers

19.0.134 The Sign Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Sign Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> The Sign Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `sign-test` — For Reviewers

19.0.135 Two-Proportion Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Two-Proportion Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Two-Proportion Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `two-proportion-test` — For Reviewers

19.0.136 Two-Sample t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Two-Sample t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Two-Sample t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `two-sample-t-test` — For Reviewers

19.0.137 Two-Way ANOVA

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Two-Way ANOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Two-Way ANOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `two-way-anova` — For Reviewers

19.0.138 Type I and Type II Errors

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Type I and Type II Errors** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Type I and Type II Errors yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `type-i-type-ii-errors` — For Reviewers

19.0.139 Weighted Kappa

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Weighted Kappa** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Weighted Kappa yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `weighted-kappa` — For Reviewers

19.0.140 Welch's t-Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Welch's t-Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Welch's t-Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `welch-t-test` — For Reviewers

19.0.141 Wilcoxon Signed-Rank Test

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Wilcoxon Signed-Rank Test** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Wilcoxon Signed-Rank Test yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `wilcoxon-signed-rank` — For Reviewers

Chapter 20

TRIPOD-AI — prediction models

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

TRIPOD-AI — *Transparent Reporting of a multivariable prediction model for Individual Prognosis Or Diagnosis, AI extension* — covers prediction models, including those built with machine learning. The checklist is at https://www.tripod-statement.org/. The templates below cover model development, validation, calibration, and fairness reporting.

This page contains **163 method-specific templates**. Each is a skeleton with placeholders in italics; the prose is written so you can paste it into a manuscript and replace only the bracketed values.

20.0.1 A Complete mlr3 Workflow

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **A Complete mlr3 Workflow** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> A Complete mlr3 Workflow yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `mlr3-workflow` — For Reviewers

20.0.2 A Complete tidymodels Workflow

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **A Complete tidymodels Workflow** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> A Complete tidymodels Workflow yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `tidymodels-workflow` — For Reviewers

20.0.3 Accelerated Failure Time Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Accelerated Failure Time Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Accelerated Failure Time Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `accelerated-failure-time` — For Reviewers

20.0.4 Adaptive Trial Designs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Adaptive Trial Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Adaptive Trial Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `adaptive-design` — For Reviewers

20.0.5 Added-Variable Plots

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Added-Variable Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Added-Variable Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `added-variable-plots` — For Reviewers

20.0.6 Alpha-Spending Functions

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Alpha-Spending Functions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Alpha-Spending Functions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `alpha-spending` — For Reviewers

20.0.7 Anomaly Detection

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Anomaly Detection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Anomaly Detection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `anomaly-detection` — For Reviewers

20.0.8 Bagging

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Bagging** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Bagging yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bagging` — For Reviewers

20.0.9 Baseline Adjustment with ANCOVA

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Baseline Adjustment with ANCOVA** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Baseline Adjustment with ANCOVA yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `baseline-adjustment-ancova` — For Reviewers

20.0.10 Best Subset Selection

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Best Subset Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Best Subset Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `best-subset-selection` — For Reviewers

20.0.11 Beta Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Beta Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Beta Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `beta-regression` — For Reviewers

20.0.12 Bland-Altman Limits of Agreement

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bland-Altman Limits of Agreement** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Bland-Altman Limits of Agreement yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bland-altman-analysis` — For Reviewers

20.0.13 Blinding Procedures

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Blinding Procedures** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Blinding Procedures yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `blinding-procedures` — For Reviewers

20.0.14 Block Randomisation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Block Randomisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Block Randomisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `block-randomization` — For Reviewers

20.0.15 Calibration Plots

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Calibration Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions

were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Calibration Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `calibration-plots` — For Reviewers

20.0.16 CatBoost

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **CatBoost** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

CatBoost yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `catboost` — For Reviewers

20.0.17 Censoring Types

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Censoring Types** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Censoring Types yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `censoring-types` — For Reviewers

20.0.18 Centring and Scaling Predictors

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Centring and Scaling Predictors** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Centring and Scaling Predictors yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `centered-scaled-predictors` — For Reviewers

20.0.19 Checking Cox Assumptions

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Checking Cox Assumptions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Checking Cox Assumptions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cox-assumptions-check` — For Reviewers

20.0.20 Class Weights for Imbalanced Data

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Class Weights for Imbalanced Data** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Class Weights for Imbalanced Data yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `class-weights-imbalanced` — For Reviewers

20.0.21 Clinical Equivalence Trials

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Clinical Equivalence Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Clinical Equivalence Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `equivalence-clinical` — For Reviewers

20.0.22 Cluster-Randomised Trials

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cluster-Randomised Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Cluster-Randomised Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cluster-rct` — For Reviewers

20.0.23 Cohen's Kappa

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cohen's Kappa** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Cohen's Kappa yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cohens-kappa` — For Reviewers

20.0.24 Competing Risks: Cumulative Incidence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Competing Risks: Cumulative Incidence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Competing Risks: Cumulative Incidence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `competing-risks-cif` — For Reviewers

20.0.25 Concordance and the C-Index

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Concordance and the C-Index** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Concordance and the C-Index yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `concordance-c-index` — For Reviewers

20.0.26 Conditional Power

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Conditional Power** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Conditional Power yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `conditional-power` — For Reviewers

20.0.27 Conditional Survival

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Conditional Survival** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Conditional Survival yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `conditional-survival` — For Reviewers

20.0.28 Contrasts in R

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Contrasts in R** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Contrasts in R yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `contrasts-in-r` — For Reviewers

20.0.29 Convolutional Neural Networks: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Convolutional Neural Networks: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Convolutional Neural Networks: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cnn-introduction` — For Reviewers

20.0.30 Cook's Distance

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cook's Distance** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Cook's Distance yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cooks-distance` — For Reviewers

20.0.31 Cox Proportional Hazards Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cox Proportional Hazards Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Cox Proportional Hazards Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cox-proportional-hazards` — For Reviewers

20.0.32 Cox Regression as Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cox Regression as Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Cox Regression as Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `survival-regression-cox` — For Reviewers

20.0.33 Cross-Validation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cross-Validation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Cross-Validation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cross-validation` — For Reviewers

20.0.34 Crossover RCT Design

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Crossover RCT Design** as implemented in *[package vX.Y]* in R *[vX.Y]*.

Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Crossover RCT Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `rct-design-crossover` — For Reviewers

20.0.35 Cutpoint Selection

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Cutpoint Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Cutpoint Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cutpoint-selection` — For Reviewers

20.0.36 DBSCAN as an ML Tool

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **DBSCAN as an ML Tool** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

DBSCAN as an ML Tool yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `clustering-dbscan-ml` — For Reviewers

20.0.37 Decision Trees (CART)

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Decision Trees (CART)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Decision Trees (CART) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `decision-trees-cart` — For Reviewers

20.0.38 Diagnostic Test Accuracy

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Diagnostic Test Accuracy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Diagnostic Test Accuracy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `diagnostic-accuracy` — For Reviewers

20.0.39 Dirichlet Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Dirichlet Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Dirichlet Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `dirichlet-regression` — For Reviewers

20.0.40 Dropout and Early Stopping

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Dropout and Early Stopping** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Dropout and Early Stopping yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `dropout-early-stopping` — For Reviewers

20.0.41 Dummy Coding

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Dummy Coding** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Dummy Coding yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `dummy-coding` — For Reviewers

20.0.42 Effect (Deviation) Coding

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Effect (Deviation) Coding** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with*

95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Effect (Deviation) Coding yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `effect-coding` — For Reviewers

20.0.43 Elastic Net

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Elastic Net** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Elastic Net yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `elastic-net` — For Reviewers

20.0.44 Extremely Randomised Trees

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Extremely Randomised Trees** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Extremely Randomised Trees yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `extremely-randomized-trees` — For Reviewers

20.0.45 Factorial Trials

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Factorial Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Factorial Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `factorial-trial` — For Reviewers

20.0.46 Feature Engineering

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Feature Engineering** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Feature Engineering yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `feature-engineering` — For Reviewers

20.0.47 Feature Selection

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Feature Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Feature Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `feature-selection` — For Reviewers

20.0.48 Feedforward Networks in torch

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Feedforward Networks in torch** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Feedforward Networks in torch yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `feedforward-networks-torch` — For Reviewers

20.0.49 Fine-Gray Subdistribution Hazards

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Fine-Gray Subdistribution Hazards** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Fine-Gray Subdistribution Hazards yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `fine-gray-regression` — For Reviewers

20.0.50 Frailty Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Frailty Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Frailty Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `frailty-models` — For Reviewers

20.0.51 GAMMs: Additive Mixed Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **GAMMs: Additive Mixed Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> GAMMs: Additive Mixed Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `gamm-additive-mixed` — For Reviewers

20.0.52 GAMs: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **GAMs: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> GAMs: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `gam-introduction` — For Reviewers

20.0.53 Generalised Estimating Equations

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Generalised Estimating Equations** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect*

size with 95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Generalised Estimating Equations yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `generalized-estimating-equations` — For Reviewers

20.0.54 glmer: Generalised Mixed Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **glmer: Generalised Mixed Models** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

glmer: Generalised Mixed Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `glmer-generalized-mixed` — For Reviewers

20.0.55 Gradient Boosting

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Gradient Boosting** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Gradient Boosting yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `gradient-boosting` — For Reviewers

20.0.56 Hazard, Survival, and Cumulative Hazard

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Hazard, Survival, and Cumulative Hazard** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Hazard, Survival, and Cumulative Hazard yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `hazard-survival-cumulative` — For Reviewers

20.0.57 Hurdle Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Hurdle Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Hurdle Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `hurdle-models` — For Reviewers

20.0.58 Interactions in Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Interactions in Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Interactions in Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `interactions-in-regression` — For Reviewers

20.0.59 Interim Analyses and Group Sequential Designs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Interim Analyses and Group Sequential Designs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Interim Analyses and Group Sequential Designs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `interim-analysis-group-sequential` — For Reviewers

20.0.60 Interval-Censored Data

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Interval-Censored Data** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Interval-Censored Data yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `interval-censored-data` — For Reviewers

20.0.61 Intraclass Correlation Coefficient (ICC)

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Intraclass Correlation Coefficient (ICC)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Intraclass Correlation Coefficient (ICC) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `icc-continuous-agreement` — For Reviewers

20.0.62 Isolation Forest

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Isolation Forest** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Isolation Forest yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `isolation-forest` — For Reviewers

20.0.63 Isotonic Regression Calibration

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Isotonic Regression Calibration** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Isotonic Regression Calibration yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `isotonic-regression` — For Reviewers

20.0.64 ITT vs Per-Protocol Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **ITT vs Per-Protocol Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> ITT vs Per-Protocol Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `itt-vs-pp-analysis` — For Reviewers

20.0.65 Joint Longitudinal-Survival Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Joint Longitudinal-Survival Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Joint Longitudinal-Survival Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `joint-longitudinal-survival` — For Reviewers

20.0.66 k-Means as an ML Tool

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **k-Means as an ML Tool** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> k-Means as an ML Tool yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `clustering-kmeans-ml` — For Reviewers

20.0.67 k-Nearest Neighbours Classification

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **k-Nearest Neighbours Classification** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> k-Nearest Neighbours Classification yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `knn-classification` — For Reviewers

20.0.68 k-Nearest Neighbours Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **k-Nearest Neighbours Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> k-Nearest Neighbours Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `knn-regression` — For Reviewers

20.0.69 Kaplan-Meier Estimation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Kaplan-Meier Estimation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Kaplan-Meier Estimation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `kaplan-meier` — For Reviewers

20.0.70 Kernel SVMs

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Kernel SVMs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Kernel SVMs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `svm-kernel` — For Reviewers

20.0.71 Landmark Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Landmark Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Landmark Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `landmark-analysis` — For Reviewers

20.0.72 Lasso Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Lasso Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Lasso Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `lasso-regression` — For Reviewers

20.0.73 Left Truncation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Left Truncation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Left Truncation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `left-truncation` — For Reviewers

20.0.74 Leverage and Influence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Leverage and Influence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Leverage and Influence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `leverage-influence` — For Reviewers

20.0.75 LightGBM

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **LightGBM** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> LightGBM yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `lightgbm` — For Reviewers

20.0.76 Likelihood Ratios

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Likelihood Ratios** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Likelihood Ratios yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `likelihood-ratios` — For Reviewers

20.0.77 LIME Explanations

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **LIME Explanations** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> LIME Explanations yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `lime-explanations` — For Reviewers

20.0.78 Linear Discriminant as ML

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Linear Discriminant as ML** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Linear Discriminant as ML yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `linear-discriminant-ml` — For Reviewers

20.0.79 Linear Regression Assumptions

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Linear Regression Assumptions** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Linear Regression Assumptions yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `regression-assumptions` — For Reviewers

20.0.80 Linear Support Vector Machines

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Linear Support Vector Machines** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Linear Support Vector Machines yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `svm-linear` — For Reviewers

20.0.81 Log-Rank Test: Details

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Log-Rank Test: Details** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Log-Rank Test: Details yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `log-rank-test-details` — For Reviewers

20.0.82 Logistic Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `logistic-regression` — For Reviewers

20.0.83 Logistic Regression as ML

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Logistic Regression as ML** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Logistic Regression as ML yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `logistic-regression-ml` — For Reviewers

20.0.84 Logistic Regression Diagnostics

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Logistic Regression Diagnostics** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Logistic Regression Diagnostics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `logistic-regression-diagnostics` — For Reviewers

20.0.85 Minimisation Algorithm

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Minimisation Algorithm** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Minimisation Algorithm yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `minimization-algorithm` — For Reviewers

20.0.86 Missing Data in RCTs

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Missing Data in RCTs** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Missing Data in RCTs yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `missing-data-rct` — For Reviewers

20.0.87 Mixed-Effects Models: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Mixed-Effects Models: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Mixed-Effects Models: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `mixed-models-intro` — For Reviewers

20.0.88 Model Selection with AIC and BIC

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Model Selection with AIC and BIC** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Model Selection with AIC and BIC yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `model-selection-aic-bic` — For Reviewers

20.0.89 Multi-State Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Multi-State Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Multi-State Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `multi-state-models` — For Reviewers

20.0.90 Multicollinearity and VIF

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Multicollinearity and VIF** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Multicollinearity and VIF yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `multicollinearity-vif` — For Reviewers

20.0.91 Multinomial Logistic Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multinomial Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multinomial Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multinomial-logistic` — For Reviewers

20.0.92 Multiple Imputation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multiple Imputation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multiple Imputation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multiple-imputation` — For Reviewers

20.0.93 Multiple Linear Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multiple Linear Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multiple Linear Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `multiple-linear-regression` — For Reviewers

20.0.94 Naive Bayes

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Naive Bayes** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Naive Bayes yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `naive-bayes` — For Reviewers

20.0.95 Negative Binomial Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Negative Binomial Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Negative Binomial Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `negative-binomial-regression` — For Reviewers

20.0.96 Nested Cross-Validation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Nested Cross-Validation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95%*

confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Nested Cross-Validation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `nested-cv` — For Reviewers

20.0.97 Nested vs Crossed Random Effects

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Nested vs Crossed Random Effects** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Nested vs Crossed Random Effects yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `nested-crossed-random-effects` — For Reviewers

20.0.98 Neural Networks: Introduction

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Neural Networks: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Neural Networks: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `neural-networks-intro` — For Reviewers

20.0.99 Non-Inferiority Margin Selection

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Non-Inferiority Margin Selection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Non-Inferiority Margin Selection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `non-inferiority-margin` — For Reviewers

20.0.100 O'Brien-Fleming Boundary

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **O'Brien-Fleming Boundary** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> O'Brien-Fleming Boundary yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `obrien-fleming-boundary` — For Reviewers

20.0.101 Offsets in Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Offsets in Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Offsets in Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `offsets-in-regression` — For Reviewers

20.0.102 Ordinal Logistic Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Ordinal Logistic Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Ordinal Logistic Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `ordinal-logistic` — For Reviewers

20.0.103 Parallel-Group RCT Design

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Parallel-Group RCT Design** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Parallel-Group RCT Design yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `rct-design-parallel` — For Reviewers

20.0.104 Parametric Survival: Log-Normal

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Parametric Survival: Log-Normal** as implemented in *[package vX.Y]* in R

[vX.Y]. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Parametric Survival: Log-Normal yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `parametric-survival-lognormal` — For Reviewers

20.0.105 Parametric Survival: Weibull

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Parametric Survival: Weibull** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Parametric Survival: Weibull yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `parametric-survival-weibull` — For Reviewers

20.0.106 Partial Correlation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Partial Correlation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Partial Correlation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `partial-correlation` — For Reviewers

20.0.107 Partial Dependence Plots

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Partial Dependence Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Partial Dependence Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `partial-dependence-plots` — For Reviewers

20.0.108 Platt Scaling

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Platt Scaling** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Platt Scaling yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `platt-scaling` — For Reviewers

20.0.109 Pocock Boundary

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Pocock Boundary** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Pocock Boundary yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `pocock-boundary` — For Reviewers

20.0.110 Poisson Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Poisson Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Poisson Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `poisson-regression` — For Reviewers

20.0.111 Polynomial Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Polynomial Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Polynomial Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `polynomial-regression` — For Reviewers

20.0.112 Predictive Values and Prevalence

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Predictive Values and Prevalence** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Predictive Values and Prevalence yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `predictive-values-prevalence` — For Reviewers

20.0.113 Preprocessing with recipes

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Preprocessing with recipes** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Preprocessing with recipes yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `recipes-preprocessing` — For Reviewers

20.0.114 Probit Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Probit Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Probit Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `probit-regression` — For Reviewers

20.0.115 Quantile Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Quantile Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Quantile Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `quantile-regression` — For Reviewers

20.0.116 Quasi-Poisson Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Quasi-Poisson Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Quasi-Poisson Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `quasi-poisson` — For Reviewers

20.0.117 R-Squared and Adjusted R-Squared

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **R-Squared and Adjusted R-Squared** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

R-Squared and Adjusted R-Squared yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `r-squared-adjusted` — For Reviewers

20.0.118 Random Forests

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Random Forests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Random Forests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `random-forest` — For Reviewers

20.0.119 Random-Intercept Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Random-Intercept Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Random-Intercept Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `random-intercepts` — For Reviewers

20.0.120 Random-Slope Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Random-Slope Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Random-Slope Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `random-slopes` — For Reviewers

20.0.121 Randomisation Methods

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Randomisation Methods** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Randomisation Methods yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `randomization-methods` — For Reviewers

20.0.122 Recurrent Events

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Recurrent Events** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Recurrent Events yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `recurrent-events` — For Reviewers

20.0.123 Regularisation in ML

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Regularisation in ML** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with*

95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Regularisation in ML yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `regularization-ridge-lasso-ml` — For Reviewers

20.0.124 Reliability and Cronbach's Alpha

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Reliability and Cronbach's Alpha** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Reliability and Cronbach's Alpha yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `reliability-cronbach-alpha` — For Reviewers

20.0.125 Residual Diagnostics

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Residual Diagnostics** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Residual Diagnostics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `residual-diagnostics` — For Reviewers

20.0.126 Restricted Mean Survival Time (RMST)

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Restricted Mean Survival Time (RMST)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Restricted Mean Survival Time (RMST) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `restricted-mean-survival-time` — For Reviewers

20.0.127 Ridge Regression

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Ridge Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Ridge Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `ridge-regression` — For Reviewers

20.0.128 RNNs and LSTMs: Introduction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **RNNs and LSTMs: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

RNNs and LSTMs: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `rnn-lstm-intro` — For Reviewers

20.0.129 Robust Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Robust Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Robust Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `robust-regression` — For Reviewers

20.0.130 ROC Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **ROC Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

ROC Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `roc-analysis` — For Reviewers

20.0.131 ROC and AUC for Logistic Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **ROC and AUC for Logistic Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary*

outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

ROC and AUC for Logistic Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `roc-auc-for-logistic` — For Reviewers

20.0.132 Royston-Parmar Flexible Parametric Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Royston-Parmar Flexible Parametric Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Royston-Parmar Flexible Parametric Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `flexible-parametric-royston-parmar` — For Reviewers

20.0.133 Sample Size Re-Estimation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Sample Size Re-Estimation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Sample Size Re-Estimation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `sample-size-reestimation` — For Reviewers

20.0.134 Schoenfeld Residuals

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Schoenfeld Residuals** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Schoenfeld Residuals yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `schoenfeld-residuals` — For Reviewers

20.0.135 Sensitivity Analyses in Clinical Trials

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Sensitivity Analyses in Clinical Trials** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Sensitivity Analyses in Clinical Trials yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `sensitivity-analysis-clinical` — For Reviewers

20.0.136 SHAP Values

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **SHAP Values** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

SHAP Values yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `shap-values` — For Reviewers

20.0.137 Simple Linear Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Simple Linear Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Simple Linear Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `simple-linear-regression` — For Reviewers

20.0.138 Simulating Survival Data

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Simulating Survival Data** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Simulating Survival Data yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `survival-simulation` — For Reviewers

20.0.139 SMOTE for Imbalanced Classes

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **SMOTE for Imbalanced Classes** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with*

95% confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

SMOTE for Imbalanced Classes yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `smote-imbalanced` — For Reviewers

20.0.140 Spline Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Spline Regression** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Spline Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `spline-regression` — For Reviewers

20.0.141 Stacking Ensembles

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Stacking Ensembles** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Stacking Ensembles yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `stacking-ensembles` — For Reviewers

20.0.142 Stepped-Wedge Trial

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Stepped-Wedge Trial** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Stepped-Wedge Trial yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `stepped-wedge-trial` — For Reviewers

20.0.143 Stepwise Regression: Pitfalls

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Stepwise Regression: Pitfalls** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Stepwise Regression: Pitfalls yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `stepwise-regression-pitfalls` — For Reviewers

20.0.144 Stratified Cox Models

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Stratified Cox Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Stratified Cox Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `stratified-cox` — For Reviewers

20.0.145 Stratified Randomisation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Stratified Randomisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Stratified Randomisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `stratified-randomization` — For Reviewers

20.0.146 Subgroup Analyses

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Subgroup Analyses** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Subgroup Analyses yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `subgroup-analysis` — For Reviewers

20.0.147 Subgroup Forest Plots

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Subgroup Forest Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*.

Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Subgroup Forest Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `forest-plot-subgroup` — For Reviewers

20.0.148 Supervised Learning: Overview

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Supervised Learning: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Supervised Learning: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `supervised-learning-overview` — For Reviewers

20.0.149 The Bias-Variance Tradeoff

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **The Bias-Variance Tradeoff** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

The Bias-Variance Tradeoff yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bias-variance-tradeoff` — For Reviewers

20.0.150 The Brier Score for Survival

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Brier Score for Survival** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> The Brier Score for Survival yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `brier-score-survival` — For Reviewers

20.0.151 The Nelson-Aalen Estimator

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **The Nelson-Aalen Estimator** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> The Nelson-Aalen Estimator yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `nelson-aalen-estimator` — For Reviewers

20.0.152 Time-Dependent ROC

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Time-Dependent ROC** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*.

Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Time-Dependent ROC yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `time-dependent-roc` — For Reviewers

20.0.153 Time-Varying Covariates

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Time-Varying Covariates** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Time-Varying Covariates yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `time-varying-covariates` — For Reviewers

20.0.154 Tobit Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Tobit Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Tobit Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `tobit-regression` — For Reviewers

20.0.155 Train-Test Splits

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Train-Test Splits** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Train-Test Splits yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `train-test-split` — For Reviewers

20.0.156 Transformers: Overview

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Transformers: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Transformers: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `transformer-overview` — For Reviewers

20.0.157 Truncated Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Truncated Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Truncated Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `truncated-regression` — For Reviewers

20.0.158 Variable Importance

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Variable Importance** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Variable Importance yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `variable-importance` — For Reviewers

20.0.159 Weighted Kappa

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Weighted Kappa** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Weighted Kappa yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `weighted-kappa` — For Reviewers

20.0.160 Weighted Log-Rank Tests

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Weighted Log-Rank Tests** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Weighted Log-Rank Tests yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `weighted-log-rank` — For Reviewers

20.0.161 Worked lme4::lmer Examples

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Worked lme4::lmer Examples** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Worked lme4::lmer Examples yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `lme4-lmer-examples` — For Reviewers

20.0.162 XGBoost

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **XGBoost** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> XGBoost yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `xgboost` — For Reviewers

20.0.163 Zero-Inflated Models

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Zero-Inflated Models** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Zero-Inflated Models yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `zero-inflated-models` — For Reviewers

Chapter 21

PRISMA — systematic reviews & meta-analyses

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

PRISMA — *Preferred Reporting Items for Systematic reviews and Meta-Analyses* — covers systematic reviews and meta-analyses. The 27-item checklist and flow diagram are at https://www.prisma-statement.org/. The templates below cover pairwise and network meta-analysis methods.

This page contains **25 method-specific templates**. Each is a skeleton with placeholders in italics; the prose is written so you can paste it into a manuscript and replace only the bracketed values.

21.0.1 Bivariate SROC Curves

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bivariate SROC Curves** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Bivariate SROC Curves yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bivariate-sroc` — For Reviewers

21.0.2 Converting Between Effect Sizes

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Converting Between Effect Sizes** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Converting Between Effect Sizes yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `effect-size-conversion` — For Reviewers

21.0.3 Cumulative Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cumulative Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Cumulative Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `cumulative-meta-analysis` — For Reviewers

21.0.4 Egger's Test for Funnel Asymmetry

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Egger's Test for Funnel Asymmetry** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Egger's Test for Funnel Asymmetry yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `egger-test` — For Reviewers

21.0.5 Fixed-Effect vs. Random-Effects Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Fixed-Effect vs. Random-Effects Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Fixed-Effect vs. Random-Effects Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `fixed-vs-random-effects` — For Reviewers

21.0.6 Forest Plots in Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Forest Plots in Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Forest Plots in Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `forest-plot-meta` — For Reviewers

21.0.7 Funnel Plots

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Funnel Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence*

interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Funnel Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `funnel-plot-meta` — For Reviewers

21.0.8 Individual Participant Data Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Individual Participant Data Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Individual Participant Data Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `ipd-meta-analysis` — For Reviewers

21.0.9 Leave-One-Out Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Leave-One-Out Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Leave-One-Out Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `leave-one-out` — For Reviewers

21.0.10 Meta-Analysis of Diagnostic Accuracy

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Meta-Analysis of Diagnostic Accuracy** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Meta-Analysis of Diagnostic Accuracy yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `diagnostic-test-meta` — For Reviewers

21.0.11 Meta-Regression

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Meta-Regression** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Meta-Regression yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `meta-regression` — For Reviewers

21.0.12 Network Meta-Analysis: Introduction

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Network Meta-Analysis: Introduction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Network Meta-Analysis: Introduction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `network-meta-analysis-intro` — For Reviewers

21.0.13 NMA Consistency and Inconsistency

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **NMA Consistency and Inconsistency** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> NMA Consistency and Inconsistency yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `nma-consistency` — For Reviewers

21.0.14 NMA League Tables

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **NMA League Tables** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> NMA League Tables yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `nma-league-tables` — For Reviewers

21.0.15 Pooling Correlation Coefficients

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Pooling Correlation Coefficients** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Pooling Correlation Coefficients yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `correlation-pooling` — For Reviewers

21.0.16 Pooling Log-Odds Ratios

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Pooling Log-Odds Ratios** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Pooling Log-Odds Ratios yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `log-odds-ratio-pooling` — For Reviewers

21.0.17 Pooling Risk Ratios

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Pooling Risk Ratios** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Pooling Risk Ratios yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `risk-ratio-pooling` — For Reviewers

21.0.18 Prediction Intervals in Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Prediction Intervals in Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Prediction Intervals in Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `prediction-interval-meta` — For Reviewers

21.0.19 Quantifying Heterogeneity: Q and I^2

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Quantifying Heterogeneity: Q and I^2** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Quantifying Heterogeneity: Q and I^2 yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `heterogeneity-q-i2` — For Reviewers

21.0.20 Selection Models for Publication Bias

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Selection Models for Publication Bias** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Selection Models for Publication Bias yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `selection-models-bias` — For Reviewers

21.0.21 Standardised Mean Difference: Hedges' g

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Standardised Mean Difference: Hedges' g** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Standardised Mean Difference: Hedges' g yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `smd-hedges-g` — For Reviewers

21.0.22 Subgroup Meta-Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Subgroup Meta-Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Subgroup Meta-Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `subgroup-meta-analysis` — For Reviewers

21.0.23 SUCRA Rankings

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **SUCRA Rankings** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

SUCRA Rankings yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `nma-sucra` — For Reviewers

21.0.24 Tau-Squared Estimation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Tau-Squared Estimation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Tau-Squared Estimation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `tau-squared-estimation` — For Reviewers

21.0.25 Trim-and-Fill

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Trim-and-Fill** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Trim-and-Fill yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `trim-and-fill` — For Reviewers

Chapter 22

ARRIVE — animal research

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

ARRIVE — *Animal Research: Reporting of In Vivo Experiments* — covers preclinical animal research. The 21-item checklist is at https://arriveguidelines.org/. The templates below cover the statistical analyses most commonly required.

This page contains **50 method-specific templates**. Each is a skeleton with placeholders in italics; the prose is written so you can paste it into a manuscript and replace only the bracketed values.

22.0.1 Alignment with BWA and Bowtie

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Alignment with BWA and Bowtie** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Alignment with BWA and Bowtie yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `bwa-bowtie-alignment` — For Reviewers

22.0.2 ATAC-Seq Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **ATAC-Seq Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> ATAC-Seq Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `atac-seq-analysis` — For Reviewers

22.0.3 Batch Correction with ComBat

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Batch Correction with ComBat** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Batch Correction with ComBat yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `batch-correction-combat` — For Reviewers

22.0.4 Bulk RNA-seq Differential Expression with DESeq2

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Bulk RNA-seq Differential Expression with DESeq2** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Bulk RNA-seq Differential Expression with DESeq2 yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `bulk-rnaseq-differential-expression` — For Reviewers

22.0.5 Cell-Type Annotation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Cell-Type Annotation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Cell-Type Annotation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `scrna-cell-type-annotation` — For Reviewers

22.0.6 ChIP-Seq Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **ChIP-Seq Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> ChIP-Seq Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `chip-seq-analysis` — For Reviewers

22.0.7 Copy Number Variation Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Copy Number Variation Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*.

Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Copy Number Variation Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `cnv-analysis` — For Reviewers

22.0.8 Counting Reads with featureCounts

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Counting Reads with featureCounts** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Counting Reads with featureCounts yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `counting-reads-feature-counts` — For Reviewers

22.0.9 Differential Expression with edgeR

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Differential Expression with edgeR** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Differential Expression with edgeR yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `edger-differential-expression` — For Reviewers

22.0.10 Differential Expression with limma-voom

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Differential Expression with limma-voom** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Differential Expression with limma-voom yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `limma-voom` — For Reviewers

22.0.11 DNA Methylation Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **DNA Methylation Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> DNA Methylation Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `methylation-analysis` — For Reviewers

22.0.12 Drug-Target Interaction Mining

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Drug-Target Interaction Mining** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Drug-Target Interaction Mining yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `drug-target-interaction` — For Reviewers

22.0.13 FASTQ Quality Control

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **FASTQ Quality Control** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

FASTQ Quality Control yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `fastq-quality-control` — For Reviewers

22.0.14 Finding Marker Genes

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Finding Marker Genes** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Finding Marker Genes yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `scrna-marker-genes` — For Reviewers

22.0.15 Gene Annotation with biomaRt

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Gene Annotation with biomaRt** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Gene Annotation with biomaRt yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `biomart-annotation` — For Reviewers

22.0.16 Gene Ontology Enrichment

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Gene Ontology Enrichment** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Gene Ontology Enrichment yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `gene-ontology-enrichment` — For Reviewers

22.0.17 GSEA Preranked Analysis

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **GSEA Preranked Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

GSEA Preranked Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `gsea-preranked` — For Reviewers

22.0.18 GSVA Single-Sample Enrichment

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **GSVA Single-Sample Enrichment** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> GSVA Single-Sample Enrichment yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `gsva-single-sample` — For Reviewers

22.0.19 Heatmaps for RNA-seq

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Heatmaps for RNA-seq** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Heatmaps for RNA-seq yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `heatmaps-rnaseq` — For Reviewers

22.0.20 Integration with Harmony

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Integration with Harmony** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Integration with Harmony yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `scrna-integration-harmony` — For Reviewers

22.0.21 KEGG Pathway Enrichment

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **KEGG Pathway Enrichment** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

KEGG Pathway Enrichment yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `pathway-enrichment-kegg` — For Reviewers

22.0.22 MA Plots

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **MA Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

MA Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `ma-plots` — For Reviewers

22.0.23 Metagenomic Profiling

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Metagenomic Profiling** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95%*

confidence interval reported as the primary outcome]. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Metagenomic Profiling yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `metagenomic-profiling` — For Reviewers

22.0.24 Microbiome Analysis with DADA2

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Microbiome Analysis with DADA2** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Microbiome Analysis with DADA2 yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `microbiome-dada2` — For Reviewers

22.0.25 Microbiome Diversity Metrics

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Microbiome Diversity Metrics** as implemented in *[package vX.Y]* in R *[vX.Y].* Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

Microbiome Diversity Metrics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `microbiome-diversity` — For Reviewers

22.0.26 Multi-Omics Integration

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multi-Omics Integration** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multi-Omics Integration yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `multiomics-integration` — For Reviewers

22.0.27 Multiple Sequence Alignment

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Multiple Sequence Alignment** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Multiple Sequence Alignment yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `msa-biostrings` — For Reviewers

22.0.28 PCA of RNA-seq Samples

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **PCA of RNA-seq Samples** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> PCA of RNA-seq Samples yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `pca-of-rnaseq` — For Reviewers

22.0.29 Phylogenetic Trees with ape

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Phylogenetic Trees with ape** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Phylogenetic Trees with ape yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `phylogenetic-trees-ape` — For Reviewers

22.0.30 Population Genetics Basics

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Population Genetics Basics** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Population Genetics Basics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `population-genetics-basics` — For Reviewers

22.0.31 Protein Structure Prediction

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Protein Structure Prediction** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Protein Structure Prediction yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `protein-structure-prediction` — For Reviewers

22.0.32 Proteomics with MSstats

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Proteomics with MSstats** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Proteomics with MSstats yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `proteomics-msstats` — For Reviewers

22.0.33 Pseudoalignment with Salmon and Kallisto

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Pseudoalignment with Salmon and Kallisto** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Pseudoalignment with Salmon and Kallisto yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `salmon-kallisto-pseudoalignment` — For Reviewers

22.0.34 Read Trimming and Adapter Removal

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Read Trimming and Adapter Removal** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Read Trimming and Adapter Removal yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `read-trimming-adapters` — For Reviewers

22.0.35 RNA-seq Normalisation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **RNA-seq Normalisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

RNA-seq Normalisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `rnaseq-normalization` — For Reviewers

22.0.36 scRNA Clustering

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **scRNA Clustering** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

scRNA Clustering yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `scrna-clustering` — For Reviewers

22.0.37 scRNA Normalisation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **scRNA Normalisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> scRNA Normalisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `scrna-normalization` — For Reviewers

22.0.38 scRNA QC and Filtering

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **scRNA QC and Filtering** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> scRNA QC and Filtering yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `scrna-qc-filtering` — For Reviewers

22.0.39 Sequence Alignment: Overview

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Sequence Alignment: Overview** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Sequence Alignment: Overview yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `sequence-alignment-overview` — For Reviewers

22.0.40 Single-Cell with Seurat

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Single-Cell with Seurat** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Single-Cell with Seurat yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `single-cell-seurat-intro` — For Reviewers

22.0.41 Spatial Transcriptomics

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Spatial Transcriptomics** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Spatial Transcriptomics yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `spatial-transcriptomics` — For Reviewers

22.0.42 STAR: Spliced Alignment

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **STAR: Spliced Alignment** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

STAR: Spliced Alignment yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `star-spliced-alignment` — For Reviewers

22.0.43 Structural Variant Detection

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Structural Variant Detection** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Structural Variant Detection yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `structural-variant-detection` — For Reviewers

22.0.44 Surrogate Variable Analysis (SVA)

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Surrogate Variable Analysis (SVA)** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Surrogate Variable Analysis (SVA) yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `surrogate-variable-analysis-sva` — For Reviewers

22.0.45 Trajectory Analysis

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Trajectory Analysis** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Trajectory Analysis yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `scrna-trajectory-analysis` — For Reviewers

22.0.46 Transcript-to-Gene Summarisation

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Transcript-to-Gene Summarisation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]. [Effect size with 95% confidence interval reported as the primary outcome].* Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any].*

Results paragraph (copy-paste):

> Transcript-to-Gene Summarisation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]. [Sensitivity analysis result].*

Reviewer checklist for this method. → `tx-to-gene-summarization` — For Reviewers

22.0.47 Variant Annotation with VEP

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Variant Annotation with VEP** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Variant Annotation with VEP yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `annotation-variant-vep` — For Reviewers

22.0.48 Variant Calling with GATK

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **Variant Calling with GATK** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

Variant Calling with GATK yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `variant-calling-gatk` — For Reviewers

22.0.49 VCF Manipulation

Methods paragraph (copy-paste, fill in italics):

[N=] participants/observations were analysed using **VCF Manipulation** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use $\alpha = 0.05$; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

VCF Manipulation yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `vcf-manipulation` — For Reviewers

22.0.50 Volcano Plots

Methods paragraph (copy-paste, fill in italics):

> *[N=]* participants/observations were analysed using **Volcano Plots** as implemented in *[package vX.Y]* in R *[vX.Y]*. Assumptions were checked: *[list checks performed]*. *[Effect size with 95% confidence interval reported as the primary outcome]*. Two-sided p-values use α = *0.05*; *[multiple-comparison adjustment if any]*.

Results paragraph (copy-paste):

> Volcano Plots yielded *[estimate]* (95% CI *[lower, upper]*, p = *[value]*). *[Diagnostic / assumption summary]*. *[Sensitivity analysis result]*.

Reviewer checklist for this method. → `volcano-plots` — For Reviewers

22.1 Week 4, Session 5 — Explanation vs prediction; reporting

Workflow lab: Goal → Approach → Execution → Check → Report.

22.1.1 Learning objectives

- Distinguish explanatory from predictive modelling, and choose the correct evaluation metric for each.
- Map STROBE, TRIPOD, STARD, and CONSORT onto the study designs each is meant for.
- Produce a publication-ready regression table with `gtsummary` whose numbers trace directly to the fitted model.

22.1.2 Prerequisites

The regression tools covered earlier in the book.

22.1.3 Background

Breiman's 1998 essay *The Two Cultures* and Shmueli's 2010 essay *To Explain or to Predict?* draw the same line from opposite sides. In explanatory modelling the goal is inference — what is the estimated effect of X on Y, adjusted for confounders? — and the right evaluation metric is bias-unbiasedness, interval coverage, and interpretability. In predictive modelling the goal is accuracy on unseen data, and the right evaluation metric is out-of-sample loss, calibration, and decision utility. The statistical machinery overlaps but the habits around it do not; a regression that makes an excellent explanation often makes a lacklustre prediction, and vice versa.

Reporting guidelines are the mechanism by which the field enforces discipline around these two cultures. STROBE for observational studies, CONSORT for randomised trials, STARD for diagnostic accuracy, and TRIPOD (now TRIPOD-AI) for prediction models each specify the items a reader needs to evaluate the claim. A checklist filled in as you write is much easier than one filled in at submission.

22.1.4 Setup

```
library(tidyverse)
library(gtsummary)
library(broom)
library(palmerpenguins)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

22.1.5 1. Goal

Fit a single linear model to the penguins data and present it two ways: once as an explanatory analysis (what drives body mass?) and once as a predictive model (can we predict body mass on held-out birds?).

22.1.6 2. Approach

```
peng <- penguins |> drop_na(body_mass_g, flipper_length_mm, sex, species)
fit_expl <- lm(body_mass_g ~ flipper_length_mm + sex + species, data =
 ↪  peng)
```

22.1.7 3. Execution

```
peng |>
  ggplot(aes(flipper_length_mm, body_mass_g, colour = species)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE, linewidth = 0.8) +
  labs(x = "Flipper length (mm)", y = "Body mass (g)")
```

22.1.8 4. Check

Explanatory evaluation: effect sizes with intervals, residual diagnostics, and adjusted R^2.

```
glance(fit_expl) |> select(r.squared, adj.r.squared, sigma, df,
 ↪  df.residual)
```

Predictive evaluation: a 5-fold split, hand-coded to keep the example transparent.

```
k <- 5
folds <- sample(rep(seq_len(k), length.out = nrow(peng)))
rmse_k <- sapply(seq_len(k), function(i) {
  tr <- peng[folds != i, ]
  te <- peng[folds == i, ]
  f  <- lm(body_mass_g ~ flipper_length_mm + sex + species, data = tr)
  sqrt(mean((predict(f, te) - te$body_mass_g)^2))
})
mean_rmse <- mean(rmse_k)
mean_rmse
```

22.1.9 5. Report

```
tbl_regression(fit_expl, intercept = TRUE) |>
  modify_caption("**Table 1. Linear-regression estimates for body mass
   ↪  (g).**")
```

> In the Palmer penguins dataset (n = `nrow(peng)`), body mass was associated with flipper length, sex, and species (adjusted R^2 = `round(summary(fit_expl)$adj.r.squared,2)`). Out-of-sample performance, estimated by 5-fold cross-validation, was RMSE = `round(mean_rmse,0)` g.

22.1.9.1 Reporting-guideline map

Design	Guideline	URL
Randomised trial	CONSORT	https://www.consort-statement.org/
Observational study	STROBE	https://www.strobe-statement.org/
Diagnostic-accuracy study	STARD	https://www.equator-network.org/reporting-guidelines/stard/
Prediction-model study	TRIPOD / TRIPOD-AI	https://www.tripod-statement.org/
Systematic review	PRISMA	http://prisma-statement.org/

22.1.10 Common pitfalls

- Choosing a predictor set by R^2 and reporting a predictive claim, or tuning a predictive model and reporting causal-sounding coefficients.

- Filling in a reporting checklist at submission rather than while drafting.
- Assuming in-sample R^2 generalises.

22.1.11 Further reading

- Breiman L. (2001). *Statistical Modeling: The Two Cultures.*
- Shmueli G. (2010). *To Explain or to Predict?* Statistical Science.
- Writing a report.

22.1.12 Session info

```
sessionInfo()
```

Chapter 23

The research workflow

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

APPENDIX · RESEARCH WORKFLOW

Nine stages from a research question to knowledge that survives replication.

Statistics is a thread running through a much longer process. The diagram below names the stages of that process and links each to the parts of the curriculum that cover it in detail. No stage is optional; errors made at one are paid for at the next.

```
flowchart TD
  Q[1. Question] --> M[2. Measurements]
  M --> D[3. Design]
  D --> A[4. Acquisition]
  A --> Desc[5. Description]
  Desc --> An[6. Analysis]
  An --> I[7. Interpretation / Prediction]
  I --> V[8. Validation]
  V --> K[9. Knowledge / Decisions]
```

23.1 1. Question

A well-formed research question has a population, an exposure or intervention, a comparator, an outcome, and a time frame. The PICO framework from evidence-based medicine is a useful checklist. A vague question leads to a vague design and a vague answer; a well-formed question survives peer review before a single datum is collected. *Covered in:* Scientific process and research workflow, Systematic reviews and prisma.

23.2 2. Measurements

Every question depends on the measurement scale you choose to answer it. Accuracy, precision, reliability, and the difference between a measurement and a proxy are the vocabulary. *Covered in:* Data types tidy data accuracy and precision, Kappa icc bland altman.

23.3 3. Design

Design is where you choose what the data can and cannot tell you. A well-designed observational study answers some questions better than a badly designed RCT; a good RCT answers them all. *Covered in:* Observational designs and strobe.

23.4 4. Acquisition

Getting the data into a computer faithfully is itself a statistical problem: missingness, measurement error, and batch effects start here. *Covered in:* Import joins and missingness with dplyr, Mcar mar mnar.

23.5 5. Description

Before any inference, look at the data. A well-designed plot is often the whole answer. *Covered in:* Ggplot2 grammar and multi panel layouts, Descriptive statistics and table 1.

23.6 6. Analysis

The formal model that takes the data and returns an estimate, an interval, and a decision. *Covered in:* Courses 1–4, throughout.

23.7 7. Interpretation / prediction

What does the estimate mean in the scale of the science? Effect size, clinical significance, and the distinction between an average and an individual prediction live here. *Covered in:* Calibration discrimination roc auc brier score, Time dependent brier ipa external validation.

23.8 8. Validation

Does the finding hold in data not used to find it? External validation, nested CV, and replication pool here. *Covered in:* Cross validation nested cv bootstrap 632, Time dependent brier ipa external validation.

23.9 9. Knowledge / decisions

The act of writing it down so that others can trust, cite, and build on the finding.
Covered in: Writing a report, Explanation vs prediction reporting.

Chapter 24

Glossary

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

APPENDIX · GLOSSARY

Plain-English definitions of the terms this curriculum uses the most.

Each entry links to the first lab in which the term appears.

24.1 A

α **(alpha).** The probability of a type I error — rejecting a true null hypothesis. Conventional values are 0.05 and 0.01. Hypothesis testing p values type i ii errors.

ANCOVA. Analysis of covariance: a linear model combining categorical predictors and continuous covariates, commonly used to adjust RCT analyses for baseline. Ancova in rcts baseline adjustment.

ANOVA. Analysis of variance: a linear model with categorical predictors. One way anova and contrasts emmeans.

Assumption. A condition the chosen test needs to be interpretable — normality, equal variance, independence, linearity. Two sample and paired t tests.

24.2 B

Bias. Systematic deviation of an estimator from the quantity being estimated. Populations samples and the central limit theorem.

Bootstrap. Resampling with replacement to approximate the sampling distribution of a statistic. Bootstrap and permutation tests.

Brier score. Mean squared error between predicted probabilities and observed outcomes. Calibration discrimination roc auc brier score.

24.3 C

CI (confidence interval). A range of plausible values for a parameter; in repeated sampling, 95% CIs cover the true value 95% of the time. Bootstrap and permutation tests.

Cohen's d. Standardised difference in means; a common effect size for two-group comparisons. Two sample and paired t tests.

Competing risk. An event whose occurrence precludes or alters the probability of the event of interest. Competing risks and multistate models.

Confounding. A third variable distorting the association between exposure and outcome. Multiple regression confounding interaction centring.

Cox model. A proportional-hazards regression for time-to-event outcomes. Survival primer km log rank cox ph.

CV (cross-validation). Splitting data into folds to estimate generalisation error. Cross validation nested cv bootstrap 632.

24.4 D

DAG (directed acyclic graph). A graphical representation of causal assumptions. Dags with dagitty and ggdag.

24.5 E

Effect size. A standardised measure of the magnitude of an effect, independent of sample size. Two sample and paired t tests.

24.6 F

FDR (false discovery rate). The expected proportion of false positives among rejected nulls. Fdr knockoffs replication crisis.

Fisher's exact test. Exact test of independence for 2×2 tables, appropriate when expected cell counts are small. Two proportions chi square risk and odds.

24.7 G

GAM (generalised additive model). A regression with smooth non-linear terms, fitted via penalised splines. Gams with mgcv.

GEE (generalised estimating equations). A marginal-model approach for clustered or repeated data. Glmms and gee.

GLM (generalised linear model). A regression with a link function and exponential-family error. Logistic regression binomial glm.

24.8 H

Hazard ratio. Ratio of hazard rates between two groups in a survival model. Survival primer km log rank cox ph.

24.9 I

ICC (intraclass correlation). Proportion of variance attributable to clustering. Kappa icc bland altman.

Interaction. The effect of one predictor depends on the value of another. Multiple regression confounding interaction centring.

IPTW (inverse-probability-of-treatment weighting). Propensity-score method that reweights observations to emulate a trial. Propensity scores and iptw.

24.10 K

Kaplan-Meier. Non-parametric estimator of the survival function. Survival primer km log rank cox ph.

Kruskal-Wallis. Non-parametric one-way ANOVA on ranks. Non parametric tests.

24.11 L

Lasso. L1-regularised regression; produces sparse coefficient estimates. Regularisation ridge lasso elastic net.

Likelihood. The probability of the observed data under a model, viewed as a function of the parameters. Maximum likelihood estimation.

Linear model. A model of the form $y = X\beta + \varepsilon$. Simple linear regression.

Logistic regression. GLM with a logit link for binary outcomes. Logistic regression binomial glm.

24.12 M

MAR (missing at random). Missingness depends on observed variables only. Mcar mar mnar.

MCAR (missing completely at random). Missingness is independent of all variables. Mcar mar mnar.

MCMC. Markov-chain Monte Carlo; the Bayesian posterior-sampling workhorse. Brms stan loo hierarchical models.

MDE (minimum detectable effect). Smallest effect your study has power to detect. Power closed form and simulation.

Meta-analysis. Combining effect estimates across studies. Meta analysis basics.

Mixed model. Regression combining fixed and random effects. Linear mixed models with lme4.

MNAR (missing not at random). Missingness depends on unobserved values. Mcar mar mnar.

Multiple imputation. Imputing missing values several times and pooling the analyses. Multiple imputation with mice.

24.13 N

Non-parametric. A test or estimator that makes weak distributional assumptions. Non parametric tests.

24.14 O

Odds ratio. Ratio of odds between two groups; the natural scale for logistic regression. Logistic regression binomial glm.

Outlier. An observation far from the bulk of the data. Diagnostics residuals qq leverage cook s distance vif.

Overdispersion. Variance exceeding the model's nominal variance; common in Poisson regression. Poisson and negative binomial regression.

24.15 P

p-value. Probability of data as or more extreme than observed, assuming the null. Hypothesis testing p values type i ii errors.

Paired test. A test comparing matched observations rather than independent samples. Two sample and paired t tests.

PCA (principal components analysis). Linear dimension reduction by orthogonal projection onto directions of maximum variance. Pca factor analysis cca lda.

Permutation test. Inference by shuffling labels to build a null distribution. Bootstrap and permutation tests.

Poisson regression. GLM with a log link for counts. Poisson and negative binomial regression.

Power. Probability of detecting an effect if it exists; $1 - \beta$. Sample size power and quarto reporting.

Pre-registration. A timestamped record of the research plan before data analysis. Pre registration and statistical analysis plans.

Propensity score. Probability of treatment given covariates. Propensity scores and iptw.

Pseudoreplication. Treating correlated observations as independent replicates. Bench and translational design.

24.16 R

Random effect. A model coefficient treated as a draw from a distribution. Linear mixed models with lme4.

Randomisation. Allocating units to arms by a chance mechanism. Randomised controlled trials.

Regression to the mean. Tendency of extreme values to be closer to the mean on remeasurement. Dichotomisation change scores regression to the mean.

Reliability. Consistency of repeated measurements. Kappa icc bland altman.

Resampling. Bootstrap, permutation, and cross-validation collectively. Bootstrap and permutation tests.

Residual. Observed minus predicted. Diagnostics residuals qq leverage cook s distance vif.

Risk ratio. Ratio of event probabilities between two groups. Two proportions chi square risk and odds.

Robust SE. A standard error computed without assuming homoscedasticity. Robust regression weighted ls hc sandwich standard errors.

ROC / AUC. Receiver operating characteristic; area under it measures discrimination. Calibration discrimination roc auc brier score.

24.17 S

SAP (statistical analysis plan). The formal plan for a trial's analysis, written before data are seen. Pre registration and statistical analysis plans.

SE (standard error). Standard deviation of a sampling distribution. Populations samples and the central limit theorem.

SEM (standard error of the mean). Standard error of a sample mean. Populations samples and the central limit theorem.

Shapiro-Wilk. A test of normality, powerful in small samples. Two sample and paired t tests.

Spearman correlation. Rank-based correlation. Pearson and spearman correlation.

Survival. Time-to-event analysis. Survival primer km log rank cox ph.

24.18 T

Type I / II error. False-positive and false-negative errors of a test. Hypothesis testing p values type i ii errors.

24.19 V

VIF (variance inflation factor). Measure of collinearity among regression predictors. Diagnostics residuals qq leverage cook s distance vif.

24.20 W

Wilcoxon test. Non-parametric test for paired (signed-rank) or unpaired (rank-sum) data. Non parametric tests.

Chapter 25

Common errors

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

APPENDIX · COMMON ERRORS

A field guide to the R, Quarto, and `renv` errors that will bite you first.

Each entry lists the error message, the usual cause, and the fix that works most of the time.

25.1 R

Error: `thereisnopackagecalled'xyz'`

Cause: the package is not installed in the library R is looking in.

Fix: `install.packages("xyz")`. If you are using `renv`, run `renv::restore()` instead to keep the lock file authoritative.

Error: `couldnotfindfunction"ggplot"`

Cause: package is installed but not loaded.

Fix: `library(ggplot2)` (or `library(tidyverse)`).

Error: `non-finitevaluesuppliedbyoptim`

Cause: the model failed to converge, often because of extreme starting values, perfect separation (logistic regression), or a misspecified formula.

Fix: rescale predictors, use sensible starting values, or for logistic regression check for zero cells in the outcome-by-predictor table.

Error: `contrastscanbeappliedonlytofactorswith2ormorelevels`

Cause: a factor predictor has been filtered down to a single level.

Fix: check `table(your_data$predictor)` and restore levels, or drop that predictor from the model.

Error: `ErrorinUseMethod("predict"):noapplicablemethodfor'predict'applied toanobjectofclass"c('list')"`

Cause: you passed the wrong object to `predict()`. Usually a list instead of the fitted model.

Fix: check the object with `class()` and pass the model object itself, e.g. `predict(fit,newdata=nd)`.

Error: `Loadingrequiredpackage:foo—thereisnopackagecalled'foo'`

Cause: an installed package has a dependency that is missing.

Fix: `install.packages("foo",dependencies=TRUE)`.

25.2 Quarto

Error: `quarto:commandnotfound`

Cause: Quarto is not on your PATH.

Fix: install from https://quarto.org and restart your terminal. On macOS, check that `/usr/local/bin` (or the Homebrew prefix) is in your PATH.

Error: `Failedtorender:yoursystemismissingpandoc`

Cause: older versions of R Quarto missed the bundled pandoc.

Fix: upgrade Quarto to the latest release.

Error: `Errorrunningfiltershortcodes`

Cause: a Quarto extension is missing after cloning.

Fix: run `quartoadd` for the extension, or ignore if the extension is optional.

Error: `outputfile:docs/index.htmldoesnotexist`

Cause: `quartorender` was interrupted or `output-dir` changed.

Fix: re-run `quartorender` from a clean state; if the interruption wedged `.quarto/`, delete that cache directory and try again.

Error: `GitHubPagesdeploymentfailed:404`

Cause: Pages is pointing at the wrong branch or path.

Fix: in the repo Settings → Pages, select the `gh-pages` branch and the `/(root)` folder. Our workflow deploys to `gh-pages` by default.

25.3 renv

Error: `renv::restore()failedwith:cannotinstallpackage'xyz'fromsource`

Cause: a system library is missing. The error above this line usually names the header file.

Fix: install the matching `-dev` package on Linux, the corresponding Homebrew package on macOS, or Rtools on Windows. Common culprits:

- `libcurl4-openssl-dev` for `curl`
- `libxml2-dev` for `XML` / `xml2`
- `libssl-dev` for `openssl`
- `libfontconfig1-dev`, `libfreetype6-dev`, `libharfbuzz-dev`, `libfribidi-dev` for `systemfonts`, `ragg`, `textshaping`
- `libgit2-dev` for `gert`

Error: `renvisoutofsyncwiththelockfile`

Cause: the project library differs from `renv.lock`.

Fix: `renv::status()` to see what, then `renv::restore()` to synchronise.

25.4 knitr / rmarkdown

Error: `pandocdocumentconversionfailedwitherror2`

Cause: YAML front matter is malformed — often an unquoted colon in a title.

Fix: quote the offending string. `title:"Two:peasinapod"`.

Error: `Errorinchunkname:object'xyz'notfound`

Cause: an earlier chunk was skipped or cached stale.

Fix: clear the knitr cache (`knitr::clean_cache()`) and re-render; check chunk evaluation with `eval=TRUE`.

25.5 Git

Error: `fatal:refusingtomergeunrelatedhistories`

Cause: you cloned one repo and then tried to pull from a different initialisation.

Fix: only run `gitpull--allow-unrelated-histories` if you are sure you want to merge.

Error: `Permissiondenied(publickey)`

Cause: your SSH key is not registered with GitHub.

Fix: add your key via Settings → SSH and GPG keys, or switch the remote to HTTPS: `gitremoteset-urloriginhttps://github.com/...`.

Chapter 26

Writing a report

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

APPENDIX · WRITING A REPORT

How to turn an analysis into a document a reviewer, a PI, or a future version of yourself will trust.

26.1 IMRaD

Biomedical writing follows a durable convention: **Introduction, Methods, Results, and Discussion.** The structure is not an aesthetic choice — it mirrors the order in which the reader can evaluate your work. You motivate the question, describe exactly what you did, present the findings, and only then interpret them. Pre-registration and statistical analysis plans make the Methods section possible to write before the Results exist, and the five-step template taught throughout this site feeds directly into the Methods paragraph for every inferential test.

26.2 Effect sizes and intervals

Every result worth reporting is reported with both a point estimate and an interval. A p-value alone is rarely useful: it tells you whether the data are surprising under the null, but not how large the effect is or how much uncertainty you have about its magnitude. A concluding statement that will not embarrass you in five years reads like this:

> Mean arterial pressure was lower in the treatment group than in the placebo group by 6.4 mmHg (95% CI: 2.1 to 10.7; $p = 0.004$; Cohen's $d = 0.42$).

Notice the three things: the direction and magnitude of the effect, the interval, and — optionally — a standardised effect size. Nothing else is needed for the reader to think for themselves.

26.3 Figures

Two rules: **one figure, one idea**, and **no figure without a caption that could stand alone**. A caption tells the reader what the figure shows (*"Forest plot of odds ratios for each arm"*), what the units are, how many observations contribute, and — for anything inferential — what the error bars or bands represent (95% CI? SD? SE of the mean?). Use `theme_minimal()` with the site accent palette where colour is discretionary, and avoid colour-only encoding: pair colour with shape or linetype so colourblind readers can follow along.

26.4 Tables

Tables are rarely the best way to communicate a result that a plot can show, but some results genuinely need them — baseline characteristics, multi-row regression tables, multiple-imputation pooling summaries. Use `gtsummary` for all three. The point of a table is the contrast between rows, so put the thing you want the reader to compare next to itself and round numbers to sensible precision (three significant figures is usually plenty; two for proportions).

26.5 Citing R and packages

R and its packages are freely maintained by volunteers who deserve citations. At the foot of every report, include something like:

> Analysis was performed in R 4.4.1 (R Core Team 2024) using the `t idyverse` (Wickham et al. 2019), `lme4` (Bates et al. 2015), and `rsta narm` (Goodrich et al. 2024) packages.

You can get the correct citation for any package with `citation("pkgname")`. The bibliography it returns is already in BibTeX form.

26.6 Reporting guidelines

Every substantive biomedical study has a reporting guideline. Use them; reviewers will use them whether you do or not.

- **Randomised controlled trial:** CONSORT.
- **Observational study:** STROBE.
- **Diagnostic-accuracy study:** STARD.
- **Prediction-model study:** TRIPOD and TRIPOD-AI.
- **Systematic review:** PRISMA.

- **Animal research:** ARRIVE.

A checklist filled in as you write is much easier than one filled in at the end. Many journals now require a signed checklist as a supplementary file; bookmark the above now.

26.7 Reproducibility

The document you submit should come from a single render of a single source. If the numbers in the text are not the numbers in the `sessionInfo()` block at the foot of the same file, the analysis is not reproducible. Quarto, `renv`, and a version-controlled repository are together enough to get this right; the `targets` package is the next step up for anything large enough to need a dependency graph.

Chapter 27

Cheatsheets

R. Heller · Date: 2026-05-08 · Last changed: 2026-10-30

One-page topical summaries, aggregated here as a printable reference. Each is a self-contained reminder of the syntax, the assumptions, and the diagnostics that go with the methods it covers.

27.1 Foundations

27.1.1 Toolchain, tidy data, and `ggplot2`

27.1.1.1 The research workflow

```
Question→Measurements→Design→Acquisition→Description→Analysis→Interpretation→Validation→Knowledge
```

Biological vs statistical significance: "$p < 0.05$" $\neq$ "effect matters".

27.2 R / RStudio / Quarto / renv

Task	Command
Install packages	`install.packages("pkg")`
Pin project	`renv::init()` / `renv::snapshot()` / `renv::restore()`
Render one file	`quartorenderpath/to/file.qmd`
Preview with live reload	`quartopreview`
Render only slides	`quartorenderfile.qmd--torevealjs`

27.3 Data types and scales

Scale	Example	Statistic
Nominal	sex, site	counts, proportions
Ordinal	pain 0–10	median, IQR, rank tests
Interval	°C	mean, SD
Ratio	kg, time	mean, SD, ratios

Accuracy ≠ precision. Bias shifts the target; variance spreads the shots.

27.4 Tidy data

1. One variable per column.
2. One observation per row.
3. One observational unit per table.

27.5 `dplyr` verbs

```
df |>
  filter(age >= 18) |>
  select(id, age, sbp) |>
  mutate(sbp_z = (sbp - mean(sbp)) / sd(sbp)) |>
  group_by(arm) |>
  summarise(mean_sbp = mean(sbp, na.rm = TRUE), n = n())
```

Joins: `left_join`, `inner_join`, `anti_join` by a key.

Missingness: `is.na()`, `drop_na()`, `tidyr::complete()`.

27.6 `ggplot2` grammar

```
ggplot(data, aes(x, y, colour = group)) +
  geom_point() +
  facet_wrap(~ factor) +
  labs(x = "x label", y = "y label") +
  theme_minimal()
```

Layer order: `data→aes→geom→facet→scales→labels→theme`.

27.7 Plot families

- One variable continuous → **histogram**, **density**.

- Two continuous → **scatter**, sometimes binned / smoothed.
- One categorical + one continuous → **boxplot**, **violin**, **strip**.
- Two categorical → **bar with fill**, **mosaic**.

27.8 Decision rule

- Cannot read the data into a tibble → fix data entry first.
- Tibble reads but has missing or mixed types → clean before modelling.
- Plot is ugly or cluttered → trust the plot, not the summary statistic.

27.9 Common pitfalls

- Running statistics on untyped / partially-missing data.
- Adjusting colour scales instead of simplifying the chart.
- Reporting mean + SD for a skewed variable.
- Forgetting `set.seed()` in any simulation chunk.

27.10 Further reading

- Wickham, *R for Data Science* (2e).
- Posit ggplot2 and dplyr cheatsheets.

27.10.1 Descriptive statistics, Bayes, and distributions

27.10.1.1 Descriptive statistics

Situation	Summary
Roughly symmetric, no outliers	mean ± SD
Skewed or has outliers	median, IQR, range
Categorical	counts and proportions
Grouped	Table 1 via `gtsummary::tbl_summary()`

27.11 Contingency tables

```
table(df$x, df$y)
prop.table(table(df$x, df$y), margin = 1)
```

27.12 Bayes' theorem

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

In odds form: **posterior odds = prior odds × likelihood ratio.**

27.13 Diagnostic testing

Metric	Formula
Sensitivity	TP / (TP + FN)
Specificity	TN / (TN + FP)
PPV	TP / (TP + FP)
NPV	TN / (TN + FN)
LR+	Se / (1 − Sp)
LR−	(1 − Se) / Sp

PPV collapses at low prevalence — even a 95%-accurate test is mostly false positives.

27.14 Discrete distributions

Distribution	Use	R
Bernoulli(p)	single trial	`rbinom(n,1,p)`
Binomial(n, p)	# successes	`dbinom/pbinom/rbinom`
Poisson(λ)	rare events	`dpois/ppois/rpois`
Negative binomial	overdispersed counts	`MASS::rnegbin`, `dnbinom`

Binomial → Poisson as n grows and p shrinks with $np = \lambda$.

27.15 Continuous distributions

Distribution	Use
Normal(μ, σ)	almost everything via the CLT
Student $t(\nu)$	inference about means, small n
$\chi^2(\nu)$	variance, counts, GoF
F(ν_1, ν_2)	variance ratios, ANOVA
Exponential(λ)	time between events, memoryless

R uses the `d/p/q/r` prefix: density, CDF, quantile, random.

27.16 Q-Q plot

```
ggplot(df, aes(sample = x)) + stat_qq() + stat_qq_line()
```

S-shape → heavy tails. U-shape → skew. Plot before the test.

27.17 Decision rule

- Reporting a mean? First check a histogram.
- Testing a proportion? First sketch a 2×2 table.
- Choosing a distribution? Simulate before believing.

27.18 Common pitfalls

- Quoting PPV without disclosing prevalence.
- Assuming normality because $n > 30$.
- Using mean ± SD on skewed data.

27.19 Further reading

- Altman, *Practical Statistics for Medical Research*.
- Gelman et al., *Bayesian Data Analysis*, ch. 1.

27.19.1 Sampling, estimation, and one-sample tests

27.19.1.1 Populations vs samples

- **Parameter** = truth about the population (μ, σ, π).
- **Estimator** = statistic computed on a sample ($\bar{x}$, s, $\hat{p}$).
- **Bias** = $E[\hat{\theta}] - \theta$. **MSE** = bias^2 + variance.

27.20 Central limit theorem

For iid samples with finite variance, as $n \to \infty$: $\bar{X} \sim N(\mu, \sigma/\sqrt{n})$

Holds roughly by $n \approx 30$ for most non-pathological distributions.

27.21 Standard error

$\text{SE}(\bar{X}) = \sigma/\sqrt{n}$, estimated by $s/\sqrt{n}$.

27.22 Bootstrap

```r
B <- 2000
boot_mean <- replicate(B, mean(sample(x, replace = TRUE)))
quantile(boot_mean, c(0.025, 0.975))   # 95% percentile CI
sd(boot_mean)                           # bootstrap SE
```

27.23 Permutation test

```r
obs  <- mean(a) - mean(b)
pool <- c(a, b)
null <- replicate(5000, {
  idx <- sample(seq_along(pool), length(a))
  mean(pool[idx]) - mean(pool[-idx])
})
mean(abs(null) >= abs(obs))   # two-sided p
```

27.24 Maximum likelihood

Given data x and model $f(x;\theta)$:

- $\ell(\theta) = \sum \log f(x_i;\theta)$.
- $\hat{\theta}_{\text{MLE}} = \arg\max_\theta \ell(\theta)$.
- Fisher information $I(\theta)$; asymptotic SE $= 1/\sqrt{I(\hat{\theta})}$.

27.25 One-sample tests (five-step template)

Hypothesis → Visualise → Assumptions → Conduct → Conclude.

Test	R	Assumptions
One-sample t	`t.test(x,mu=0)`	roughly normal or large n
One-proportion	`prop.test(k,n,p=0.5)` or `binom.test`	$np, n(1-p) \geq 5$ for `prop.test`

27.26 Hypothesis-testing vocabulary

- **Type I**: reject true H_0 (probability α).
- **Type II**: accept false H_0 (probability β).
- **Power** $= 1 - \beta$.
- A p-value is **not** the probability that H_0 is true.

27.27 Decision rule

- Want a CI? Bootstrap first, then ask whether a formula applies.
- Want a test? State H_0 and α in writing before running it.
- Want to know if n is big enough? Simulate the CLT for your outcome.

27.28 Common pitfalls

- Reporting SE when you meant SD (or vice versa).
- Using the 95% CI to "prove" there is no effect.
- Running many tests and quoting the smallest p-value.

27.29 Further reading

- Harrell, *Biostatistics for Biomedical Research*, ch. 3–5.
- Efron & Tibshirani, *An Introduction to the Bootstrap*.

27.29.1 Two-group comparisons, effect sizes, and power

27.29.1.1 Choosing a comparison

Design	Continuous outcome	Binary outcome
Two independent groups	`t.test(y~g)` (Welch); Wilcoxon rank-sum	`prop.test` / `fisher.test`
Paired / pre-post	`t.test(pre,post,paired =TRUE)`; Wilcoxon signed-rank	McNemar
>2 groups, continuous	`aov(y~g)`; Kruskal-Wallis	chi-square

27.30 Effect sizes

Statistic	What it reports
Mean difference + 95% CI	primary for continuous
Cohen's d	standardised mean difference
Hedges' g	small-sample correction of d
Risk ratio (RR) / odds ratio (OR)	two proportions
Risk difference	absolute, clinically intuitive

```
effectsize::cohens_d(y ~ g)
```

27.31 Two proportions

```r
prop.test(c(tA, tB), c(nA, nB))         # asymptotic
fisher.test(matrix(c(tA, nA - tA,
                     tB, nB - tB), 2))  # exact, small cells
```

Report RR (or OR) with 95% CI, not just the p-value.

27.32 Correlation

Method	Captures	Assumptions
Pearson	linear association, continuous	bivariate normal, no outliers
Spearman	monotonic association	rank-based, robust
Kendall	concordant/discordant pairs	robust, slow on large data

```r
cor.test(x, y, method = "spearman")
```

27.33 Non-parametric tests

Test	Replaces
Wilcoxon rank-sum (Mann-Whitney)	two-sample t
Wilcoxon signed-rank	paired t
Kruskal-Wallis	one-way ANOVA
Sign test	paired t, when even ranks fail

27.34 Power and sample size

```r
pwr::pwr.t.test(d = 0.5, power = 0.80, sig.level = 0.05,
                type = "two.sample")
```

Family	Function
two-sample t	`pwr.t.test`
two proportions	`pwr.2p.test`, `pwr.2p2n.test`
correlation	`pwr.r.test`
one-way ANOVA	`pwr.anova.test`

Simulation-based power for anything the textbooks skip: `simr::powerSim`.

27.35 Reporting with `gtsummary`

```
library(gtsummary)
trial |>
  tbl_summary(by = arm, statistic = list(all_continuous() ~ "{mean}
   ↪  ({sd})")) |>
  add_p() |>
  add_overall()
```

27.36 Decision rule

- Ask: one-group, two-group, paired, or many-group?
- Check: normal-ish, or should I use a rank-based test?
- Report: point estimate + 95% CI + effect size; p-value last, not first.

27.37 Common pitfalls

- Equal-variance t-test (default in some languages) when variances differ.
- Reporting OR when the audience expects RR (or vice versa).
- Forgetting that the p-value of a paired test depends on the pairing.
- Chaining correlations until one is "significant".

27.38 Further reading

- Altman, Machin et al., *Statistics with Confidence*.
- `gtsummary` docs.

27.39 Regression

27.39.1 Linear regression and diagnostics

27.39.1.1 Correlation vs regression

- Correlation: symmetric; both variables random (**Model II**).
- Regression: asymmetric; predictor fixed, outcome random (**Model I**).
- Use Model II (e.g. `smatr::sma`) when both variables have error.

27.40 Simple linear regression

$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2)$

```
fit <- lm(y ~ x, data = df)
broom::tidy(fit, conf.int = TRUE)
```

```
broom::glance(fit)
```

27.41 Multiple regression

```
fit <- lm(y ~ x1 + x2 + x1:x2, data = df)
```

Concept	Fix
Confounding	adjust by including the confounder
Interaction	include `x1:x2` (or `x1*x2`)
Non-linearity	`splines::ns(x,4)` or `I(x^2)`
Scale sensitivity	centre / standardise predictors

27.42 Diagnostics

```
plot(fit)                    # base: 4 diagnostic plots
performance::check_model(fit)   # tidy overview
car::vif(fit)                # multicollinearity
```

Plot	What it flags
Residuals vs fitted	non-linearity, non-constant variance
Normal Q-Q	non-normal residuals
Scale-location	heteroscedasticity
Leverage / Cook's	influential outliers

VIF $> 5 \rightarrow$ investigate collinearity; $> 10 \rightarrow$ fix it.

27.43 Robust / weighted

```
MASS::rlm(y ~ x, data = df)          # robust to outliers
lm(y ~ x, weights = 1 / var_i)        # weighted LS

# Heteroscedasticity-consistent SEs
library(sandwich); library(lmtest)
coeftest(fit, vcov = vcovHC(fit, type = "HC3"))
```

27.44 Decision rule

- Residual plot ugly? Transform, add terms, or switch link.

- VIF explodes? Drop one of the collinear predictors or combine them.
- Outlier with high Cook's distance? Investigate before deleting.
- Variance depends on X? HC3 SEs or weighted LS.

27.45 Common pitfalls

- Interpreting β_1 as "effect of X" under confounding.
- Reporting R^2 on in-sample data as if it were generalisable.
- Including an interaction but only reporting main effects.
- Centring categorical predictors (don't — use reference coding).

27.46 Further reading

- Harrell, *Regression Modeling Strategies*, ch. 4–5.
- Fox, *Applied Regression Analysis*.

27.46.1 ANOVA, mixed models, and non-linear regression

27.46.1.1 One-way ANOVA

```
aov(y ~ group, data = df) |> summary()
```

ANOVA is a linear model with a categorical predictor. The F-test compares between-group to within-group variance.

27.47 Contrasts with `emmeans`

```
library(emmeans)
emm <- emmeans(fit, ~ group)
pairs(emm, adjust = "tukey")   # all pairwise
contrast(emm, list(TrtVsCtrl = c(-1, 1, 1, 1) / 3))
```

Pre-specify contrasts before looking at the data; correct for multiplicity.

27.48 Two-way / factorial ANOVA

```
aov(y ~ A * B, data = df) |> summary()
emmip(fit, A ~ B)   # interaction plot
```

Interaction means "effect of A differs by level of B". Report the interaction first; main effects are conditional.

27.49 Repeated measures / blocking

- RCBD: `aov(y~treatment+Error(block))`.
- Repeated measures: move to a mixed model.

```
library(lme4); library(lmerTest)
lmer(y ~ treatment + time + (1 | subject), data = df)
```

27.50 GAMs — smooth non-linear terms

```
library(mgcv); library(gratia)
fit <- gam(y ~ s(x, k = 10) + z, data = df)
summary(fit)      # edf tells you how "wiggly"
draw(fit)         # smooth + CI
```

edf $\approx 1 \rightarrow$ nearly linear; $> 4 \rightarrow$ clearly non-linear.

27.51 Non-linear regression (`nls`)

```
# Michaelis-Menten: y = Vmax * x / (K + x)
fit <- nls(y ~ Vmax * x / (K + x),
           data = df, start = list(Vmax = 1, K = 1))
```

Start values matter. If it fails, plot first to guess reasonable starts.

27.52 Decision rule

- Categorical predictor, > 2 levels $\rightarrow$ ANOVA + contrasts.
- Factorial design $\rightarrow$ include interaction, report it first.
- Effect obviously curved $\rightarrow$ GAM with spline; else try `nls`.
- Repeated measures $\rightarrow$ mixed model, not repeated-measures ANOVA.

27.53 Common pitfalls

- Tukey HSD without pre-specified contrasts of interest.
- ANOVA $p < 0.05$ reported alone — without naming *which* groups differ.
- Forcing a GAM onto monotonic data that `nls` fits cleanly.
- Ignoring the random effect in clustered designs (pseudoreplication).

27.54 Further reading

- Wood, *Generalized Additive Models*, 2e.

- `emmeans` vignette.

27.54.1 Generalised linear models and prediction evaluation

27.54.1.1 GLM link functions

Outcome	Family	Link	R
Continuous	gaussian	identity	`lm()`
Binary	binomial	logit	`glm(y~x,family=binomial)`
Count	poisson	log	`glm(y~x,family=poisson,offset=log(t))`
Overdispersed count	neg. binomial	log	`MASS::glm.nb`
Ordinal	cumulative	logit	`MASS::polr`
Nominal	multinomial	logit	`nnet::multinom`

27.55 Logistic regression

```r
fit <- glm(y ~ x1 + x2, data = df, family = binomial)
exp(coef(fit))                       # odds ratios
exp(confint(fit))                    # 95% CI on OR
predict(fit, newdata = nd, type = "response")
```

- Check for perfect separation (huge SEs).
- Interpret ORs cautiously; RR is more intuitive for the audience.

27.56 ANCOVA in an RCT

Adjust for baseline; **do not** analyse the change score.

```r
lm(y_followup ~ arm + y_baseline, data = trial)
```

More efficient than simple t-test on change scores when baseline and follow-up are correlated.

27.57 Poisson / negative binomial

```r
glm(cases ~ x + offset(log(person_years)),
    family = poisson, data = df)
MASS::glm.nb(cases ~ x + offset(log(person_years)), data = df)
```

Check dispersion: `sum(residuals(fit,type="pearson")^2)/df.residual`. If > 1.5, switch to NB.

27.58 Evaluation — calibration + discrimination

Metric	Means	R
Calibration plot	predicted vs observed	`rms::val.prob`, manual bin
ROC / AUC	rank ordering	`pROC::roc(y,phat)`
Brier score	overall accuracy	`mean((phat-y)^2)`
Calibration slope / intercept	systematic bias	from logistic recalibration

```r
library(pROC)
roc_obj <- roc(y, phat)
auc(roc_obj); ci.auc(roc_obj)
plot(roc_obj)
```

27.59 Decision rule

- Binary outcome → logistic; report OR + 95% CI.
- Count outcome → Poisson; check overdispersion; NB if needed.
- Trial analysis → ANCOVA, not change score.
- Prediction model → calibration curve first, ROC second, decision curves third.

27.60 Common pitfalls

- Quoting AUC without calibration (a discriminating but miscalibrated model is dangerous).
- Ignoring offsets in count data.
- Using ordinal logit when the proportional-odds assumption fails.
- Presenting logistic regression coefficients on the log-odds scale without OR.

27.61 Further reading

- Harrell, *RMS*, ch. 10–12.
- Steyerberg, *Clinical Prediction Models*.

27.61.1 Agreement, survival primer, and reporting

27.61.1.1 Don't dichotomise

Cutting a continuous variable at the median loses $\approx 37\%$ of the information for a linear association. Keep predictors continuous; model non-linearity with a spline if needed.

27.62 Change scores vs ANCOVA

- Change scores suffer from **regression to the mean**.
- ANCOVA (adjust for baseline) is the efficient analysis in an RCT.
- In observational studies with unequal baselines, both approaches answer subtly different questions — state which.

27.63 Agreement & reliability

Statistic	Use	R
Cohen's κ	two raters, categorical	`irr::kappa2(cbind(r1,r2))`
Weighted κ	ordinal categorical	`irr::kappa2(...,weight="squared")`
ICC(2,1) / ICC(3,1)	continuous	`psych::ICC(df)`
Bland-Altman	two methods, continuous	plot $\bar{xy}$ vs $y - x$

```r
mean_diff <- mean(y - x)
loa <- mean_diff + c(-1.96, 1.96) * sd(y - x)
ggplot(df, aes((x + y)/2, y - x)) + geom_point() +
  geom_hline(yintercept = c(mean_diff, loa), linetype = 2)
```

27.64 Survival primer

Concept	R
Kaplan-Meier	`survfit(Surv(time,event)~g)` + `ggsurvfit::ggsurvfit`
Log-rank test	`survdiff(Surv(time,event)~g)`
Cox PH model	`coxph(Surv(time,event)~x1+x2)`
PH check	`cox.zph(fit)`

- Report a hazard ratio with 95% CI.

- PH violated? → time-varying coefficient or stratify.
- Interpret HR only after checking for non-informative censoring.

27.65 Decision curves, NRI, IDI

- **Decision curve**: net benefit vs threshold probability — dominates "treat all" and "treat none" when useful.
- **NRI**: how many events move up / non-events move down in risk.
- **IDI**: mean change in predicted probability by class.
- Report decision curve first; NRI/IDI as secondary.

27.66 Explanation vs prediction (Shmueli)

Explanatory	Predictive
Goal: inference about β	Goal: minimise out-of-sample loss
Tools: ANOVA, diagnostics, intervals	Tools: CV, regularisation, ensembles
Metric: interval coverage, bias	Metric: RMSE, AUC, Brier on hold-out

27.67 Reporting guidelines

Design	Guideline
Randomised trial	CONSORT
Observational	STROBE
Diagnostic-accuracy	STARD
Prediction model (incl. AI)	TRIPOD / TRIPOD-AI
Systematic review	PRISMA
Animal research	ARRIVE

27.68 Decision rule

- Continuous predictor + continuous outcome → no median splits.
- Two-group with repeated measurement → ANCOVA.
- New rater / device → Bland-Altman + ICC.
- Time-to-event → KM + Cox, always check PH.
- Prediction model → TRIPOD checklist at submission.

27.69 Common pitfalls

- Reporting κ on near-constant outcomes (high % agreement, low κ).

- Quoting median survival that is never reached.
- Forgetting to mention PH assumption checking.
- Claiming a "prediction model" built on the same data used for evaluation.

27.70 Further reading

- Harrell, *BBR*, ch. 17–18.
- Royston & Altman, *Prognosis research*.

27.71 Design & Causal Inference

27.71.1 Study designs and power

27.71.1.1 Observational designs

Design	Strengths	Weaknesses
Cohort	temporality, incidence	expensive, loss to follow-up
Case-control	rare outcomes, cheap	recall + selection bias
Cross-sectional	prevalence, screening	no temporality
Case-crossover	within-person comparison	for transient exposures only

STROBE checklist: https://www.strobe-statement.org/

27.72 Trial designs

Type	When
Parallel-group	most common; independent arms
Crossover	stable chronic condition, carry-over washable
Cluster	intervention at group level (schools, clinics)
Factorial	two or more interventions, test interactions
Adaptive	pre-specified modifications based on interim data
Non-inferiority	new treatment not worse than control by margin Δ
Equivalence	two-sided non-inferiority

CONSORT for RCTs: https://www.consort-statement.org/

27.73 Bench / translational

- **Blocking** reduces nuisance variation (plate, day, operator).
- **Factorial** tests interactions efficiently.
- **Split-plot** handles two levels of randomisation.
- **Pseudoreplication**: technical replicates ≠ biological replicates.

27.74 Power — closed form

```
library(pwr)
pwr.t.test(d = 0.5, power = 0.80, sig.level = 0.05,
           type = "two.sample")
pwr.2p.test(h = ES.h(0.3, 0.2), power = 0.80)
pwr.r.test(r = 0.3, power = 0.80)
pwr.anova.test(k = 4, f = 0.25, power = 0.80)
```

Effect-size conventions (Cohen): small $d = 0.2$, medium 0.5, large 0.8.

27.75 Power — simulation

```
library(simr)
# Build a pilot model, increase N, or tweak fixef, then:
ps <- powerSim(model, nsim = 500, test = fixed("arm"))
ps
```

Simulation wins for any design the textbook skips: mixed models, adaptive rules, non-standard outcomes.

27.76 Decision rule

- Randomise if you can. If not, draw a DAG and name the biases.
- Power calculation before the protocol freeze, not after data collection.
- Cluster randomisation → design effect $1 + (\bar{m} - 1)\rho$; inflate N.
- Bench experiments → treat batch, plate, and operator as random effects.

27.77 Common pitfalls

- Planning a cluster RCT without inflating N for design effect.
- Borrowing a pilot effect size without acknowledging noise.
- Running a non-inferiority trial as if it were superiority.
- Treating technical replicates as if they were biological.

27.78 Further reading

- Matthews, *Introduction to Randomized Controlled Clinical Trials*.
- Chow & Liu, *Design and Analysis of Clinical Trials*.

27.78.1 Missing data, mixed models, and longitudinal

27.78.1.1 Missingness mechanisms

Mechanism	Definition	Complete-case analysis?
MCAR	missingness independent of data	unbiased but inefficient
MAR	missingness depends on **observed** data	biased, imputation helps
MNAR	missingness depends on **unobserved** values	imputation under model only

Test is impossible without assumptions — reason about the mechanism.

27.79 Multiple imputation with `mice`

```r
library(mice)
imp <- mice(df, m = 20, seed = 42, printFlag = FALSE)
fit <- with(imp, lm(y ~ x1 + x2))
pooled <- pool(fit)
summary(pooled, conf.int = TRUE)
```

Rule of thumb: $m \geq 100 \times$ fraction of missing information, but 20 is a fine starting point.

27.80 Linear mixed models

```r
library(lme4); library(lmerTest)
lmer(y ~ treatment * time + (1 | subject), data = df)
lmer(y ~ treatment * time + (time | subject), data = df)  # random slope
```

- Random intercept: each cluster has its own baseline.
- Random slope: each cluster has its own trajectory.
- ICC $\rho = \sigma_u^2/(\sigma_u^2 + \sigma_e^2)$.

REML for variance components; ML only for likelihood-ratio tests on fixed effects.

27.81 GLMMs & GEE

```r
library(lme4)
glmer(y ~ x + (1 | cluster), data = df, family = binomial)

library(geepack)
geeglm(y ~ x, id = cluster, data = df,
       family = binomial, corstr = "exchangeable")
```

- GLMM: subject-specific effects, good for prediction.
- GEE: population-average effects, robust to working-correlation misspecification (when combined with robust SEs).

27.82 Time series basics

```r
library(forecast)
ts_obj <- ts(x, frequency = 12)
decompose(ts_obj) |> plot()
auto.arima(ts_obj)

library(changepoint)
cpt.mean(x, method = "PELT") |> plot()
```

27.83 Decision rule

- Missing data $> 5\%$ $\rightarrow$ at least sensitivity analysis; prefer MI.
- Repeated measurements $\rightarrow$ mixed model, not repeated-measures ANOVA.
- Binary outcome with clusters $\rightarrow$ GLMM (if subject effects matter) or GEE (if marginal effect is the estimand).
- Time series with trend / seasonality $\rightarrow$ decompose before modelling.

27.84 Common pitfalls

- Last-observation-carried-forward; usually biased.
- Imputing after transformation rather than in the right scale.
- Ignoring random effects when clusters are small in number (degrees of freedom correction).
- Quoting GEE coefficients as if they were subject-specific.

27.85 Further reading

- van Buuren, *Flexible Imputation of Missing Data*.

- Fitzmaurice, Laird, Ware, *Applied Longitudinal Analysis*.

27.85.1 Causal inference methods

27.85.1.1 Time-varying covariates / landmarking

- Counting-process data: split follow-up into intervals.
- Landmark analysis: define a landmark time t^*, condition on surviving to t^*, classify by status at t^*.

```
library(survival)
coxph(Surv(tstart, tstop, event) ~ x, data = long)
```

- Immortal-time bias: exposed group can only be exposed if they live long enough to be exposed → spurious survival benefit.

27.86 Competing risks

```
library(tidycmprsk); library(ggsurvfit)
cif <- cuminc(Surv(time, event_f) ~ x, data = df)
ggsurvfit::ggcuminc(cif, outcome = "event1")

# Fine-Gray subdistribution hazard for event1
crr(Surv(time, event_f) ~ x, data = df, failcode = "event1")
```

- Cause-specific HR answers "given you are event-free, what is the hazard for event k?"
- Subdistribution HR answers "what is the hazard for event k in the whole cohort, treating competing events as obstacles?"

27.87 DAGs

```
library(dagitty); library(ggdag)
dag <- dagitty("dag { X -> Y ; C -> X ; C -> Y }")
adjustmentSets(dag, exposure = "X", outcome = "Y")
impliedConditionalIndependencies(dag)
```

- Adjust for confounders, **not** colliders.
- Adjusting for a mediator estimates the direct effect only.

27.88 Propensity scores / IPTW

```
library(MatchIt); library(cobalt)
m <- matchit(treat ~ x1 + x2, data = df, method = "nearest")
```

```r
love.plot(m)   # check balance: SMD < 0.1 is the goal
fit <- lm(y ~ treat, data = match.data(m))
```

IPTW: weight by $1/\hat{e}_i$ for treated, $1/(1-\hat{e}_i)$ for untreated. Stabilised weights stay closer to 1; trim extremes to avoid outliers.

27.89 G-methods, IV, DiD, RDD

Method	Identifying assumption
G-formula / IPTW	positivity + no unmeasured confounding
Instrumental variables	exclusion restriction + relevance
Difference-in-differences	parallel trends
Regression discontinuity	continuity at the cutoff

27.90 Heterogeneous treatment effects

- Causal forest (`grf::causal_forest`) or meta-learners (S-, T-, X-).
- Report conditional ATEs with adjusted intervals; avoid spurious subgroup p-values.

27.91 Decision rule

- Treatment assigned over time → time-varying Cox.
- Multiple competing events → cumulative incidence, Fine-Gray.
- Observational causal question → DAG first, choose identification strategy second.
- Want the whole distribution of treatment effect? → causal forest.

27.92 Common pitfalls

- Adjusting for a post-treatment variable (induces collider bias).
- Quoting HR from a Cox model that violates PH.
- Matching but never checking balance.
- Calling IV estimates "generalisable" when they are local to compliers.

27.93 Further reading

- Hernán & Robins, *Causal Inference: What If*.
- Therneau & Grambsch, *Modeling Survival Data*.

27.93.1 Synthesis and pre-registration

27.93.1.1 Systematic review workflow

1. PICO: Population, Intervention, Comparator, Outcome.
2. Register protocol on PROSPERO.
3. Build search across ≥ 2 databases; peer-review with a librarian.
4. Two-screener title/abstract and full-text.
5. Extract, appraise (RoB 2 / ROBINS-I), synthesise.
6. Report with the PRISMA 2020 checklist and flow diagram.

27.94 Meta-analysis

```r
library(metafor)
dat <- escalc(measure = "RR",
              ai = tpos, bi = tneg, ci = cpos, di = cneg,
              data = studies)
fit <- rma(yi, vi, data = dat, method = "REML")
fit
forest(fit, header = TRUE)
funnel(fit); regtest(fit)
```

Statistic	Meaning
τ^2	between-study variance
I^2	% of total variance due to heterogeneity
Egger's test	funnel-plot asymmetry (small-study effects)

Fixed-effect only when heterogeneity is effectively zero. Default to random effects.

27.95 Network meta-analysis

```r
library(netmeta)
net <- netmeta(TE, seTE, treat1, treat2, studlab,
               data = dat, sm = "MD", random = TRUE)
netgraph(net)
netrank(net, small.values = "good")    # SUCRA ranking
decomp.design(net)                     # inconsistency
```

Transitivity + consistency are the core assumptions.

27.96 Infectious disease — SIR / SEIR

```r
library(deSolve)
sir <- function(t, y, p) with(as.list(c(y, p)), {
  N <- S + I + R
  list(c(-beta * S * I / N,
          beta * S * I / N - gamma * I,
          gamma * I))
})
out <- ode(c(S = 999, I = 1, R = 0),
           times = 0:120,
           func = sir,
           parms = c(beta = 0.3, gamma = 0.1))
```

$R_0 = \beta/\gamma$. Herd-immunity threshold $\approx 1 - 1/R_0$.

27.97 Pre-registration and SAP

A pre-registration specifies, before data analysis:

- Primary outcome + exact estimand.
- Analysis model (fixed terms, adjustments, interactions).
- Handling of missing data.
- Multiplicity control.
- Sample-size justification.

OSF and ClinicalTrials.gov are the two common registries.

27.98 Decision rule

- More than a handful of studies on a question → plan a systematic review.
- Multiple comparators of interest → network meta-analysis.
- Outbreak modelling → start with SIR, extend only as data justify.
- Confirmatory analysis → pre-registered; exploratory analyses are labelled as such in the manuscript.

27.99 Common pitfalls

- Declaring a review “systematic” without a pre-registered protocol.
- Pooling studies with fundamentally different populations / outcomes.
- Interpreting SUCRA rankings as if they were certain.
- Calling a compartmental model a “forecast” without uncertainty.

27.100 Further reading

- Higgins et al., *Cochrane Handbook for Systematic Reviews*.
- Keeling & Rohani, *Modeling Infectious Diseases in Humans and Animals*.
- Nosek et al. (2018), *The preregistration revolution*.

27.101 Machine Learning & High-Dimensional Data

27.101.1 Cross-validation, regularisation, and unsupervised

27.101.1.1 Cross-validation

Scheme	Use
k-fold (k = 5 or 10)	baseline, fast
Repeated k-fold	reduces Monte Carlo noise
Leave-one-out	small n, high variance
Stratified k-fold	imbalanced class labels
Grouped / blocked	correlated units (patients, sites)
Nested	honest evaluation of tuned models

Nested CV is the only defensible way to estimate generalisation error when you are also tuning hyperparameters.

27.102 Regularisation

```r
library(glmnet)
x <- model.matrix(y ~ . - 1, df); y <- df$y
fit <- cv.glmnet(x, y, alpha = 1)     # alpha = 1 = lasso; 0 = ridge
coef(fit, s = "lambda.1se")           # sparse solution
plot(fit)                              # CV curve
```

- `lambda.min`: empirical minimum of CV error.
- `lambda.1se`: more regularised, simpler, within 1 SE of min.
- Elastic net: `alpha` between 0 and 1 blends ridge and lasso.

27.103 Multivariate workhorses

Method	Purpose
PCA	variance → low-d summary; `prcomp(x,scale.=TRUE)`
FA	latent-variable modelling; `psych::fa`
CCA	correlation structure between two sets
LDA	supervised low-d projection; `MASS::lda`

```
pc <- prcomp(x, scale. = TRUE)
summary(pc)   # proportion of variance per PC
```

Scale before PCA when variables are on different units.

27.104 Clustering

```
km <- kmeans(scale(x), centers = 4, nstart = 25)
hc <- hclust(dist(scale(x)), method = "ward.D2")
library(mclust); bic <- mclustBIC(x)      # model-based, BIC-selected
```

- Choose k by silhouette, gap statistic, or BIC.
- Scale first. K-means assumes round clusters of similar size.

27.105 Non-linear embeddings

```
library(uwot)
umap_xy <- umap(scale(x), n_neighbors = 15, min_dist = 0.1)

library(Rtsne)
ts <- Rtsne(scale(x), perplexity = 30)$Y
```

Embeddings are *visualisations*, not distances you should input into downstream stats. Global geometry is often distorted.

27.106 Decision rule

- Lots of predictors, small n → lasso; report `lambda.1se`.
- Exploring structure → PCA + UMAP on scaled data.
- Clustering for biology → model-based + sensitivity to k.
- Tuning anything → nested CV.

27.107 Common pitfalls

- Selecting features on full data, then estimating error by k-fold.

- PCA before scaling when features are on different units.
- Interpreting UMAP distances as real distances.
- Using AUC on rank-ordered hold-out error as if it were CV error.

27.108 Further reading

- Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning*.
- James et al., *ISLR2*.

27.108.1 Trees, boosting, and interpretability

27.108.1.1 Tree-based models

Model	R	Strengths
CART	`rpart::rpart`	interpretable single tree
Random forest	`ranger::ranger`	robust, OOB error, variable importance
XGBoost	`xgboost::xgb.train`	top tabular performance, careful tuning
LightGBM	`lightgbm::lgb.train`	fast on large data

```r
library(ranger)
fit <- ranger(y ~ ., data = df, importance = "permutation",
              num.trees = 500, mtry = floor(sqrt(ncol(df) - 1)))
fit$prediction.error   # OOB error
```

27.109 Interpretability

```r
library(DALEX); library(iml)

ex  <- explain(fit, data = df |> select(-y), y = df$y)
ip  <- model_parts(ex)           # permutation importance
pd  <- model_profile(ex, variables = "x1")   # partial dependence
sh  <- Shapley$new(Predictor$new(fit, data = df, y = df$y),
                   x.interest = df[1, ])     # SHAP
```

PDP assumes feature independence; if features are correlated, use ALE plots.

27.110 Tabular neural networks with `torch`

```
# sketch only — typically in a loop on GPU
library(torch)
nn <- nn_sequential(
  nn_linear(p, 64), nn_relu(),
  nn_linear(64, 1)
)
opt <- optim_adam(nn$parameters)
```

On small biomedical tabular data, boosted trees usually beat NN. Use NN when you need end-to-end training with images, sequences, or text.

27.111 tidymodels pipeline

```
library(tidymodels)
rec  <- recipe(y ~ ., data = df) |>
  step_normalize(all_numeric_predictors()) |>
  step_dummy(all_nominal_predictors())

mod  <- rand_forest(mtry = tune(), trees = 500) |>
  set_engine("ranger") |> set_mode("classification")

wf   <- workflow() |> add_recipe(rec) |> add_model(mod)
cv   <- vfold_cv(df, v = 5, strata = y)
tuned <- tune_grid(wf, cv, grid = 20)
collect_metrics(tuned)
```

27.112 Decision rule

- Tabular, < 100k rows $\rightarrow$ gradient-boosted trees.
- Need explanations $\rightarrow$ SHAP + PDP, validate with held-out.
- Images / sequences $\rightarrow$ CNN / transformer, use `torch`.
- Reproducibility $\rightarrow$ `tidymodels` pipeline + fixed seed + locked recipe.

27.113 Common pitfalls

- Reporting feature importance from a model trained on the full data.
- Interpreting PDPs when features are strongly correlated.
- Claiming NN superiority without CV on the same split as a tree model.
- Forgetting to set a seed for any tune / split operation.

27.114 Further reading

- Biecek & Burzykowski, *Explanatory Model Analysis*.
- Molnar, *Interpretable Machine Learning*.

27.114.1 Bayesian and survival ML

27.114.1.1 Bayesian thinking

$$\underbrace{p(\theta \mid y)}_{\text{posterior}} \propto \underbrace{p(y \mid \theta)}_{\text{likelihood}} \underbrace{p(\theta)}_{\text{prior}}$$

- Priors are part of the model — specify and defend them.
- Posterior summaries: mean, median, 95% credible interval.
- No p-values; use posterior probability of direction, Bayes factors, or LOO.

27.115 `brms` / Stan

```
library(brms)
fit <- brm(y ~ x + (1 | group), data = df,
           prior = c(prior(normal(0, 1), class = "b"),
                     prior(exponential(1), class = "sd")),
           chains = 4, iter = 2000, seed = 42)

summary(fit)
loo(fit)               # leave-one-out cross-validation
pp_check(fit)           # posterior predictive checks
```

Prior predictive check before fitting: simulate y from the prior and confirm the implications are plausible.

27.116 Biomarker statistics

Question	Statistic
Does a biomarker classify?	AUC, cut-point (Youden's index)
Does it add over an existing model?	ΔAUC, NRI, IDI, decision curves
Does it move prognosis?	Calibration plus discrimination
Is a cut-off reproducible?	Bootstrap CI on the cut-off

```
pROC::roc(y, biomarker) |>
  pROC::coords("best", best.method = "youden")
```

27.117 Survival ML

Model	R
Random survival forest	`randomForestSRC::rfsrc(Surv(...)~., data)`
Gradient-boosted Cox	`xgboost` with `objective="survival:cox"`
DeepSurv (conceptual)	`torch` with partial-likelihood loss

Evaluate with **time-dependent AUC**, **integrated Brier score**, and **IPA** (index of prediction accuracy = 1 − Brier(model) / Brier(null)).

```
library(timeROC)
r <- timeROC(T = df$time, delta = df$event,
             marker = pred, cause = 1, times = c(365, 730))
```

27.118 External validation

- Never trust the apparent performance on training data.
- Minimum: split-sample or CV on internal data.
- Better: external validation in a second cohort.
- Report calibration slope / intercept, discrimination, and decision curves.

27.119 Decision rule

- Rare outcome with need for uncertainty → Bayesian, not bootstrap of MLE.
- New biomarker → NRI + decision curve, not just ΔAUC.
- Survival prediction → time-dependent Brier / IPA, not global AUC.
- Any clinical claim → external validation before publication.

27.120 Common pitfalls

- Reporting a posterior with default flat priors on unscaled predictors.
- Selecting the Youden cut-off on the full data and using the same data to evaluate sensitivity / specificity.
- Quoting C-statistic at a single time point and calling it survival ML.
- Publishing a prediction model without TRIPOD-compliant reporting.

27.121 Further reading

- Gelman et al., *Bayesian Data Analysis*, 3e.

- Royston & Altman, *External validation of a Cox prognostic model.*

27.121.1 Omics, FDR, and TRIPOD-AI

27.121.1.1 Bulk RNA-seq

```
library(DESeq2)
dds <- DESeqDataSetFromMatrix(counts, coldata, ~ condition)
dds <- DESeq(dds)
res <- results(dds, contrast = c("condition", "B", "A"))
```

```
library(edgeR)
dge <- DGEList(counts, group = coldata$condition)
dge <- calcNormFactors(dge)
design <- model.matrix(~ condition, coldata)
fit <- glmQLFit(estimateDisp(dge, design), design)
qlf <- glmQLFTest(fit, coef = 2)
topTags(qlf)
```

Package	Distribution	Notes
DESeq2	negative binomial	shrinks LFCs; standard for small cohorts
edgeR	negative binomial	QL-F test has good type-I control
limma-voom	mean-variance weights	fast on very large studies

27.122 Enrichment

```
library(fgsea)
fgsea(pathways, stats = stat_vector, nperm = 10000) |>
  arrange(padj)

# Over-representation with a hypergeometric test
phyper(k - 1, K, N - K, n, lower.tail = FALSE)
```

Always pre-rank by a *signed* statistic (log-FC × -log10 p), not by raw p alone.

27.123 Single-cell RNA-seq (Seurat pattern)

```
library(Seurat)
so <- CreateSeuratObject(counts) |>
```

```
  NormalizeData() |>
  FindVariableFeatures() |>
  ScaleData() |>
  RunPCA() |>
  FindNeighbors() |>
  FindClusters(resolution = 0.4) |>
  RunUMAP(dims = 1:20)
DimPlot(so)
```

27.124 FDR, knockoffs, replication

Correction	When
Bonferroni	few planned tests, strict control
Benjamini-Hochberg	many tests, accept proportion of false discoveries
Knockoffs	controlled variable selection with FDR
Permutation / SEA	small n, non-standard models

```
p.adjust(pvals, method = "BH")
```

Replication crisis shorthand: pre-register + report effect sizes + share data. Power, not p-values.

27.125 TRIPOD-AI + fairness

- TRIPOD-AI extends TRIPOD for prediction models built with ML.
- Report: data provenance, training/evaluation splits, subgroup performance, calibration, external validation.
- Fairness: stratify AUC / calibration by sex, ancestry, age bands. Disparities in calibration are often larger than in AUC.

27.126 Reproducibility at scale with `targets`

```
# _targets.R
library(targets)
tar_option_set(packages = c("tidyverse"))
list(
  tar_target(raw, read_csv("data/raw.csv")),
  tar_target(clean, clean_data(raw)),
```

```
  tar_target(fit, fit_model(clean)),
  tar_target(report, quarto::quarto_render("report.qmd"))
)
```

```
Rscript -e 'targets::tar_make()'
```

27.127 Decision rule

- Bulk DE → DESeq2 for small samples, edgeR/limma for large.
- Many hypotheses → BH by default; knockoffs for variable selection.
- scRNA-seq → Seurat pipeline with batch correction + cluster annotation.
- ML prediction model → TRIPOD-AI + subgroup analysis.
- Any multi-step analysis → `targets` pipeline.

27.128 Common pitfalls

- Running DE on a pathway-level aggregate rather than gene-level.
- Cherry-picking a pathway database post-hoc.
- Over-clustering scRNA-seq data until "known" groups appear.
- Calling an ML model "fair" after checking only overall AUC.
- Submitting a paper without a `sessionInfo()` or a lock file.

27.129 Further reading

- Love, Huber, Anders (2014), *Moderated estimation of fold change…*
- Collins et al. (2024), *TRIPOD-AI.*
- Peng, *Reproducible Research with R.*

Colophon

R. Heller · Date: 2026-05-11 · Last changed: 2026-10-30

```
#> Built with:
#>   bookdown:    0.48
#>   rmarkdown:   2.32
#>   knitr:       1.52
#>   R:           R version 4.6.2 (2026-10-29)
#>   Build date:  2026-11-14
#>   Commit:      82efa03
```

LLM assistance (self-hosted Mistral Le Chat via Ollama/`ollamar`, plus GitHub Copilot in RStudio) was used for copyediting, formatting, and scaffolding tasks only — see Acknowledgments.

```
- Session info
 ↪  ---------------------------------------------------------------
 setting  value
 version  R version 4.6.2 (2026-10-29)
 os       Ubuntu 26.04.1 LTS
 system   x86_64, linux-gnu
 ui       X11
 language (EN)
 collate  C.UTF-8
 ctype    C.UTF-8
 tz       UTC
 date     2026-11-14
 pandoc   3.8.3 @ /opt/hostedtoolcache/pandoc/3.8.3/x64/ (via rmarkdown)
 quarto   NA

------------------------------------------------------------------------
 ↪  ------
```

Citing this Guide

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-09

The suggested citation is:

> **Heller, R.** (2026). *Methods in R: A Topic Reference for Biomedical Research and Peer Review* (Version 1.0.0). Self-published via GitHub Pages. https://cttir.github.io/methods-in-r/.

Download the reference as BibTeX or .ris.

References

R. Heller · Date: 2026-05-11 · Last changed: 2026-11-10

This library collects the general references and the software citations for this edition.

Agresti, A. (2010). *Analysis of Ordinal Categorical Data*. Wiley, 2 edition.

Agresti, A. (2013). *Categorical Data Analysis*. Wiley, 3 edition.

Agresti, A. and Coull, B. A. (1998). Approximate is better than "exact" for interval estimation of binomial proportions. *The American Statistician*, 52(2):119–126.

Allaire, J., Xie, Y., Dervieux, C., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., and Iannone, R. (2026). *rmarkdown: Dynamic Documents for R*. R package version 2.32.

Bakeman, R. (2005). Recommended effect size statistics for repeated measures designs. *Behavior Research Methods*, 37(3):379–384.

Bishara, A. J. and Hittner, J. B. (2012). Testing the significance of a correlation with nonnormal data. *Psychological Methods*, 17(3):399–417.

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum, 2 edition.

Conover, W. J., Johnson, M. E., and Johnson, M. M. (1981). A comparative study of tests for homogeneity of variances. *Technometrics*, 23(4):351–361.

Delacre, M., Lakens, D., and Leys, C. (2017). Why psychologists should by default use Welch's t-test. *International Review of Social Psychology*, 30(1):92–101.

Dinno, A. (2015). Nonparametric pairwise multiple comparisons in independent groups using Dunn's test. *The Stata Journal*, 15(1):292–300.

Divine, G., Norton, H. J., Hunt, R., and Dienemann, J. (2013). A review of analysis and sample size calculation considerations for Wilcoxon tests. *Anesthesia and Analgesia*, 117(3):699–710.

Fabrigar, L. R., Wegener, D. T., MacCallum, R. C., and Strahan, E. J. (1999). Evaluating the use of exploratory factor analysis in psychological research. *Psychological Methods*, 4(3):272–299.

Faraway, J. J. (2015). *Linear Models with R*. Chapman and Hall/CRC, 2 edition.

Field, A. (2018). *Discovering Statistics Using IBM SPSS Statistics*. SAGE, 5 edition.

Fox, J. and Weisberg, S. (2019). *An R Companion to Applied Regression*. SAGE, 3 edition.

Ghasemi, A. and Zahediasl, S. (2012). Normality tests for statistical analysis: A guide for non-statisticians. *International Journal of Endocrinology and Metabolism*, 10(2):486–489.

Gueorguieva, R. and Krystal, J. H. (2004). Move over ANOVA: Progress in analyzing repeated-measures data. *Archives of General Psychiatry*, 61(3):310–317.

Harrell, F. E. (2015). *Regression Modeling Strategies*. Springer, 2 edition.

Hollander, M., Wolfe, D. A., and Chicken, E. (2014). *Nonparametric Statistical Methods*. Wiley, 3 edition.

Hosmer, D. W., Lemeshow, S., and Sturdivant, R. X. (2013). *Applied Logistic Regression*. Wiley, 3 edition.

Kassambara, A. (2023). *rstatix: Pipe-Friendly Framework for Basic Statistical Tests*.

Kaufman, L. and Rousseeuw, P. J. (2009). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley.

Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30(1–2):81–93.

Kerby, D. S. (2014). The simple difference formula: An approach to teaching nonparametric correlation. *Comprehensive Psychology*, 3.

Maxwell, S. E., Delaney, H. D., and Kelley, K. (2017). *Designing Experiments and Analyzing Data*. Routledge, 3 edition.

Newson, R. (2002). Parameters behind "nonparametric" statistics. *The Stata Journal*, 2(1):45–64.

Patil, I. (2021). Visualizations with statistical details: The ggstatsplot approach. *Journal of Open Source Software*, 6(61):3167.

R Core Team (2026). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software*, 48(2):1–36.

Rousseeuw, P. J. and Leroy, A. M. (2005). *Robust Regression and Outlier Detection*. Wiley.

Sharpe, D. (2015). Chi-square test is statistically significant: Now what? *Practical Assessment, Research and Evaluation*, 20(8).

Sjoberg, D. D., Whiting, K., Curry, M., Lavery, J. A., and Larmarange, J. (2021). Reproducible summary tables with the gtsummary package. *The R Journal*, 13(1):570–580.

Wickham, H. et al. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.

Xie, Y. (2014). knitr: A comprehensive tool for reproducible research in R. In Stodden, V., Leisch, F., and Peng, R. D., editors, *Implementing Reproducible Computational Research*. Chapman and Hall/CRC. ISBN 978-1466561595.

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2016). *bookdown: Authoring Books and Technical Documents with R Markdown*. Chapman and Hall/CRC, Boca Raton, Florida.

Xie, Y. (2026a). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.48.

Xie, Y. (2026b). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.52.

Xie, Y., Allaire, J., and Grolemund, G. (2018). *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida.

Xie, Y., Dervieux, C., and Riederer, E. (2020). *R Markdown Cookbook*. Chapman and Hall/CRC, Boca Raton, Florida.