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`
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`

Method	Source slug
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
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)

Lab
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 ($\mathrm{BIC} = -84$); 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:

$$Y \mid x \sim \mathrm{Beta}(\mu\phi, (1-\mu)\phi),$$

with $\mathrm{logit}(\mu) = x^\top \beta$ 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

```
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 $Y = (Y_1, \ldots, Y_k)$ with $\sum_j Y_j = 1$ and $Y_j > 0$ follows a Dirichlet distribution $\mathrm{Dir}(\alpha_1, \ldots, \alpha_k)$. 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

```
# 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)

# 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

$$\|y - X\beta\|^2 + \lambda\left[\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2^2\right].$$

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 $\lambda_{1\text{se}}$ 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) ($\Delta\text{QIC} = 4$)." 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

$$g\big(\mathbb{E}[Y \mid u]\big) = X^\top \beta + Z^\top u, \qquad u \sim \mathcal{N}(0, \Sigma),$$

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

```
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(), ziformula = ~ ...)` 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; `countreg` 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

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 $\lambda_{\min}$ and $\lambda_{1\text{se}}$. 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 $\lambda_{1\text{se}}$ 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 mpg² at the chosen λ." Always report which λ rule was used; $\lambda_{\min}$ tends to over-fit, while $\lambda_{1\text{se}}$ 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: $|\mathrm{DFBETA}_{ij}| > 2/\sqrt{n}$.

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 ($\mathrm{SD} = 5.92$, 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 ≥ 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 ≈ 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 ($\Delta\mathrm{AIC}$): < 2 inconclusive, 4–7 moderate preference, > 10 decisive. Relative model weights $\exp(-\Delta\mathrm{AIC}/2)$ 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

```
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 $\exp(-\Delta\mathrm{AIC}/2)$, 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 $x, x^2, x^3, \ldots$.
- 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

```r
library(nnet); library(broom); library(gtsummary)

data(iris)
fit <- multinom(Species ~ Sepal.Length + Sepal.Width, data = iris, trace = FALSE)

# 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: $Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

OLS estimates: $\hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top Y$. 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): $\mathrm{VIF}_j = 1/(1 - R_j^2)$, 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

```r
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 $\log(\text{person-years})$, 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 $\log(\text{exposure})$, 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

$$\log \frac{\mathrm{P}(Y \le j)}{\mathrm{P}(Y > j)} = \alpha_j - X^\top \beta, \qquad j = 1, \ldots, k-1,$$

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 $Y \sim \text{Poisson}(\mu)$ 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
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

```r
# Simulated quadratic relationship
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

```r
d ← mtcars
d$am ← factor(d$am)

fit_probit ← glm(am ~ wt + hp, data = d,
                 family = binomial(link = "probit"))
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::probitmfx()` 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 $\mathrm{Var}(Y) = \phi\mu$ (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 | cluster)` 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: $\lambda_{\min}$ minimises CV error, $\lambda_{1\text{se}}$ 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 $\lambda_{1\text{se}} = 0.5$ 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 mpg² versus 9.8 mpg² 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 $\lambda_{1\text{se}}$ for more shrinkage than $\lambda_{\min}$; 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; `hdi` for high-dimensional inference including debiased ridge; `caret` and `tidymodels` 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

```r
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:

$$\mathrm{AUC} = \mathrm{P}(\hat{p}_i > \hat{p}_j \mid Y_i = 1, Y_j = 0).$$

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

```
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 $\Delta\mathrm{AUC} = 0.04$ ($p = 0.12$), 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

```r
library(MASS)

# 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

```
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:

$$L(\beta, \sigma) = \prod_{i \notin \text{cens}} \phi\left(\frac{Y_i - X_i^\top \beta}{\sigma}\right) \cdot \prod_{i \in \text{cens}} \Phi\left(\frac{L - X_i^\top \beta}{\sigma}\right).$$

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
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 handling; `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 prespecifying 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 β·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`.

```r
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

```r
summary(fit_gam)
AIC(fit_lm, fit_gam)
```

```r
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

```r
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·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]]$"F value"[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.

```r
# 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

```r
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:

```r
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(glance(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)