11 Survival Analysis

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

Time-to-event methods: Kaplan-Meier, Cox regression, time-varying covariates, competing risks, multistate models, and survival-targeted machine learning. Censoring is treated as a first-class concept, not a footnote.

This chapter contains **30 method pages** and **5 labs**. If you are not sure which method to read, return to Chapter 0 and follow the decision tree to the right node.

11.1 Method pages

Method	Source slug
Accelerated Failure Time Models	`accelerated-failure-time`
Censoring Types	`censoring-types`
Checking Cox Assumptions	`cox-assumptions-check`
Competing Risks: Cumulative Incidence	`competing-risks-cif`
Concordance and the C-Index	`concordance-c-index`
Conditional Survival	`conditional-survival`
Cox Proportional Hazards Regression	`cox-proportional-hazards`
Fine-Gray Subdistribution Hazards	`fine-gray-regression`
Frailty Models	`frailty-models`
Hazard, Survival, and Cumulative Hazard	`hazard-survival-cumulative`
Interval-Censored Data	`interval-censored-data`
Joint Longitudinal-Survival Models	`joint-longitudinal-survival`
Kaplan-Meier Estimation	`kaplan-meier`
Landmark Analysis	`landmark-analysis`
Left Truncation	`left-truncation`
Log-Rank Test: Details	`log-rank-test-details`
Multi-State Models	`multi-state-models`
Parametric Survival: Log-Normal	`parametric-survival-lognormal`
Parametric Survival: Weibull	`parametric-survival-weibull`
Recurrent Events	`recurrent-events`
Restricted Mean Survival Time (RMST)	`restricted-mean-survival-time`
Royston-Parmar Flexible Parametric Models	`flexible-parametric-royston-parmar`
Schoenfeld Residuals	`schoenfeld-residuals`
Simulating Survival Data	`survival-simulation`
Stratified Cox Models	`stratified-cox`

Method	Source slug
The Brier Score for Survival	`brier-score-survival`
The Nelson-Aalen Estimator	`nelson-aalen-estimator`
Time-Dependent ROC	`time-dependent-roc`
Time-Varying Covariates	`time-varying-covariates`
Weighted Log-Rank Tests	`weighted-log-rank`

11.2 Labs

Lab
Survival primer: KM, log-rank, Cox PH
Time-varying covariates and immortal-time bias
Competing risks and multistate models
Survival ML (random survival forest, DeepSurv)
Time-dependent Brier, IPA, external validation

11.3 Accelerated Failure Time Models

11.3.1 Introduction

Accelerated failure time (AFT) models express covariate effects directly on the *time scale*: each covariate scales the expected survival time by a multiplicative factor. The model is $\log T = x^\top \beta + \sigma W$ with W a standardised noise (extreme-value, Normal, or logistic, depending on the family). The exponentiated coefficient $\exp(-\beta_j)$ is an "acceleration factor" — the multiplier on the time scale per unit increase in x_j. AFT is the natural framework when communicating effects to clinical audiences who reason in survival times rather than hazards, and it remains valid under non-proportional hazards where Cox regression fails.

11.3.2 Prerequisites

A working understanding of parametric survival distributions (Weibull, log-Normal, log-logistic) and the difference between hazard-ratio and acceleration-factor interpretations.

11.3.3 Theory

Common AFT families:

- **Weibull**: log-time errors are extreme-value distributed; this is the unique family with both PH and AFT interpretations, with $\beta_{\text{PH}} = -k\beta_{\text{AFT}}$ where k is the Weibull shape.

- **Log-Normal**: errors are Normal on the log-time scale; produces non-monotone hazards.
- **Log-logistic**: errors are logistic on the log-time scale; produces non-monotone hazards with heavier tails than log-Normal and a closed-form quantile function.
- **Generalised Gamma**: a three-parameter family that nests Weibull, log-Normal, and exponential; useful for selecting among nested AFT alternatives via likelihood-ratio tests.

The acceleration-factor interpretation is direct: $\exp(\beta_j) = 0.7$ means survival times are 30 % shorter for each unit increase in x_j; $\exp(\beta_j) = 1.3$ means 30 % longer.

11.3.4 Assumptions

The chosen parametric family is correct (verifiable through residual diagnostics and AIC comparisons), independent non-informative censoring, and the absence of unmodelled time-varying covariate effects.

11.3.5 R Implementation

```
library(survival); library(flexsurv)

data(lung)

# Weibull AFT
aft_w ← survreg(Surv(time, status) ~ age + sex, data = lung,
                dist = "weibull")
summary(aft_w)

# Log-normal AFT
aft_ln ← survreg(Surv(time, status) ~ age + sex, data = lung,
                 dist = "lognormal")

# flexsurv alternative
flex_w ← flexsurvreg(Surv(time, status) ~ age + sex, data = lung, dist = "weibull")
flex_w

# AFT interpretation
exp(coef(aft_w)[-1])   # acceleration factors (time ratio)
```

11.3.6 Output & Results

`survreg()` returns AFT coefficients on the log-time scale plus the scale parameter; `flexsurvreg()` provides equivalent estimates with cleaner prediction interfaces. Exponentiating the slope coefficients gives acceleration factors interpretable as time ratios.

11.3.7 Interpretation

A reporting sentence: “A Weibull AFT model estimated that each additional year of age accelerated the event by a factor of 0.98 (95 % CI 0.97 to 1.00) — equivalently, a 2 % shorter expected time per year — and females had 30 % longer expected survival than

males (acceleration factor 1.30, 95 % CI 1.06 to 1.59).” Choose the AFT-vs-PH framing based on the audience: clinicians who think in survival times prefer acceleration factors, while those used to hazard ratios prefer PH.

11.3.8 Practical Tips

- Weibull AFT is equivalent to Weibull PH up to a fixed transformation; choose the parameterisation that matches the audience.
- Use AIC across distributions (exponential, Weibull, log-Normal, log-logistic, generalised gamma) to select the family; AICs differ substantially when the hazard shape varies.
- Check goodness of fit with Cox-Snell residuals (`flexsurv::resid()`) — a Cox-Snell residual against $\exp(1)$ should follow the unit exponential.
- The log–log survival plot is the standard diagnostic for Weibull suitability; clear non-linearity rules it out and motivates a different family.
- For non-monotone hazards (log-Normal, log-logistic), AFT is mandatory — there is no PH analogue without a time-varying coefficient.
- For very small samples, AFT models are more efficient than Cox because the parametric structure adds information; for large samples with PH, Cox is competitive without distributional assumptions.

11.3.9 R Packages Used

`survival` for `survreg()` with multiple AFT families; `flexsurv` for `flexsurvreg()` with cleaner prediction and richer family support including generalised gamma; `eha` for additional parametric extensions; `rstpm2` for spline-based parametric models that bridge AFT and Royston-Parmar.

11.3.10 For Reviewers

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

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

11.3.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.4 The Brier Score for Survival

11.4.1 Introduction

The Brier score for survival is a prediction-loss metric that combines calibration and discrimination into one number, evaluated at a chosen time horizon t. It generalises the binary Brier score to right-censored data via inverse-probability-of-censoring weights, so a model that predicts a survival probability $\hat{S}_i(t)$ for each subject can be assessed against the actual observed status at t. Unlike the C-index, which captures only ranking, the Brier score penalises miscalibrated probabilities — a model that ranks well but consistently overpredicts survival will score worse on Brier than on concordance.

11.4.2 Prerequisites

A working understanding of survival predictions, calibration vs discrimination, censoring, and the binary Brier score $(\hat{p} - y)^2$.

11.4.3 Theory

At time t, the censoring-weighted Brier score is

$$\mathrm{BS}(t) = \frac{1}{n} \sum_{i=1}^{n} w_i(t) \big(Y_i(t) - \hat{S}_i(t)\big)^2,$$

where $Y_i(t) = \mathbf{1}\{T_i > t\}$ is the observed survival indicator (when defined) and $w_i(t)$ is an inverse-probability-of-censoring weight that adjusts for the missing $Y_i(t)$ when subject i is censored before t. The integrated Brier score (IBS, also called the continuous ranked probability score for survival) averages $\mathrm{BS}(t)$ over a follow-up window and gives a single-number summary suited to model selection.

A useful reference is the Brier score of the marginal Kaplan-Meier estimator (no covariates); a model that delivers a lower IBS than the null KM has produced informative predictions.

11.4.4 Assumptions

The censoring weights $w_i(t)$ are correctly estimated, typically via the Kaplan-Meier of the censoring distribution, which assumes non-informative censoring conditional on covariates.

11.4.5 R Implementation

```r
library(pec); library(survival)

data(lung)
fit <- coxph(Surv(time, status) ~ age + sex, data = lung, x = TRUE, y = TRUE)
pec_out <- pec(list(fit), data = lung,
               formula = Surv(time, status) ~ 1,
               times = seq(100, 800, by = 100))

plot(pec_out)
crps(pec_out)
```

11.4.6 Output & Results

`pec()` returns time-dependent Brier scores at the requested horizons together with the null KM benchmark. `crps()` returns the integrated Brier score (continuous ranked probability score) over the requested window. The plot overlays the model and reference Brier curves; a model curve below the KM curve at every t represents predictive improvement throughout follow-up.

11.4.7 Interpretation

A reporting sentence: "The Cox model achieved an integrated Brier score of 0.16 over 100–800 days, compared to 0.21 for the null Kaplan-Meier; predictive improvement was largest between 200 and 500 days, where the model's time-specific Brier was about 30 % below the reference." Pair the IBS with a C-index; together they cover both calibration and discrimination.

11.4.8 Practical Tips

- Compare the model to the null KM at every time point you report; without that reference, an absolute Brier score is hard to interpret.
- Time-dependent Brier curves reveal when the model is most and least informative; this often identifies follow-up windows where extra covariates would help.
- For honest assessment, use cross-validation: `pec(..., splitMethod = "cv10")` performs 10-fold CV and adjusts for optimism.
- Integrated Brier (IBS) is a single-number summary suited to model ranking; report it alongside per-time-point Brier curves for transparency.
- Complement with calibration plots (predicted vs observed survival probabilities at t) and the C-index; the three together give a full predictive-performance picture.
- Brier-score weights depend on a censoring model; if censoring depends on covariates, fit a Cox model for the censoring time and use it instead of the marginal KM.

11.4.9 R Packages Used

`pec` for `pec()`, `crps()`, and prediction-error curves; `riskRegression` for unified discrimination, calibration, and Brier metrics with bootstrap CIs; `survival` for the Cox fit; `timeROC` for time-dependent AUCs that complement the Brier-score view.

11.4.10 For Reviewers

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

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

11.4.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.5 Censoring Types

11.5.1 Introduction

Censoring is the defining feature of survival data: the exact event time is unknown but partially observed, because follow-up ended, the event was screened periodically, or the subject was lost to observation. Handling censoring correctly is what separates survival analysis from ordinary regression on time-to-event endpoints. The wrong censoring assumption is one of the most consequential analytic errors in clinical studies, so identifying the censoring type and choosing methods that respect it is the first step of any time-to-event workflow.

11.5.2 Prerequisites

A working understanding of time-to-event concepts, follow-up windows, and the difference between an event and an end-of-follow-up timestamp.

11.5.3 Theory

The main censoring categories:

- **Right censoring**: the event had not yet occurred when observation ended. The most common type, the default in clinical trials.
- **Left censoring**: the event occurred before the first observation; the exact time is unknown but is in $[0, t_{\text{first observation}}]$.
- **Interval censoring**: the event occurred within a known interval $(L, R]$, typical in studies with periodic check-ups.

Two design-based subtypes of right censoring:

- **Type I**: fixed follow-up time; censoring time is identical for all administratively censored subjects.
- **Type II**: follow until a pre-specified number of events occur; censoring time depends on the data realisation.

Informative censoring is the most dangerous failure mode: the censoring mechanism is related to the underlying event risk (e.g., subjects drop out because they are doing badly). It violates the independence assumption that nearly every standard method requires and can severely bias estimates without warning.

11.5.4 Assumptions

Non-informative censoring conditional on covariates included in the model. If censoring depends on prognosis through unobserved variables, sensitivity analyses (informative-censoring bounds, multiple imputation) are needed.

11.5.5 R Implementation

```r
library(survival)

# Right-censored data: time and status
Surv(time = c(5, 8, 12, 6), event = c(1, 1, 0, 1))

# Interval-censored: event occurred between left and right
Surv(time = c(5, 8), time2 = c(10, 12), type = "interval2")

# Counting-process style (time-varying covariates): start, stop, status
Surv(time = c(0, 5, 10), time2 = c(5, 10, 15), event = c(0, 0, 1),
     type = "counting")
```

11.5.6 Output & Results

`Surv` objects display censoring through a `+` suffix on right-censored times, brackets for interval-censored intervals, and pairs for counting-process intervals. Every downstream survival function (`survfit`, `coxph`, `survreg`) takes a `Surv` object as the response and dispatches on its censoring type.

11.5.7 Interpretation

A reporting sentence: “Of 228 patients, 165 (72 %) experienced the event and 63 (28 %) were administratively right-censored at study end (median follow-up 310 days, range 5 to 1,022). No interval-censored or left-censored events were present. Censoring is assumed non-informative; a sensitivity analysis under tipping-point assumptions confirmed conclusions are robust to plausible departures.” Always quantify censoring and discuss its assumed mechanism.

11.5.8 Practical Tips

- Right censoring is the default; specifying other types requires the `type` argument to `Surv()`.

- Interval-censored data require `icenReg` or `survival` with `Surv(L, R, type = "interval2")`; treating intervals as exact event times biases estimates.
- Check for informative censoring by comparing baseline characteristics of censored vs event subjects; large differences are a warning sign.
- Distinguish administrative censoring (study end) from loss to follow-up (subject left); the two have different mechanisms and warrant separate sensitivity analyses.
- Plot censoring patterns: a histogram of censoring times alongside event times often reveals administrative cut-offs and dropout clusters that affect interpretation.
- For competing risks, treating competing events as censoring is itself a censoring choice; cumulative incidence analysis avoids it.

11.5.9 R Packages Used

`survival` for `Surv()` with all censoring types and the standard estimators; `icenReg` for interval-censored regression; `cmprsk` and `tidycmprsk` for competing-risks alternatives; `epi` for tools targeted at epidemiological censoring scenarios.

11.5.10 For Reviewers

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

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

11.5.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.6 Competing Risks: Cumulative Incidence

11.6.1 Introduction

A competing risk is an event whose occurrence prevents the event of interest from happening — death from a non-disease cause prevents disease-specific death, retirement prevents return-to-work, and so on. In this setting the standard Kaplan-Meier estimator overestimates the cumulative incidence of the event of interest, because it treats competing events as censoring rather than as the absorbing states they actually are. The cumulative incidence function (CIF), estimated non-parametrically by Aalen-Johansen, is the correct summary: it gives the probability of having experienced each cause by time t in the presence of all other competing causes.

11.6.2 Prerequisites

A working understanding of survival analysis, censoring, and the Kaplan-Meier estimator; familiarity with multi-state thinking ("alive → cause-1 event" vs "alive → cause-2 event") helps.

11.6.3 Theory

The CIF for cause k is

$$\mathrm{CIF}_k(t) = \mathrm{P}(T \le t, \text{cause} = k) = \int_0^t S(u^-) h_k(u)\, \mathrm{d}u,$$

where h_k is the cause-specific hazard and S is the overall (any-cause) survival. Summing CIFs across causes recovers $1 - S(t)$. The Aalen-Johansen estimator generalises Nelson-Aalen for multi-state processes; Gray's test compares CIFs across groups, with the null that the CIF for the event of interest is equal across groups. For regression, the Fine-Gray subdistribution-hazards model targets covariate effects on the CIF directly, whereas cause-specific Cox models target effects on the cause-specific hazard — the two answer different scientific questions.

11.6.4 Assumptions

Non-informative censoring (independent of competing causes), only one event per subject (the first event ends follow-up), and accurate cause classification.

11.6.5 R Implementation

```r
library(cmprsk)

# Simulated: event causes 1 (event of interest), 2 (competing), 0 (censored)
set.seed(2026)
t_time ← rexp(300, 0.3)
cause ← sample(0:2, 300, replace = TRUE, prob = c(0.3, 0.5, 0.2))
group ← factor(sample(c("A", "B"), 300, replace = TRUE))

fit ← cuminc(t_time, cause, group)
plot(fit, xlab = "Time", ylab = "Cumulative incidence",
     color = c("red", "blue", "green", "orange"))

# Gray's test for equality of CIFs
print(fit)
```

11.6.6 Output & Results

`cuminc()` returns CIF estimates for each combination of cause and group, plus Gray's test χ^2 statistics for equality across groups within each cause. The plot overlays the four CIF curves; cumulative incidences for the two causes within each group sum to less than the corresponding $1 - S$ at every time.

11.6.7 Interpretation

A reporting sentence: "The five-year cumulative incidence of event 1 was 25 % in group A and 40 % in group B (Gray's test $p = 0.01$); event 2 incidences were 20 % vs 18 % ($p = 0.65$). KM analysis would have estimated event-1 incidence in group B at 52 %, illustrating the upward bias of ignoring the competing cause." Always present CIFs for every competing cause and explain why KM is inappropriate.

11.6.8 Practical Tips

- Never report $1 - \hat{S}(t)$ from Kaplan-Meier as "cumulative incidence" in competing-risks settings; it overestimates the true cumulative incidence.
- Plot the CIF for every competing cause so readers see the partition of the overall event probability across causes.
- For regression effects on the CIF (e.g., "this covariate increases the cumulative incidence of cause 1"), use Fine-Gray subdistribution hazards (`cmprsk::crr` or `riskRegression::FGR`).
- For mechanistic effects on cause-specific hazards (e.g., "this drug accelerates cause-1 events but the drop in event-1 cumulative incidence is driven by competing event 2"), use cause-specific Cox models.
- Gray's test compares CIFs; the log-rank test on cause-1 events with cause-2 events censored compares cause-specific hazards. Report both when they disagree.
- For visualisation in `ggplot2` , `tidycmprsk` and `ggsurvfit` provide tidy interfaces with cleaner output than base `cmprsk::plot.cuminc()` .

11.6.9 R Packages Used

`cmprsk` for `cuminc()` and `crr()` (Fine-Gray); `tidycmprsk` for tidy-style competing-risks workflows; `riskRegression` for `FGR()` , `CSC()` (cause-specific Cox), and unified prediction interfaces; `ggsurvfit` for ggplot-friendly CIF plots.

11.6.10 For Reviewers

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

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

11.6.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.7 Concordance and the C-Index

11.7.1 Introduction

Harrell's concordance index (C-index) generalises the ROC AUC to censored survival data. It is the probability that, for a randomly chosen pair of subjects, the one with the higher model-predicted risk actually fails first. The C-index is therefore the headline measure of *discrimination* for any survival model — Cox regression, parametric survival, machine-learning predictors — and the default reporting standard alongside the model's coefficients in clinical prediction-model papers.

11.7.2 Prerequisites

A working understanding of survival analysis, Cox regression, censoring, and the AUC for binary classifiers; familiarity with the distinction between calibration and discrimination.

11.7.3 Theory

Define a pair (i, j) as *comparable* if it can be determined which subject failed first — at least one observed event with the other subject either failing later or surviving past the first failure. The C-index is the proportion of comparable pairs for which the higher predicted risk corresponds to the earlier observed event:

$$C = \mathrm{P}(\hat{\eta}_i > \hat{\eta}_j \mid T_i < T_j, \delta_i = 1).$$

$C = 0.5$ corresponds to random ranking and $C = 1.0$ to perfect discrimination. Variants address specific limitations of Harrell's original index: Uno's C-statistic uses inverse-probability-of-censoring weights for unbiased estimation under heavy or non-uniform censoring; time-dependent AUCs (Heagerty, Uno) describe how discrimination changes across the follow-up period; the integrated Brier score is a complementary calibration-discrimination compound.

11.7.4 Assumptions

Risk scores come from a survival model whose linear predictor $\hat{\eta} = X\hat{\beta}$ is a meaningful ranking; the censoring mechanism is non-informative for Harrell's C and may be informative for Uno's C up to its weighting.

11.7.5 R Implementation

```r
library(survival)

data(lung)
fit ← coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
summary(fit)$concordance

# Time-dependent c-index via survcomp or timeROC
```

11.7.6 Output & Results

`summary(fit)$concordance` returns Harrell's C-index together with its standard error. For richer diagnostics, `survcomp::concordance.index()` adds bootstrap confidence intervals and `timeROC::timeROC()` returns time-dependent AUCs at user-specified landmarks.

11.7.7 Interpretation

A reporting sentence: "The Cox model achieved a Harrell's C-index of 0.63 (95 % CI 0.56 to 0.70) on the development data, indicating modest discrimination; on the held-out validation cohort the C-index dropped to 0.59, suggesting some optimism that the bootstrap procedure did not fully correct." Always pair the development-set C with an external or cross-validated estimate; in-sample concordance is optimistic.

11.7.8 Practical Tips

- $C > 0.7$ is conventionally considered acceptable for clinical use, $C > 0.8$ good; thresholds depend on the application and on what alternative classifiers achieve.
- Use Uno's C-statistic (`survAUC::UnoC()`) when censoring is heavy or non-uniform; Harrell's C can be biased upward in those settings.
- Report bootstrap confidence intervals via `survcomp::concordance.index()`; the analytical SE in `coxph` output is conservative.
- Time-dependent AUCs (`timeROC::timeROC()`) complement the overall C-index by showing when discrimination is highest and lowest across follow-up.
- For external validation, compute the C-index on an independent dataset; a development-set C without validation should never be the only discrimination measure reported.
- Pair discrimination (C-index) with calibration (calibration-in-the-large, calibration slope, Brier score); a model can rank well but predict poorly calibrated absolute probabilities.

11.7.9 R Packages Used

`survival` for `coxph()` with built-in concordance computation; `survcomp` for `concordance.index()` and bootstrap CIs; `timeROC` for time-dependent AUCs; `pec` for prediction-error curves and integrated Brier scores; `riskRegression` for unified discrimination and calibration metrics.

11.7.10 For Reviewers

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

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

11.7.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.8 Conditional Survival

11.8.1 Introduction

Conditional survival is the probability of surviving an additional time t given that a patient has already survived to time s. It answers the clinically relevant question that absolute survival cannot: a five-year survival probability of 30 % at diagnosis says little to a patient who has already lived three years. The conditional five-year-from-now estimate may be 70 % or higher, dramatically updating the prognosis. Survivorship clinics, oncology long-term follow-up, and patient counselling all rely on conditional rather than unconditional survival.

11.8.2 Prerequisites

A working understanding of the survival function $S(t)$, conditional probability, and the Kaplan-Meier estimator.

11.8.3 Theory

By the definition of conditional probability,

$$\mathrm{P}(T > t + s \mid T > s) = \frac{\mathrm{P}(T > t + s)}{\mathrm{P}(T > s)} = \frac{S(t + s)}{S(s)}.$$

A patient who has survived s years is now in the risk set at s; their next-t-year survival is the ratio of survival probabilities. For Kaplan-Meier estimates, this is computed from $\hat{S}(s)$ and $\hat{S}(t + s)$ directly. Confidence intervals require variance propagation through the ratio; the log or log-log transformation gives intervals that respect the $[0, 1]$ bounds.

11.8.4 Assumptions

A reliable Kaplan-Meier or parametric survival estimate at both s and $t + s$; in practice this means sufficient follow-up so that $\hat{S}(t + s)$ is not based on a tiny remaining cohort.

11.8.5 R Implementation

```r
library(survival)

data(lung)
fit ← survfit(Surv(time, status) ~ 1, data = lung)

# Unconditional 2-year survival
summary(fit, times = c(500, 1000))$surv   # approximate 2- and 3-year

# Conditional 1-year survival given 1-year survivorship
times ← summary(fit, times = c(365, 730))
cond_surv ← times$surv[2] / times$surv[1]
cond_surv
```

11.8.6 Output & Results

The ratio of two KM survival estimates at chosen landmark times. For confidence intervals, transform onto the log-log scale, compute SEs by the delta method, and back-transform. The `condsurv` package automates the computation including bootstrap intervals.

11.8.7 Interpretation

A reporting sentence: “Unconditional one-year survival was 48 %; conditional on having survived one year, the probability of surviving an additional year was 66 % (95 % CI 56 % to 75 %); conditional on having survived two years, the next-year survival rose to 78 %.” Always report the landmark time s explicitly; conditional survival is meaningless without it.

11.8.8 Practical Tips

- Report conditional survival at clinically meaningful landmarks: one-year, five-year, ten-year survivor checkpoints in oncology, post-transplant 90-day survival in transplantation.
- `condsurv` automates the computation and produces clean tables and plots; it handles confidence intervals via the log-log transformation by default.
- Confidence intervals must respect the $[0, 1]$ bounds; the log-log transformation is the standard choice and is preferable to the naive Wald interval.
- For dynamic prediction (conditional survival as a function of s), landmark analyses or joint longitudinal-survival models give more refined estimates that incorporate post-baseline information.
- Conditional survival is particularly useful in patient counselling because it answers the question patients actually ask: “Given that I’m still alive, how much longer can I expect to live?” rather than the population-level survival reported at diagnosis.
- Pair conditional-survival reports with the underlying KM curve so readers can see how the conditional estimate evolves across the cohort’s experience.

11.8.9 R Packages Used

`survival` for the Kaplan-Meier baseline; `condsurv` for tabular conditional survival with CIs; `survminer::ggsurvplot()` for KM curves to display alongside conditional summaries; `dynpred` and `JM` for dynamic prediction beyond the simple ratio of survival probabilities.

11.8.10 For Reviewers

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

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

11.8.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.9 Checking Cox Assumptions

11.9.1 Introduction

A fitted Cox model is only as good as its assumptions: proportional hazards across covariate levels, the right functional form for continuous predictors, and the absence of one or two highly influential observations driving the coefficient estimates. Three residual families address these in turn — Schoenfeld for proportional hazards, martingale for functional form, and DFBETA-style influence diagnostics for unduly leveraged observations. Reporting all three is now standard for any clinical-trial Cox analysis.

11.9.2 Prerequisites

A working understanding of Cox proportional-hazards regression, residual-based diagnostics, and the difference between the proportional-hazards assumption and the linearity-of-the-linear-predictor assumption.

11.9.3 Theory

- **Schoenfeld residuals** are computed at each event time as the difference between the observed covariate value and the expected value across the risk set. Under proportional hazards, they should have no systematic relationship with time. `cox.zph()` tests the correlation between scaled residuals and a time transform (default $g(t) = \log t$).

- **Martingale residuals** are observed minus expected events for each subject. Plotting them against a continuous covariate exposes non-linearity: a smooth trend indicates the linear-predictor assumption fails for that covariate.
- **Deviance residuals** normalise martingale residuals to make outlying observations visible on a roughly symmetric scale.
- **DFBETA** measures how much each coefficient changes when each individual observation is removed; large absolute values flag influential observations whose exclusion changes the conclusions.

11.9.4 Assumptions

A Cox model has been fitted, with sufficient events to estimate residuals reliably (typically at least 10 events per covariate, and substantially more for stable diagnostic plots).

11.9.5 R Implementation

```
library(survival); library(survminer)

data(lung)
fit ← coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)

# PH test
zph ← cox.zph(fit)
print(zph)
ggcoxzph(zph)

# Linearity check for continuous age
ggcoxfunctional(Surv(time, status) ~ age,
  data = na.omit(lung[, c("time", "status", "age")]))

# Influential observations
ggcoxdiagnostics(fit, type = "dfbeta")
```

11.9.6 Output & Results

`cox.zph()` returns per-covariate χ^2 tests and a global test for proportional hazards. `ggcoxzph()` plots the smoothed scaled Schoenfeld residuals with a horizontal reference at the constant-HR null. `ggcoxfunctional()` plots martingale residuals against the continuous covariate with a smoother. `ggcoxdiagnostics()` displays DFBETA for each coefficient.

11.9.7 Interpretation

A reporting sentence: “Diagnostics for the Cox model showed no PH violations (Schoenfeld global $p = 0.08$, per-covariate $p > 0.10$), approximately linear effects for age (martingale residuals showed no systematic trend), and three influential observations (DFBETA > 0.1 in absolute value) that were investigated and retained after sensitivity analysis showed coefficients changed by less than 5 %.” Always report the global PH test, the per-covariate tests, and any sensitivity analysis you ran.

11.9.8 Practical Tips

- If PH fails for a covariate, choose between stratification (`+ strata(var)`) and a time-varying coefficient via `tt()` or `survSplit()`; the choice depends on whether the stratifier's effect is itself of interest.
- If linearity fails for a continuous covariate, add a restricted cubic spline (`rms::rcs()` or `splines::ns()`) instead of dichotomising; categorisation discards information.
- Influential observations identified by DFBETA should be inspected for data errors first; if real, rerun without them as a sensitivity analysis and report any material changes.
- The global Schoenfeld test aggregates per-covariate tests; a global non-significant result with one significant per-covariate test still warrants action on that covariate.
- For continuous covariates with non-linear effects, the martingale-residual plot against the variable is often more informative than any formal test.
- All three diagnostics should appear in a model-checking section; reporting only the model output without diagnostics is no longer acceptable in clinical-trial publications.

11.9.9 R Packages Used

`survival` for `cox.zph()`, `residuals()`, and `coxph()` with strata; `survminer` for `ggcoxzph()`, `ggcoxfunctional()`, `ggcoxdiagnostics()`, and ggplot-flavoured Cox diagnostics; `rms` for `cph()` with built-in spline and validation tools; `flexsurv` and `timereg` for parametric and additive-hazards alternatives when assumptions fail.

11.9.10 For Reviewers

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

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

11.9.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.10 Cox Proportional Hazards Regression

11.10.1 Introduction

Cox proportional hazards regression is the most widely used survival regression model in clinical and epidemiological research. Its central insight is that hazard *ratios* between subjects can be estimated without specifying or estimating the baseline hazard $h_0(t)$. The partial likelihood, introduced by D. R. Cox in 1972, conditions on the risk set at each event time and depends only on the regression coefficients, leaving the baseline hazard unspecified. The result is a semi-parametric model that combines the interpretability of regression with the flexibility of non-parametric survival analysis.

11.10.2 Prerequisites

A working understanding of hazard functions, the proportional-hazards assumption, and the partial-likelihood concept.

11.10.3 Theory

The Cox model is

$$h(t \mid x) = h_0(t)\exp(x^\top \beta).$$

The partial likelihood over event times is

$$L_p(\beta) = \prod_{i:\delta_i=1} \frac{\exp(x_i^\top \beta)}{\sum_{j \in R(t_i)} \exp(x_j^\top \beta)},$$

where $R(t_i)$ is the risk set just before t_i. Maximising L_p produces $\hat\beta$ whose exponentiated components are hazard ratios. Tied event times require a handling rule: Breslow's approximation (default) is fast and consistent under no ties, Efron's is more accurate when ties are common, and the exact-marginal method is the gold standard for discrete-time data.

The model assumes the hazard ratio between any two covariate patterns is constant over time — proportional hazards. When this fails, options include stratification, time-varying coefficients, or switching to a parametric AFT model.

11.10.4 Assumptions

Proportional hazards across covariate levels, independent non-informative censoring, and the absence of unmodelled time-varying covariate effects. Adequate events-per-variable (EPV) ratios — at least 10–15 events per regression coefficient — for stable inference.

11.10.5 R Implementation

```r
library(survival); library(broom)

data(lung)
fit ← coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
tidy(fit, conf.int = TRUE, exponentiate = TRUE)

# Check PH assumption
cox.zph(fit)
```

11.10.6 Output & Results

`tidy(fit, exponentiate = TRUE)` returns a tibble of hazard ratios, confidence intervals, and Wald p-values for each covariate. `summary(fit)` adds the likelihood-ratio test, score test, and Wald global tests for the joint null. `cox.zph()` reports the proportional-hazards diagnostic for each covariate and globally.

11.10.7 Interpretation

A reporting sentence: "A multivariable Cox proportional-hazards model adjusted for age, sex, and ECOG performance status estimated that each additional year of age increased the hazard of death by 2 % (HR 1.02, 95 % CI 1.00 to 1.04, $p = 0.04$); female sex was associated with a 40 % lower hazard (HR 0.60, 95 % CI 0.42 to 0.85, $p = 0.004$). The proportional-hazards assumption was supported (global Schoenfeld test $p = 0.55$)." Always pair coefficient reporting with PH diagnostics.

11.10.8 Practical Tips

- Always check proportional hazards via `cox.zph()` after fitting; reporting Cox HRs without PH diagnostics is no longer acceptable in clinical-trial publications.
- Use Efron's tie-handling method (`ties = "efron"`) for tied event times; the default in older software is Breslow, which is less accurate.
- Report HRs with 95 % confidence intervals; p-values alone are inadequate, and a CI that includes 1.0 is more interpretable than a $p > 0.05$.
- Small events-per-variable (EPV) ratios inflate standard errors and lead to unstable estimates; report the EPV alongside the model.
- For continuous covariates, check linearity in the log-hazard via martingale residuals or restricted cubic splines; categorising a continuous covariate to "fix" non-linearity discards information.
- For non-PH covariates, choose between stratification (if the stratifier is not of interest) and time-varying coefficients (if the time-varying effect is itself the question).

11.10.9 R Packages Used

`survival` for `coxph()`, `cox.zph()`, and `Surv()`; `broom` for `tidy()` and `glance()` summaries; `survminer` for ggplot-style diagnostic plots; `rms` for `cph()` with built-in spline and bootstrap-validation tools.

11.10.10 For Reviewers

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

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

11.10.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.11 Fine-Gray Subdistribution Hazards

11.11.1 Introduction

In competing-risks data, two questions often co-exist: "what is this covariate's mechanistic effect on the event of interest" and "what is its effect on the population-level probability of experiencing that event by time t?" Cause-specific Cox regression answers the first; Fine-Gray subdistribution-hazards regression answers the second. The two questions can disagree dramatically — a covariate can leave the cause-specific hazard unchanged while still raising or lowering the cumulative incidence because it changes the competing-event hazard. Reporting only one when both are relevant is a common analytical mistake.

11.11.2 Prerequisites

A working understanding of competing-risks data, cumulative incidence functions, and the cause-specific vs subdistribution-hazards distinction.

11.11.3 Theory

The subdistribution hazard for cause 1 is

$$h_1^{\text{sub}}(t \mid x) = \lim_{\Delta t \to 0} \frac{\mathrm{P}\big(t \le T < t + \Delta t,\ C = 1 \mid T \ge t \text{ or } (T < t \text{ and } C \ne 1),\ x\big)}{\Delta t}.$$

The risk set keeps subjects who have already experienced a competing event, with diminishing weights. Under the Fine-Gray model

$$h_1^{\text{sub}}(t \mid x) = h_{10}(t)\exp(x^\top \beta),$$

the exponentiated coefficient $\exp(\beta_j)$ is a *subdistribution hazard ratio* and maps directly onto changes in cumulative incidence: a $\mathrm{sHR} > 1$ means higher cumulative incidence of cause 1 per unit of covariate. The model is fitted by a modified partial likelihood with inverse-probability-of-censoring weights.

11.11.4 Assumptions

Proportional subdistribution hazards (an analogue of proportional cause-specific hazards), independent censoring (or correctly weighted), and accurate cause classification.

11.11.5 R Implementation

```r
library(cmprsk)

set.seed(2026)
t_time ← rexp(300, 0.3)
cause  ← sample(0:2, 300, replace = TRUE, prob = c(0.3, 0.5, 0.2))
x1     ← rnorm(300); x2 ← rbinom(300, 1, 0.5)

fg ← crr(ftime = t_time, fstatus = cause, cov1 = cbind(x1, x2),
         failcode = 1, cencode = 0)
summary(fg)
```

11.11.6 Output & Results

`crr()` returns subdistribution coefficients on the log-sHR scale, standard errors, p-values, and (after `summary()`) exponentiated subdistribution hazard ratios with confidence intervals. The output structure matches Cox regression but the interpretation refers to cumulative incidence rather than cause-specific hazard.

11.11.7 Interpretation

A reporting sentence: "Fine-Gray subdistribution-hazards regression for cause 1 estimated $\mathrm{sHR} = 1.45$ for x_1 (95 % CI 1.10 to 1.90), meaning a 45 % higher cumulative incidence of cause 1 per unit increase in x_1; a cause-specific Cox model on the same data gave HR 1.32, indicating that part of the effect on cumulative incidence comes via the competing-event hazard." Always report both subdistribution and cause-specific HRs when they differ; they answer different questions.

11.11.8 Practical Tips

- Subdistribution hazard ratios do *not* equal cause-specific hazard ratios; treating them as interchangeable is a common error.
- Use cause-specific Cox for *etiologic* questions ("what affects the disease mechanism?"); use Fine-Gray for *prognostic* / prediction questions ("what affects the probability of disease over time?").
- For prediction tools (nomograms, risk calculators), Fine-Gray is the correct framework because absolute risk depends on cumulative incidence.

- Report both approaches when both questions matter; transparency about the difference prevents readers from misinterpreting one as the other.
- `riskRegression::FGR()` offers a modern alternative with cleaner diagnostics and better small-sample handling.
- Check proportional subdistribution hazards via `crskdiag` or simulated residual plots; the assumption can fail even when proportional cause-specific hazards holds.

11.11.9 R Packages Used

`cmprsk` for `crr()` and `cuminc()`; `riskRegression` for `FGR()`, `CSC()` (cause-specific Cox), and unified prediction interfaces; `tidycmprsk` for tidy-style competing-risks workflows; `survival` for Cox regression on cause-specific hazards as the complementary fit.

11.11.10 For Reviewers

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

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

11.11.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.12 Royston-Parmar Flexible Parametric Models

11.12.1 Introduction

Royston-Parmar models bridge the gap between rigid parametric survival families (exponential, Weibull, log-normal) and the assumption-light Cox semi-parametric model. They place restricted cubic splines on the log cumulative hazard (or on related transforms) so the baseline can take essentially any shape the data demand, while keeping the full parametric likelihood that delivers smooth survival, hazard, and quantile predictions for any covariate pattern. In epidemiological work — particularly population-based cancer studies — they have become the preferred default because they combine Cox-like flexibility with parametric extrapolation.

11.12.2 Prerequisites

A working understanding of Cox regression, parametric survival families, and restricted cubic splines for smooth functional forms.

11.12.3 Theory

The Royston-Parmar model writes

$$\log H(t \mid x) = s\big(\log t; \gamma\big) + x^\top \beta,$$

where s is a restricted cubic spline in $\log t$ with knots placed at quantiles of the log event-time distribution. With k internal knots, the spline has $k + 1$ parameters; common choices are 1–4 knots. Setting `scale = "hazard"` produces a proportional-hazards model; `scale = "odds"` gives a proportional-odds parameterisation; `scale = "normal"` gives a probit parameterisation, equivalent to a generalised log-normal model. Time-dependent effects can be modelled by interacting covariates with spline basis functions.

Predictions on the natural time scale are smooth and available at any horizon, including extrapolation beyond the maximum observed event time, where Cox predictions are undefined.

11.12.4 Assumptions

The chosen scale (PH, PO, or probit) is appropriate; the number of knots is not so large that the baseline is overfitted nor so small that the underlying shape is missed. AIC is the conventional knot-selection criterion.

11.12.5 R Implementation

```r
library(flexsurv)

data(lung)

# 2 internal knots (default placement at quantiles of log-event times)
fit ← flexsurvspline(Surv(time, status) ~ age + sex, data = lung,
                     k = 2, scale = "hazard")
fit
plot(fit, type = "hazard")
plot(fit, type = "survival")
```

11.12.6 Output & Results

`flexsurvspline()` returns spline coefficients $\gamma_0, \gamma_1, \ldots$, covariate effects $\hat{\beta}$ on the log-hazard scale (under PH), and a fully smooth model object. `plot(fit, type = "hazard")` overlays the model-implied hazard on a non-parametric reference, and `plot(fit, type = "survival")` does the same for the survival function. The smoothness is the headline visual difference from a Cox-based survfit step function.

11.12.7 Interpretation

A reporting sentence: “A Royston-Parmar PH model with two internal knots estimated the hazard ratio for female sex at 0.60 (95 % CI 0.42 to 0.85), close to the Cox estimate of 0.59 (0.42 to 0.83); the spline baseline captured a hazard peak around 9 months that the Weibull model missed entirely (AIC: Royston-Parmar 1{,}890, Weibull 1{,}908).” Reporting AIC across knots and against simpler parametric alternatives makes the choice transparent.

11.12.8 Practical Tips

- Start with 2–3 knots and compare AIC across 1–4; more than 4 internal knots tends to overfit small datasets.
- Use `scale = "hazard"` (PH) when the goal matches a Cox interpretation; `"odds"` for a proportional-odds analogue; `"normal"` for a probit / log-normal-like fit.
- Coefficient estimates should be close to those from a Cox model under PH; large discrepancies indicate either Cox tied-event handling differences or an issue with the spline baseline.
- Time-dependent covariate effects come from `gamma1 ~ x` interactions inside `flexsurvspline()`; this lets you test PH non-parametrically within the model.
- For predictions on a fine time grid, `summary(fit, type = "survival", t = seq(0, 1000, 10))` produces smooth survival probabilities ready for plotting.
- In population-based cancer epidemiology, Royston-Parmar models have largely replaced Cox for relative-survival and excess-mortality work; the parametric baseline gives interpretable hazard shapes and clean extrapolation.

11.12.9 R Packages Used

`flexsurv` for `flexsurvspline()` with PH, PO, and probit scales; `rstpm2` for the related Stata-compatible Royston-Parmar implementation with extended time-dependent effects; `survival` for the Cox baseline against which to compare; `ggflexsurv` for ggplot-style plotting of `flexsurvspline` objects.

11.12.10 For Reviewers

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

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

11.12.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)

- Time-dependent Brier, IPA, external validation

11.13 Frailty Models

11.13.1 Introduction

A frailty model adds an unobserved multiplicative effect — the frailty — to the hazard, allowing the survival analogue of a random effect. Frailty captures two things: unmeasured heterogeneity at the individual level (the population mixture of "robust" and "fragile" subjects with different baseline hazards), and clustered survival data where observations within a unit (family, hospital, patient with recurrent events) share an unobserved common factor. Without it, standard errors are wrong for clustered data, and population-level hazard ratios are biased toward the null because of the selection-of-survivors effect.

11.13.2 Prerequisites

A working understanding of Cox regression, random-effects models in linear and generalised linear settings, and the concept of unobserved heterogeneity.

11.13.3 Theory

The shared-frailty Cox model is

$$h_{ij}(t) = h_0(t) u_i \exp(x_{ij}^\top \beta),$$

where u_i is a positive random variable (frailty) shared across observations j within cluster i and conventionally constrained to have mean 1. The Gamma distribution is the textbook default — its conjugate structure with the partial likelihood makes estimation tractable — though log-Normal frailties are increasingly used when the Laplace-approximation likelihood is implemented.

The frailty variance $\theta = \mathrm{Var}(u_i)$ measures the strength of within-cluster correlation; $\theta = 0$ corresponds to no clustering and reduces to the standard Cox model. The likelihood-ratio test for $\theta = 0$ is on the boundary of the parameter space, so the conventional χ^2_1 reference is replaced by a 50-50 mixture of χ^2_0 and χ^2_1.

11.13.4 Assumptions

The frailty distribution (Gamma or log-Normal) is correctly specified, frailties are independent across clusters, and the standard Cox assumptions (PH within cluster) hold.

11.13.5 R Implementation

```
library(survival)

data(kidney)   # two events per patient (recurrent catheter infections)

# Shared frailty
fit ← coxph(Surv(time, status) ~ age + sex + frailty(id), data = kidney)
summary(fit)
```

11.13.6 Output & Results

`summary(fit)` returns the fixed-effect Cox coefficients (interpreted as conditional-on-frailty hazard ratios), the frailty variance estimate, and a likelihood-ratio test for the frailty term against the no-frailty null. The variance estimate is the headline measure of unobserved heterogeneity.

11.13.7 Interpretation

A reporting sentence: "A shared-Gamma frailty Cox model on 38 patients with two recurrent catheter-infection times each estimated a frailty variance of 0.41 (LRT $\chi^2_{0.5} = 6.8$, $p = 0.005$); after accounting for within-patient correlation, female sex was strongly protective (HR 0.36, 95 % CI 0.21 to 0.62)." Always report frailty variance with its significance test; a small variance suggests the frailty term is unnecessary.

11.13.8 Practical Tips

- For frailty variance close to zero, drop the frailty term and refit with the standard Cox model; the simpler fit is more efficient when no clustering is present.
- Use shared-frailty models for naturally clustered data (families, hospitals, repeated measures); individual frailties require recurrent events or external longitudinal information.
- Gamma frailty is the default choice for partial-likelihood implementations; log-Normal requires Laplace approximation and is implemented in `frailtypack` and `coxme`.
- Reported hazard ratios are *conditional* on the frailty (within-cluster); for marginal effects, integrate over the frailty distribution or use generalised estimating equations as an alternative.
- The LRT for $\theta = 0$ is on the parameter-space boundary; use the half-half mixture reference (or simulation-based p-values) to avoid conservative tests.
- For high-dimensional clustering or complex frailty structures, `coxme` provides a mixed-model interface compatible with `lme4`-style formulas.

11.13.9 R Packages Used

`survival` for `coxph()` with `frailty()`; `coxme` for mixed-effects Cox models with multiple frailties and complex random-effect structures; `frailtypack` for Gamma, log-Normal, and joint frailty models including recurrent events; `frailtyEM` for EM-based frailty estimation with cleaner SE handling.

11.13.10 For Reviewers

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

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

11.13.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.14 Hazard, Survival, and Cumulative Hazard

11.14.1 Introduction

A time-to-event distribution admits three mutually equivalent representations: the survival function $S(t)$, the hazard function $h(t)$, and the cumulative hazard $H(t)$. They contain the same information but emphasise different aspects of the process — surviving probability, instantaneous risk, and accumulated risk respectively. Switching freely among them is one of the foundational fluencies of survival analysis, because each form has natural plotting and inferential conventions and most software exposes all three.

11.14.2 Prerequisites

A working understanding of survival analysis basics, the difference between probability and rate, and integration of non-negative functions.

11.14.3 Theory

For a non-negative continuous time-to-event T with density $f(t)$:

- **Survival function**: $S(t) = \mathrm{P}(T > t)$. The probability of surviving past time t.
- **Hazard function**: $h(t) = f(t)/S(t)$. The instantaneous event rate at time t given survival up to t. Hazard is a rate (per unit time), not a probability.
- **Cumulative hazard**: $H(t) = \int_0^t h(u)\,\mathrm{d}u = -\log S(t)$. Accumulated risk by time t.

Any one of the three determines the other two: $S(t) = e^{-H(t)}$, $h(t) = -\mathrm{d}\log S(t)/\mathrm{d}t$. The Kaplan-Meier estimator targets $S(t)$ directly; the Nelson-Aalen estimator targets $H(t)$ and is preferable in the right tail where the KM step function is unstable. The hazard itself is never estimated directly without smoothing, because instantaneous rates require kernel methods or parametric assumptions.

11.14.4 Assumptions

A non-negative, almost surely continuous time-to-event with proper density (or, for discrete time, a discrete analogue with the same mutual recoverability).

11.14.5 R Implementation

```r
library(survival)

data(lung)
fit ← survfit(Surv(time, status) ~ 1, data = lung)

# Survival curve
plot(fit, conf.int = TRUE, main = "S(t)")

# Cumulative hazard
plot(fit, fun = "cumhaz", main = "H(t)")
```

11.14.6 Output & Results

Two complementary curves: the Kaplan-Meier survival function with point-wise confidence bands, and the cumulative hazard estimated via $-\log \hat{S}(t)$ (which agrees with Nelson-Aalen up to small-sample corrections). Reading both together makes the underlying hazard pattern visible — a steep $S(t)$ corresponds to a rising $H(t)$, and the slope of $H(t)$ at any time is the local hazard rate.

11.14.7 Interpretation

A reporting sentence: "Median survival was 310 days (95 % CI 280 to 360); cumulative hazard at one year was approximately 0.85, corresponding to $S(365) = \exp(-0.85) = 0.43$, consistent with the KM estimate of 42 % at one year." Quoting both representations communicates the same fact in two complementary forms readers may prefer.

11.14.8 Practical Tips

- Hazard is a rate, not a probability; never describe a hazard ratio as "doubled probability" without converting through $S(t)$ or absolute differences.
- Compare cumulative hazards across groups when discriminating risk over follow-up; the log-rank test is essentially a comparison of cumulative hazards.
- Constant hazard implies exponential survival and a linear cumulative hazard; deviations from linearity in $H(t)$ are the easiest visual diagnostic for non-exponentiality.
- Kaplan-Meier and Nelson-Aalen estimators agree closely; prefer Nelson-Aalen when you specifically need $\hat{H}(t)$ in the right tail.
- Hazard ratios are interpretable only under proportional hazards; if the assumption fails, report cumulative incidence, restricted-mean survival time, or stratify the comparison.

- For modelling a smooth hazard, use parametric (Weibull, log-normal) or flexible (Royston-Parmar splines) models; don't try to estimate $h(t)$ non-parametrically without bandwidth selection.

11.14.9 R Packages Used

`survival` for `survfit()`, `Surv()`, and the standard estimators; `flexsurv` for parametric and Royston-Parmar spline-based models that yield smooth hazards; `muhaz` for kernel-smoothed hazard estimation; `survminer::ggsurvplot()` and `survminer::ggsurvplot_combine()` for ggplot-style display of the curves.

11.14.10 For Reviewers

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

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

11.14.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.15 Interval-Censored Data

11.15.1 Introduction

Interval censoring arises whenever subjects are observed at discrete intervals and the exact event time is therefore unknown — only the interval that contains it is known. Cavity detected at a routine dental check-up, HIV seroconversion identified at a quarterly clinic visit, breakdown of a part discovered at periodic inspection: in each case the event time lies between the last "no event" inspection and the first "event observed" inspection. Treating these as exact event times biases standard survival methods; the Turnbull non-parametric maximum-likelihood estimator (NPMLE) and parametric or semi-parametric extensions handle the interval structure correctly.

11.15.2 Prerequisites

A working understanding of survival analysis, the distinction between right, left, and interval censoring, and basic Kaplan-Meier estimation as the gold-standard right-censored analogue.

11.15.3 Theory

Each subject contributes an interval $(L_i, R_i]$ within which the event occurred; right-censored subjects have $R_i = \infty$, exact events have $L_i = R_i$. Turnbull's algorithm finds the NPMLE survival function on a grid of unique left and right endpoints via an EM iteration: at each step it apportions probability mass across the candidate intervals proportional to current estimates and updates the survival estimate from those weighted contributions. Convergence yields the NPMLE up to indeterminacy in regions where the data provide no information (the so-called Turnbull intervals).

For regression, parametric models (Weibull, log-normal) and semi-parametric extensions (PH or PO with monotone splines on the baseline) accept interval-censored input directly. The likelihood is $\prod_i [S(L_i) - S(R_i)]$, which reduces to the standard product in the right-censored special case.

11.15.4 Assumptions

Independent interval censoring (the inspection schedule does not depend on the subject's underlying time-to-event), and accurate interval boundaries.

11.15.5 R Implementation

```r
library(icenReg); library(survival)

data(miceData)
head(miceData)

# NPMLE
fit_np ← ic_np(cbind(l, u) ~ grp, data = miceData)
plot(fit_np)

# Parametric: Weibull PH
fit_w ← ic_par(cbind(l, u) ~ grp, data = miceData, model = "ph", dist = "weibull")
summary(fit_w)
```

11.15.6 Output & Results

`ic_np()` returns the Turnbull NPMLE for each group with confidence bands and a step-function-style plot. `ic_par()` fits a parametric or semi-parametric regression with hazard-ratio interpretation under PH. The two complement each other: NPMLE for visual exploration, regression for covariate effects.

11.15.7 Interpretation

A reporting sentence: "Turnbull NPMLE showed lower survival in the contaminated group throughout follow-up (12-month survival 28 % vs 52 %); a Weibull PH model adjusted for sex estimated a hazard ratio of 2.1 (95 % CI 1.4 to 3.2)." When the NPMLE flatlines in regions with no information, describe these gaps in the caption rather than letting readers misinterpret the apparent constant survival.

11.15.8 Practical Tips

- `icenReg` is the most actively maintained interval-censored regression package in R; `ic_np()` for non-parametric, `ic_par()` for parametric, `ic_sp()` for semi-parametric with monotone splines.
- Exact event times in interval notation are encoded as $L = R$; right-censored subjects as $R = \infty$ or `Inf`. Most software accepts both encodings.
- The NPMLE has indeterminate intervals (regions of constant survival between Turnbull intervals); plot the NPMLE as a band rather than a single curve when visualising.
- For Cox-style proportional-hazards regression with interval-censored data, `survival::survreg(dist = "weibull")` accepts a `Surv(L, R, type = "interval2")` object directly.
- For very heavy interval censoring (long inspection intervals), consider sensitivity analyses with imputed event times at the midpoint or right endpoint and check that conclusions are robust.
- Standard Cox-Snell residuals do not directly apply; use modified residuals from `icenReg::diag_baseline()` for goodness-of-fit checks.

11.15.9 R Packages Used

`icenReg` for `ic_np()`, `ic_par()`, and `ic_sp()` interval-censored regression; `survival` for `Surv(..., type = "interval2")` and `survreg()` interval-censored support; `Icens` for additional NPMLE algorithms; `intccr` for interval-censored competing risks.

11.15.10 For Reviewers

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

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

11.15.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.16 Joint Longitudinal-Survival Models

11.16.1 Introduction

Joint longitudinal-survival models bring together two submodels — a mixed model for a longitudinally measured biomarker and a survival model for time to an event — via shared random effects so that the evolving biomarker is itself part of the hazard. This solves two problems simultaneously: the time-varying-covariate Cox model that uses last-observation-carried-forward biomarker values ignores measurement error and timing irregularity, and informative dropout (subjects censored because their biomarker is deteriorating) biases naive survival estimates. The joint formulation models both processes together and integrates over the shared random effects, yielding consistent hazard ratios for the underlying biomarker trajectory.

11.16.2 Prerequisites

A working understanding of linear mixed models, Cox regression, time-varying covariates, and the concept of informative dropout.

11.16.3 Theory

The longitudinal submodel is

$$y_{ij} = \mu_i(t_{ij}) + \varepsilon_{ij}, \qquad \mu_i(t) = X_{ij}^\top \gamma + Z_{ij}^\top b_i,$$

with $b_i \sim \mathcal{N}(0, D)$ subject-specific random effects. The survival submodel uses the latent trajectory:

$$h_i(t) = h_0(t) \exp\bigl(\alpha \mu_i(t) + x_i^\top \beta\bigr).$$

The association parameter α measures how strongly the underlying biomarker drives the hazard. The joint likelihood integrates the random effects out of both submodels simultaneously, typically via Gauss-Hermite quadrature; this is what makes joint models computationally heavier than fitting the two submodels separately.

11.16.4 Assumptions

Multivariate Normal random effects, correctly specified mean and covariance structures, no model misspecification in either submodel, and a proper proportional-hazards form (or a flexible alternative) for the survival component.

11.16.5 R Implementation

```
library(JM)

data(pbc2)
lm_fit ← lme(log(serBilir) ~ year + drug, random = ~ year | id,
             data = pbc2)
surv_fit ← coxph(Surv(years, status2) ~ drug, data = pbc2.id, x = TRUE)

joint_fit ← jointModel(lm_fit, surv_fit, timeVar = "year",
                       method = "piecewise-PH-aGH")
summary(joint_fit)
```

11.16.6 Output & Results

`summary(joint_fit)` returns the longitudinal-submodel coefficients (fixed effects, random-effect covariance), the survival-submodel coefficients (β), and the association parameter α with its standard error and 95 % confidence interval. Diagnostics include the Gauss-Hermite quadrature convergence criterion and the integrated likelihood.

11.16.7 Interpretation

A reporting sentence: "A joint longitudinal-survival model linking log-bilirubin trajectory to time to death (PBC dataset, 312 subjects) estimated the association parameter at $\alpha = 0.74$ ($\mathrm{SE} = 0.10$, $p < 0.001$); each unit increase in the underlying log-bilirubin level corresponded to a 2.1-fold higher hazard of death, after accounting for measurement error and informative dropout." Always describe both submodels in the methods so the joint structure is reproducible.

11.16.8 Practical Tips

- Use `JM` for classical maximum-likelihood joint models; `JMbayes2` for Bayesian alternatives that handle complex random-effect structures and missing data more flexibly.
- Joint models handle informative dropout — subjects censored because their biomarker is deteriorating — that biases naive Cox time-varying-covariate analyses.
- Computational cost scales rapidly with the number of random effects; start with random intercept and slope, escalate only if diagnostics demand it.
- Dynamic predictions ("given the biomarker history to time s, what is the predicted survival from s?") come from `survfitJM()` and are the headline clinical use case.
- Convergence diagnostics matter: check `joint_fit$convergence` and Gauss-Hermite node counts; raise `iter.qN` and `nQuad` if the model fails to converge.
- A simpler alternative is the time-varying Cox model with LOCF imputation; it loses efficiency and ignores measurement error but is easier to fit and explain to a non-statistical audience.

11.16.9 R Packages Used

`JM` for `jointModel()` with maximum-likelihood estimation; `JMbayes2` for Bayesian joint models with richer random-effect structures; `joineRML` for multivariate longitudinal-survival models with multiple biomarkers; `dynpred` for dynamic predictions when joint modelling is computationally infeasible.

11.16.10 For Reviewers

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

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

11.16.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.17 Kaplan-Meier Estimation

11.17.1 Introduction

The Kaplan-Meier (KM) estimator is the non-parametric maximum-likelihood estimator of the survival function from right-censored time-to-event data. Its signature step-function plot is the single most recognisable figure in clinical research. The estimator makes no distributional assumption about the time-to-event variable; it simply accumulates the conditional probability of surviving each observed event time.

11.17.2 Prerequisites

The reader should understand that a survival analysis studies time until an event, and that "right censoring" means some subjects are observed only up to a point before their event has occurred. Familiarity with R's `survival` package is not required but helps.

11.17.3 Theory

Let $t_1 < t_2 < \ldots < t_k$ denote the distinct event times observed in the data. At time t_j, let d_j be the number of events and n_j the number of subjects still at risk (not yet having had the event and not yet censored). The Kaplan-Meier estimator of the survival function is

$$\hat{S}(t) = \prod_{j : t_j \le t} \left(1 - \frac{d_j}{n_j}\right).$$

The estimator is a step function that drops at each event time by the fraction of the at-risk set experiencing the event. Censored subjects contribute to the risk set up to their censoring time but do not cause drops.

The standard-error estimator due to Greenwood is

$$\widehat{\mathrm{Var}}[\hat{S}(t)] = \hat{S}(t)^2 \sum_{j : t_j \le t} \frac{d_j}{n_j (n_j - d_j)}.$$

Pointwise confidence intervals are typically constructed on the complementary log-log scale to ensure they lie in $[0, 1]$.

The **log-rank test** compares survival between two or more groups. It sums the observed-minus-expected events at each event time across groups, yielding a test statistic that follows a χ^2_{k-1} distribution under the null hypothesis of equal survival functions.

11.17.4 Assumptions

Kaplan-Meier estimation assumes:

1. Censoring is **non-informative** – subjects who are censored at time t have the same future risk as those who remain under observation. Violated when, for example, patients drop out because they feel worse.
2. Subjects are independent.
3. Risk of event does not depend on calendar time beyond what covariates capture (important when accrual spans a long period).

11.17.5 R Implementation

```r
library(survival)
library(survminer)

data(lung, package = "survival")

fit <- survfit(Surv(time, status) ~ sex, data = lung)

print(fit)

ggsurvplot(fit,
           data = lung,
           conf.int = TRUE,
           pval = TRUE,
           risk.table = TRUE,
           legend.labs = c("Male", "Female"),
           xlab = "Days from diagnosis",
           ylab = "Survival probability",
           palette = "Set2")
```

The `Surv()` call packages the time and event-indicator columns into a single survival object; `survfit()` with a right-hand side of `sex` produces separate KM curves for each sex. `ggsurvplot()` draws the curves with pointwise confidence bands, a log-rank p-value, and a risk table below.

11.17.6 Output & Results

The `lung` dataset has 138 male and 90 female patients with advanced lung cancer. The KM median survival is 270 days for males (95% CI 212 to 310) and 426 days for females (95% CI 348 to 550). The log-rank test gives $\chi^2_1 = 10.3$, $p = 0.0013$, indicating a significantly higher survival for females.

11.17.7 Interpretation

Reporting follows the CONSORT conventions: "Median overall survival was 270 days (95% CI 212-310) for males and 426 days (95% CI 348-550) for females; log-rank $p = 0.001$ (Figure X)." The figure should always be shown with a risk table beneath it – the curve is nearly meaningless once the number at risk drops below a handful.

11.17.8 Practical Tips

- Always include a risk table. Survival curves at late follow-up with only 2 or 3 subjects remaining can swing wildly and mislead readers.
- Do not rely on the median alone as a comparison statistic; if the curves cross, the median can flip direction as the study matures.
- The log-rank test is most powerful when the proportional-hazards assumption holds. For curves that cross or diverge late, consider the Renyi (weighted) log-rank or a restricted mean survival time comparison.
- Do not attempt a KM estimate without an event in the at-risk set; the curve will simply be flat at 1.
- When follow-up is short relative to the event rate, many subjects will be censored and the KM estimate at late times will have wide confidence bands. Report the cumulative event count at a clinically relevant landmark (1-year, 5-year) rather than the tail of the curve.

11.17.9 For Reviewers

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

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

11.17.10 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias

- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.18 Landmark Analysis

11.18.1 Introduction

Landmark analysis is the standard fix for the immortal-time bias that arises when subjects are classified by a post-baseline exposure (treatment response, biomarker change, secondary event). Naively grouping subjects by a covariate measured after follow-up begins assigns "responder" status only to subjects who survived long enough to be assessed, automatically biasing comparisons in their favour. Landmark analysis sidesteps this by selecting a fixed landmark time, restricting to subjects alive and observable at that time, classifying them by covariate value *at the landmark*, and then analysing only the post-landmark survival.

11.18.2 Prerequisites

A working understanding of Cox regression, time-varying covariates, and the immortal-time bias that makes naive responder-vs-nonresponder analyses invalid.

11.18.3 Theory

At a chosen landmark time L:

1. Exclude subjects with an event before L — they cannot be classified by status at L.
2. Classify each remaining subject by their covariate value at L.
3. Treat L as the new time origin and analyse the residual survival $T_i - L$ for $T_i > L$.
4. Fit a standard Cox model on the post-landmark data.

Multiple landmarks can be combined into a "super model" with `id`-clustered variance and a smooth function of L as a covariate, gaining efficiency at the cost of additional model complexity (Van Houwelingen 2007).

11.18.4 Assumptions

The covariate value at landmark L is the relevant exposure for prognosis; sufficient follow-up beyond L remains; and the censoring mechanism after L is non-informative.

11.18.5 R Implementation

```r
library(survival)

set.seed(2026)
n <- 200
d <- data.frame(
  id = 1:n,
  time = rexp(n, 0.1),
  status = rbinom(n, 1, 0.6),
  response = rbinom(n, 1, 0.5)      # binary time-varying covariate
)

L <- 6   # landmark at 6 months

# Subjects at risk at L
at_risk <- d[d$time >= L, ]
at_risk$time_from_L <- at_risk$time - L
at_risk$status_from_L <- at_risk$status

fit <- coxph(Surv(time_from_L, status_from_L) ~ response, data = at_risk)
summary(fit)
```

11.18.6 Output & Results

A standard Cox regression on the post-landmark cohort: hazard ratio for the covariate (here `response`) measured at the landmark, with 95 % confidence interval and Wald test. The number of events and at-risk subjects shrinks after the landmark, so confidence intervals are wider than in a full-cohort analysis.

11.18.7 Interpretation

A reporting sentence: “Among the 152 patients alive at the six-month landmark, treatment responders (n = 78) had a 50 % lower hazard of subsequent death than non-responders (HR 0.50, 95 % CI 0.30 to 0.83); the analysis avoids the immortal-time bias inherent in unrestricted responder-vs-non-responder comparisons.” Always state the landmark time and the post-landmark sample size.

11.18.8 Practical Tips

- Choose the landmark time at a clinically meaningful assessment point (six-week imaging response, three-month biomarker re-measurement); arbitrary times can produce unstable estimates.
- Multiple landmarks (super-model approach) combine information across time and provide a smooth treatment-effect estimate; cluster the variance on subject id when fitting.
- Sample size drops at the landmark; power calculations should account for the post-landmark cohort, not the original.
- Compare landmark results with a full time-varying Cox model that uses the covariate as a `tt()` term; both should agree if the landmark choice is reasonable.
- Be transparent about exclusions: report how many subjects had events before L and were excluded, because this is the headline difference from a naive analysis.

- For complex time-varying patterns, consider joint longitudinal-survival models (`JM` or `joineRML`) instead of repeated landmark analyses.

11.18.9 R Packages Used

`survival` for the standard Cox fit on landmark cohorts; `dynpred` for super-model landmark analyses with smooth time effects; `JM` and `joineRML` for joint longitudinal-survival modelling that handles continuous time-varying covariates without landmarks; `mstate` for multi-state extensions.

11.18.10 For Reviewers

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

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

11.18.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.19 Left Truncation

11.19.1 Introduction

Left truncation, also called delayed entry, occurs when subjects become observable only after surviving to a certain age or calendar time. Recruiting middle-aged adults into a cohort study means anyone who died as a child or young adult is structurally absent — they could not contribute. Treating delayed-entry data as if every subject had been followed from time zero overestimates survival, because the early period is observed only for the survivors. The counting-process formulation in `survival::Surv(tstart, tstop, event)` handles this naturally by restricting risk sets at each event time to subjects actually under observation then.

11.19.2 Prerequisites

A working understanding of survival analysis, the difference between censoring (subject leaves before event) and truncation (subject only enters under specific conditions), and the counting-process formulation of survival data.

11.19.3 Theory

For a left-truncated subject who enters at time τ_i and exits at time T_i (with event indicator δ_i), the contribution to the partial likelihood at event time t should appear only when $\tau_i < t \leq T_i$. The standard $\mathrm{Surv}(\mathrm{time}, \mathrm{event})$ object does not encode the entry time τ_i and silently sets it to zero, biasing estimates upward. The two-argument form $\mathrm{Surv}(\tau, T, \mathrm{event})$ (counting-process style) carries both endpoints; `survfit()` and `coxph()` use them to construct correct risk sets.

In age-as-timescale analyses (the natural framing in chronic-disease epidemiology), every subject is left-truncated at their age at enrolment.

11.19.4 Assumptions

The truncation mechanism is independent of the event of interest given covariates: subjects do not become observable for reasons related to their underlying risk profile.

11.19.5 R Implementation

```
library(survival)

# Age-as-timescale cohort: enrolled between ages 40 and 70
set.seed(2026)
n ← 200
age_enroll ← runif(n, 40, 70)
event_age  ← age_enroll + rexp(n, 0.05)
end_age    ← pmin(event_age, age_enroll + runif(n, 5, 20))
status     ← as.integer(event_age ⩽ end_age)

# Left-truncated Cox: tstart = entry age, tstop = exit age
fit ← coxph(Surv(age_enroll, end_age, status) ~ 1)
summary(fit)

# Kaplan-Meier with delayed entry
km ← survfit(Surv(age_enroll, end_age, status) ~ 1)
plot(km)
```

11.19.6 Output & Results

`survfit()` returns a Kaplan-Meier curve on the chosen time scale (age, calendar year, or time-since-event) that conditions on each subject's entry time. The Cox model with the counting-process `Surv` object produces unbiased hazard-ratio estimates even when entry times are heavily truncated.

11.19.7 Interpretation

A reporting sentence: "Using age as the time scale with left truncation at enrolment, the conditional survival from age 40 was 0.75 at age 60 (95 % CI 0.68 to 0.82); ignoring delayed entry inflates this estimate to 0.84 by treating mid-life enrolees as if they had been followed since age 0." Always state the time scale and the truncation mechanism in the methods section.

11.19.8 Practical Tips

- Always specify both `tstart` and `tstop` arguments to `Surv()` when entry times vary across subjects; the single-argument form silently assumes all subjects entered at zero.
- Age-as-timescale cohort studies are always left-truncated; this is the most common scenario in chronic-disease epidemiology and ageing research.
- Distinguish left truncation (subject only observable after a certain time) from left censoring (event occurred before observation began but the subject is observable now); the two are handled differently.
- Without left-truncation correction, you overestimate survival of cohorts enrolled at older ages because their early-life mortality is invisible to your data.
- `survfit()` with delayed entry produces correct conditional survival curves; check that the reported time scale starts at the minimum entry time, not zero.
- For competing risks with delayed entry, `cmprsk::cuminc` and `tidycmprsk::cuminc` accept counting-process input via `entry` arguments.

11.19.9 R Packages Used

`survival` for `Surv(tstart, tstop, event)` and `survfit()` / `coxph()` with counting-process input; `Epi` for left-truncation-aware tools targeted at epidemiology; `etm` for non-parametric multi-state estimation with delayed entry; `tidycmprsk` for left-truncation-aware competing-risks analysis.

11.19.10 For Reviewers

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

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

11.19.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.20 Log-Rank Test: Details

11.20.1 Introduction

The log-rank test is the standard non-parametric test for comparing survival across two or more groups in randomised trials and observational studies. It accumulates differences between observed and expected event counts at each event time across the follow-up window and reports a chi-squared statistic. The test is most powerful under proportional hazards — the alternative for which it was designed — but variants (weighted log-rank, stratified log-rank) extend its applicability to non-proportional or covariate-adjusted settings.

11.20.2 Prerequisites

A working understanding of Kaplan-Meier estimation, censoring, and the proportional-hazards alternative as the natural target for the log-rank test.

11.20.3 Theory

At each unique event time t_j across all groups, the test compares the observed number of events in group k to the expected number under the null of equal hazards: $E_{kj} = d_j \cdot n_{kj}/n_j$, with d_j the total events at t_j and n_{kj}, n_j the numbers at risk. The standardised cumulative deviation across all event times forms a chi-squared statistic with $k-1$ degrees of freedom, where k is the number of groups.

Weighted variants (Wilcoxon, Tarone-Ware, Fleming-Harrington $G^{\rho,\gamma}$) give more weight to early or late differences and recover power when hazards are non-proportional. The stratified log-rank applies the test within strata defined by a categorical covariate and pools the contributions, controlling for that covariate while testing the main effect.

11.20.4 Assumptions

Proportional hazards across groups (for the standard log-rank), independent and non-informative censoring, and accurate event-time recording.

11.20.5 R Implementation

```r
library(survival)

data(lung)
fit <- survdiff(Surv(time, status) ~ sex, data = lung)
fit

# Stratified log-rank
survdiff(Surv(time, status) ~ sex + strata(ph.ecog), data = lung)
```

11.20.6 Output & Results

`survdiff()` returns observed and expected events per group, the chi-squared test statistic, degrees of freedom, and the asymptotic p-value. The stratified version partitions the comparison within levels of the strata variable, producing the same output structure but with the stratification controlled for.

11.20.7 Interpretation

A reporting sentence: "The log-rank test rejected the null of equal survival between sexes ($\chi^2_1 = 10.3$, $p = 0.001$); the female arm had 53 observed events compared with 86 expected under the null, while the male arm had 112 observed compared with 79 expected." Always pair the test with a hazard-ratio estimate (from Cox) and Kaplan-Meier curves; the test alone gives no effect size.

11.20.8 Practical Tips

- Use stratification (`+ strata(var)`) when controlling for a covariate without testing it; stratified log-rank tests within strata and pools.
- When hazards cross, the standard log-rank can fail to detect a real difference; use the Fleming-Harrington $G^{1,0}$ test for late differences or $G^{0,1}$ for early ones.
- For many groups, log-rank gives an omnibus test; for pairwise comparisons follow up with `pairwise_survdiff()` from `survminer` and apply Bonferroni or Hochberg correction.
- Report Kaplan-Meier curves alongside the test result; a p-value without a curve is uninterpretable for clinical readers.
- Non-informative censoring is the assumption that fails most often in observational studies; if censoring depends on prognosis, the log-rank test is biased.
- For severely non-proportional hazards (clearly crossing curves), consider RMST as the primary summary instead of relying on the log-rank.

11.20.9 R Packages Used

`survival` for `survdiff()` with stratification; `survminer::pairwise_survdiff()` for pairwise log-rank tests with multiple-comparison correction; `nph` and `coin` for weighted-log-rank variants and exact tests; `FHtest` for Fleming-Harrington $G^{\rho,\gamma}$ tests.

11.20.10 For Reviewers

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

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

11.20.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.21 Multi-State Models

11.21.1 Introduction

A multi-state model describes a subject's movement through a discrete set of states over time — healthy to diseased to dead, pre-transplant to post-transplant to rejected, or any clinical pathway with intermediate landmarks. It generalises ordinary survival (two states: alive, dead), competing risks (one origin, several absorbing states), and the illness-death model (one intermediate, one absorbing state). The framework gives a complete probabilistic picture of the disease course rather than collapsing it into a single time-to-first-event summary.

11.21.2 Prerequisites

A working understanding of Cox regression, competing-risks analysis, and the concept of transition intensities (state-specific hazards) as the multi-state generalisation of a single survival hazard.

11.21.3 Theory

A multi-state model represents states as nodes and possible transitions as directed edges. Each transition $j \to k$ has an associated transition intensity (hazard) $\alpha_{jk}(t)$. Transition-specific Cox models estimate covariate effects on each α_{jk} separately, with stratification by transition number ensuring each baseline can differ. Predicted transition probabilities $P_{jk}(s,t)$ — the probability of being in state k at time t given state j at time s — are obtained from the cumulative-hazard matrix via the product-integral, implemented in `mstate::probtrans()`.

The Markov property assumes the future depends only on the current state and time, not on the path taken to reach it; semi-Markov extensions condition on time-since-entry-to-current-state instead.

11.21.4 Assumptions

Markov property (often, with semi-Markov extensions available), proportional hazards within each transition (or relaxed via spline-based or stratified models), and accurate state classification at each time point.

11.21.5 R Implementation

```r
library(mstate)

# Example: illness-death model (3 states)
# Load example data
data(ebmt3)

tmat <- trans.illdeath()     # transition matrix
d_ms <- msprep(data = ebmt3, trans = tmat,
               time = c(NA, "prtime", "rfstime"),
               status = c(NA, "prstat", "rfsstat"))

# Fit Cox models per transition
fit <- coxph(Surv(Tstart, Tstop, status) ~ dissub + age + strata(trans),
             data = d_ms)
summary(fit)
```

11.21.6 Output & Results

`msprep()` reshapes wide-format clinical data into the transition-stratified long format that multi-state Cox models expect. The Cox fit produces transition-specific coefficients (when interactions with `strata(trans)` are included). `mstate::msfit()` and `mstate::probtrans()` deliver predicted cumulative hazards per transition and transition probabilities $P_{jk}(s, t)$ at any horizon.

11.21.7 Interpretation

A reporting sentence: “Increasing age was associated with elevated transition hazards from healthy to relapsed (HR 1.02 per year, 95 % CI 1.01 to 1.04) and from healthy to death (HR 1.04, 95 % CI 1.02 to 1.06); the predicted probability of being in the relapsed state at five years was 28 % for a 30-year-old patient and 41 % for a 60-year-old.” Predictions in absolute probabilities communicate multi-state results far better than the underlying transition-specific hazard ratios alone.

11.21.8 Practical Tips

- Draw the state diagram before any analysis; a clear picture of allowed transitions prevents misspecified transition matrices.
- `msprep()` converts wide patient-level data into the long format with one row per allowed transition per subject; this is the most common entry barrier for new users.
- Use `probtrans()` predictions to communicate results in absolute terms — readers absorb “20 % probability of death by year 5” much better than three transition hazard ratios.
- The Markov vs semi-Markov choice matters for sojourn-time-dependent hazards; `mstate` supports both via the `Tstart` argument and time-origin handling.
- For high-dimensional state spaces, sparse-state extensions in `msm` (continuous-time Markov chains for panel data) are often more tractable than full-likelihood multi-state models.
- Validate predictions on held-out data; multi-state Cox models can overfit transition-specific hazards when individual transitions have few events.

11.21.9 R Packages Used

`mstate` for the `msprep` / `msfit` / `probtrans` workflow with multi-state Cox models; `msm` for continuous-time Markov chain models on panel-observed states; `flexsurv::msfit.flexsurvreg` for parametric multi-state extensions; `etm` for non-parametric Aalen-Johansen-style transition probability estimation.

11.21.10 For Reviewers

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

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

11.21.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.22 The Nelson-Aalen Estimator

11.22.1 Introduction

The Nelson-Aalen estimator is the standard non-parametric estimator of the cumulative hazard $H(t)$ for right-censored data. It is the natural counterpart of the Kaplan-Meier estimator: where KM directly targets the survival function, Nelson-Aalen targets the cumulative hazard, and the two are linked by $S(t) = \exp(-H(t))$. Nelson-Aalen has slightly better small-sample properties than Kaplan-Meier in the right tail and is the natural input for any analysis framed in terms of hazards rather than survival probabilities, including diagnostics for proportional hazards across groups.

11.22.2 Prerequisites

A working understanding of cumulative hazards, the Kaplan-Meier estimator, and the relationship $S(t) = \exp(-H(t))$.

11.22.3 Theory

The Nelson-Aalen estimator is

$$\hat{H}(t) = \sum_{t_i \le t} \frac{d_i}{n_i},$$

with d_i the number of events at t_i and n_i the number at risk just before t_i. Each event time contributes its empirical hazard increment d_i/n_i, and the cumulative hazard is the running sum. The variance estimator is

$$\widehat{\mathrm{Var}}\big(\hat{H}(t)\big) = \sum_{t_i \le t} d_i / n_i^2,$$

which gives pointwise confidence bands on the log-log scale. The Nelson-Aalen-based survival estimator $\hat{S}(t) = \exp(-\hat{H}(t))$ differs slightly from the Kaplan-Meier product-limit estimator and is preferred when the cumulative-hazard scale is the primary reporting target.

11.22.4 Assumptions

Independent, non-informative censoring. Adequate at-risk sets across the follow-up window — the right tail of the estimator has wide variance when only a handful of subjects remain.

11.22.5 R Implementation

```
library(survival)

data(lung)
fit ← survfit(Surv(time, status) ~ 1, data = lung)

# Cumulative hazard via NA
plot(fit, fun = "cumhaz", main = "Nelson-Aalen cumulative hazard")

# Underlying values
summary(fit)$cumhaz ▷ head()
```

11.22.6 Output & Results

`survfit()` with `fun = "cumhaz"` plots the Nelson-Aalen cumulative hazard with pointwise confidence bands. `summary(fit)$cumhaz` returns the underlying numeric values; the corresponding event times come from `summary(fit)$time`.

11.22.7 Interpretation

A reporting sentence: “The Nelson-Aalen cumulative hazard reached approximately 1.5 by day 800 (95 % CI 1.3 to 1.7), corresponding to a survival probability of $\exp(-1.5) = 0.22$ at that time; the slope of $\hat{H}(t)$ in the first 200 days indicates an instantaneous hazard of approximately 0.0006 per day.” Always pair the cumulative-hazard estimate with the implied survival probability for non-statistical readers.

11.22.8 Practical Tips

- Use Nelson-Aalen for cumulative-hazard reporting and diagnostic comparisons; use Kaplan-Meier for survival probabilities.

- Compare $\log \hat{H}(t)$ across groups: parallel lines indicate proportional hazards (the Cox-model assumption); divergent or crossing lines indicate non-proportionality.
- Construct confidence bands on the log-log scale or via $H \pm z_{\alpha/2} \cdot \mathrm{SE}(H)$; the log-log version respects the non-negativity of cumulative hazard.
- For Cox regression baseline hazard estimation, Breslow's estimator is the slight variant used internally; both reduce to Nelson-Aalen when no covariates are present.
- For very small at-risk sets at late times, the estimator's variance balloons; interpret the right tail cautiously and consider truncating the plot at the last well-sampled time.
- For competing risks, the cumulative-cause-specific hazard is the obvious analogue; the Aalen-Johansen estimator generalises Nelson-Aalen to multi-state processes.

11.22.9 R Packages Used

`survival` for `survfit()` with `fun = "cumhaz"`; `mstate` for multi-state extensions including the Aalen-Johansen estimator; `etm` for the empirical transition matrix; `ggsurvfit` for ggplot-flavoured Nelson-Aalen plots integrated with downstream tidyverse work.

11.22.10 For Reviewers

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

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

11.22.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.23 Parametric Survival: Log-Normal

11.23.1 Introduction

The log-normal survival model assumes that $\log T$ follows a Normal distribution. Its distinctive feature among parametric survival models is a *non-monotone* hazard — the instantaneous risk rises early in follow-up, peaks, and then declines. This shape suits diseases or processes in which susceptible individuals fail relatively quickly while a hardy subset survives indefinitely: for example, post-surgical mortality (early peri-operative risk

that wanes), some cancer recurrences, and time-to-engagement processes in behavioural data. Where Weibull and exponential models lock in monotone hazards, log-normal accommodates the rise-and-fall pattern naturally.

11.23.2 Prerequisites

A working understanding of the log-normal distribution, parametric survival regression, and the accelerated-failure-time (AFT) parameterisation as an alternative to the proportional-hazards form.

11.23.3 Theory

The log-normal survival function is

$$S(t) = 1 - \Phi\left(\frac{\log t - \mu}{\sigma}\right),$$

with hazard

$$h(t) = \frac{(1/(t\sigma))\phi((\log t - \mu)/\sigma)}{1 - \Phi((\log t - \mu)/\sigma)}.$$

The hazard has a unique maximum near $t \approx \exp(\mu - \sigma^2)$ for $\sigma > 1$ and decreases thereafter. The AFT parameterisation $\log T = X\beta + \sigma W$ (with $W \sim \mathcal{N}(0,1)$) makes regression coefficients interpretable as multiplicative effects on the *event time* — a one-unit increase in a covariate scales the time scale by $\exp(-\beta)$.

11.23.4 Assumptions

A non-monotone (rising-then-falling) hazard, log-Normal residual structure on the log time scale, independent observations, and non-informative censoring.

11.23.5 R Implementation

```
library(survival); library(flexsurv)

data(lung)
fit ← survreg(Surv(time, status) ~ age + sex, data = lung, dist = "lognormal")
summary(fit)

# flexsurv
flex ← flexsurvreg(Surv(time, status) ~ age + sex, data = lung, dist = "lnorm")
plot(flex, type = "hazard")
```

11.23.6 Output & Results

`survreg()` returns AFT coefficients on the log-time scale plus the scale parameter σ. `flexsurvreg()` produces equivalent estimates with cleaner survival, hazard, and quantile prediction methods. `plot(flex, type = "hazard")` draws the model-implied hazard curve, making the rise-peak-decline shape visible against KM-based hazard estimates.

11.23.7 Interpretation

A reporting sentence: "A log-normal AFT model estimated that each additional year of age shortened expected event time by approximately 2 % ($\hat{\beta}_{\text{age}} = -0.02$, 95 % CI -0.04 to -0.00); the model-implied hazard peaked at approximately 9 months and declined thereafter, consistent with the observed concentration of events in the first year of follow-up." Always plot the implied hazard alongside Kaplan-Meier to verify the non-monotone shape is supported by the data.

11.23.8 Practical Tips

- Use log-normal when Kaplan-Meier or non-parametric hazard estimates show a clear "hump" of events early in follow-up; for monotone hazards, Weibull or exponential are simpler and more efficient.
- Log-logistic is a close alternative with similar non-monotone hazard but heavier tails; compare AICs across both before committing.
- AIC comparison across parametric families (exponential, Weibull, log-normal, log-logistic, generalised gamma) is the standard model-selection workflow; the generalised gamma nests several of these.
- For very flexible hazards beyond the log-normal shape, Royston-Parmar spline-based models offer more freedom while remaining parametric.
- Predictions on the natural time scale come via `predict(fit, type = "quantile")` for `survreg` or `summary(flex, type = "quantile")` for `flexsurv` objects.
- When some covariates produce monotone hazards and others produce non-monotone, consider a heterogeneous-effects extension or a flexible spline model rather than forcing one parametric family.

11.23.9 R Packages Used

`survival` for `survreg()` with `dist = "lognormal"`; `flexsurv` for `flexsurvreg()` with cleaner prediction interfaces and rich plotting; `eha` for additional parametric hazard models; `rstpm2` for Royston-Parmar spline-based parametric extensions when the log-normal hazard shape is too restrictive.

11.23.10 For Reviewers

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

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

11.23.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models

- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.24 Parametric Survival: Weibull

11.24.1 Introduction

The Weibull distribution is the most flexible parametric survival family that has a monotone hazard. Hazard is increasing when the shape parameter $k > 1$, constant (exponential) when $k = 1$, and decreasing when $k < 1$. Among the common parametric families it is unique in admitting both proportional-hazards and accelerated-failure-time interpretations, which makes it the natural starting point for parametric survival analysis whenever PH is plausible. Cleaner extrapolation, smaller standard errors, and explicit hazard shape are the practical pay-offs over the semi-parametric Cox model when its assumptions hold.

11.24.2 Prerequisites

A working understanding of the Weibull distribution, survival analysis, and the distinction between proportional-hazards and accelerated-failure-time parameterisations.

11.24.3 Theory

The Weibull survival function is

$$S(t) = \exp\bigl(-(t/\lambda)^k\bigr)\,,$$

with hazard $h(t) = (k/\lambda)(t/\lambda)^{k-1}$. The hazard is monotone: increasing for $k > 1$ (positive ageing), decreasing for $k < 1$ (early-failure regimes), and constant for $k = 1$ (exponential).

Two parameterisations coexist:

- **Accelerated-failure-time (AFT)** in `survreg(dist = "weibull")`: $\log T = X\beta + \sigma W$ with W a standard extreme-value variate; the scale is $\sigma = 1/k$.
- **Shape-scale** in `flexsurv::flexsurvreg(dist = "weibull")`: explicit shape k and scale λ, with PH-style covariate effects via interactions on λ.

The conversion between PH and AFT forms is $\beta_{\text{PH}} = -k \cdot \beta_{\text{AFT}}$.

11.24.4 Assumptions

A monotone hazard. The standard diagnostic is the log–log survival plot: $\log[-\log \hat{S}(t)]$ against $\log t$ should be approximately linear with slope k if the Weibull holds.

11.24.5 R Implementation

```
library(survival); library(flexsurv)

data(lung)

fit_survreg ← survreg(Surv(time, status) ~ age + sex, data = lung,
                      dist = "weibull")
summary(fit_survreg)

# flexsurvreg gives shape and scale directly
fit_flex ← flexsurvreg(Surv(time, status) ~ age + sex, data = lung,
                       dist = "weibull")
fit_flex
```

11.24.6 Output & Results

`survreg()` returns AFT coefficients (interpret as multiplicative effects on event time, $\exp(-\beta)$) plus the scale parameter. `flexsurvreg()` returns shape, scale, and PH-style hazard ratios directly — easier to communicate to a Cox-fluent audience. Both yield the same likelihood and the same fitted survival.

11.24.7 Interpretation

A reporting sentence: "A Weibull AFT model estimated shape parameter $\hat{k} = 1.31$ (95 % CI 1.10 to 1.55), indicating an increasing hazard over follow-up; each additional year of age accelerated the event by approximately 2 %, equivalent to a hazard ratio of 1.025 per year on the PH scale." When reporting, choose AFT or PH consistently with your audience: clinical readers expect HRs, while time-to-event-cost analyses prefer time-acceleration ratios.

11.24.8 Practical Tips

- Use the log–log survival plot (`plot(survfit(...), fun = "cloglog")`) as the main diagnostic for Weibull suitability; clear non-linearity rules it out.
- For more flexible monotone hazards, compare AIC against generalised gamma and Gompertz; for non-monotone, switch to log-normal or Royston-Parmar splines.
- Convert between PH and AFT carefully: $\beta_{\text{PH}} = -k \cdot \beta_{\text{AFT}}$, and remember that AFT confidence intervals are on the log-time scale.
- For small samples, Weibull yields smoother survival estimates than Kaplan-Meier and lower-variance covariate effects than Cox; this is its main practical advantage over Cox.
- `flexsurv` simplifies prediction (`summary(fit, type = "survival", t = ...)`) and supports time-dependent shape parameters via spline extensions.
- Check Weibull plus Cox HR estimates; they should be close under PH. Material disagreement points to an assumption failure that should be investigated.

11.24.9 R Packages Used

`survival` for `survreg()` with `dist = "weibull"`; `flexsurv` for `flexsurvreg()` with shape/scale output and richer prediction; `eha` for additional parametric extensions; `rstpm2` for Royston-Parmar spline alternatives when monotone hazards are too restrictive.

11.24.10 For Reviewers

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

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

11.24.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.25 Recurrent Events

11.25.1 Introduction

Recurrent-event analysis applies whenever a single subject can experience the event of interest multiple times during follow-up — hospital admissions, seizures, infections, asthma exacerbations, recurrent tumours. Treating only the first event collapses information and wastes power; treating all events as independent inflates effective sample size and produces invalid standard errors. Several modelling frameworks balance these extremes, each suited to a different scientific question.

11.25.2 Prerequisites

A working understanding of Cox regression, counting-process data formats, and within-subject correlation as a source of dependent observations.

11.25.3 Theory

The main approaches:

- **Andersen-Gill (AG)**: a counting-process Cox model that treats all events equally and uses cluster-robust standard errors to account for within-subject dependence. Estimates a single baseline hazard for the recurrent process.

- **Prentice-Williams-Peterson (PWP)**: stratifies by event number, allowing separate baseline hazards for the first, second, third, etc. event. Two flavours: total time (gap from study start) and gap time (gap from previous event). Use when the hazard structure changes with event order.
- **Wei-Lin-Weissfeld (WLW)**: fits a separate marginal Cox model for each event order, using only the first event for $K = 1$, the first two for $K = 2$, etc. Conceptually cleaner but increasingly out of favour because it conditions on having survived enough events to be at risk.
- **Shared-frailty**: a Cox model with a subject-level random effect u_i that captures unmeasured individual heterogeneity. Combine with AG when frailty is the natural interpretation of within-subject correlation.

11.25.4 Assumptions

The chosen correlation structure matches the data, the hazard specification (PH within strata or marginally) is appropriate, and counting-process intervals are correctly constructed (no overlaps, no gaps).

11.25.5 R Implementation

```r
library(survival)

# Counting-process data: id, tstart, tstop, status (1 = event)
data(cgd, package = "survival")

# Andersen-Gill
ag ← coxph(Surv(tstart, tstop, status) ~ treat + inherit,
           data = cgd, cluster = id)
summary(ag)

# PWP stratified by event number
cgd$event.num ← ave(cgd$status, cgd$id, FUN = cumsum) + 1
pwp ← coxph(Surv(tstart, tstop, status) ~ treat + inherit + strata(event.num),
            data = cgd, cluster = id)
summary(pwp)
```

11.25.6 Output & Results

Both fits return hazard-ratio estimates with cluster-robust standard errors. AG produces a single HR per covariate covering the full recurrent-event process; PWP produces strata-specific baselines while sharing covariate effects across strata, allowing different baseline hazards per event order.

11.25.7 Interpretation

A reporting sentence: “An Andersen-Gill counting-process Cox model estimated that prophylactic treatment reduced the recurrent-infection hazard by 60 % (HR 0.40, 95 % CI 0.25 to 0.63), with cluster-robust SEs accounting for within-subject correlation; the PWP

model stratified by event order produced a consistent estimate (HR 0.43, 95 % CI 0.26 to 0.70), suggesting the treatment effect does not vary materially across event order." Compare AG and PWP whenever the data structure permits.

11.25.8 Practical Tips

- AG is the default starting point for most applications and the easiest to interpret.
- PWP (gap-time) is appropriate when the hazard depends on time since the previous event; PWP (total time) when it depends on time since study start.
- WLW has fallen out of favour because it conditions on event order in a way that produces awkward causal interpretations.
- For unmeasured individual heterogeneity not captured by covariates, add a shared frailty term to the AG model (`+ frailty(id)`).
- Always use cluster-robust SEs (`cluster = id`); ignoring within-subject correlation produces SEs that are too small and Type I error inflated.
- Pair hazard ratios with absolute counts of events, expected rates, and cumulative mean functions; recurrent-event audiences typically reason in absolute frequencies rather than hazard ratios.

11.25.9 R Packages Used

`survival` for `coxph()` with `cluster()` and `strata()`; `frailtypack` for joint frailty models with terminal events and recurrent events fitted simultaneously; `reReg` for additional recurrent-event methods including marginal mean models; `mets` for clustered survival data with flexible dependence structures.

11.25.10 For Reviewers

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

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

11.25.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.26 Restricted Mean Survival Time (RMST)

11.26.1 Introduction

The restricted mean survival time (RMST) is the expected survival time truncated at a clinically meaningful horizon τ. Where the hazard ratio from a Cox model summarises a relative effect that requires proportional hazards to be interpretable, RMST quantifies an absolute time difference (in days, months, or years) and remains valid even when the hazard ratio is not constant. It has become an increasingly standard endpoint in oncology trials and immunotherapy studies, where crossing or delayed survival curves are common and where regulators and clinicians want a number expressed on the natural time scale.

11.26.2 Prerequisites

A working understanding of survival functions, the Kaplan-Meier estimator, and the issues with proportional-hazards assumptions in modern clinical trials.

11.26.3 Theory

For a survival function $S(t)$ and a horizon τ, the RMST is

$$\mathrm{RMST}(\tau) = \mathbb{E}[\min(T, \tau)] = \int_0^\tau S(t)\,\mathrm{d}t.$$

Geometrically it is the area under the KM curve from time zero to τ. Group differences are summarised by the difference $\mathrm{RMST}_1(\tau) - \mathrm{RMST}_0(\tau)$ or by the ratio. Standard errors come from the asymptotic theory developed by Royston and Parmar; bootstrap confidence intervals are an alternative when the asymptotic approximation is suspect. Crucially, the construction makes no assumption about the relationship between hazards in the two groups — non-proportionality, treatment-effect waning, and crossing curves are all handled gracefully.

11.26.4 Assumptions

Sufficient follow-up to estimate $S(t)$ reliably up to τ; an explicit choice of τ that is clinically meaningful and within the support of the data; non-informative censoring up to τ.

11.26.5 R Implementation

```r
library(survRM2)

data(lung)

# Truncation at 600 days
rmst2(time = lung$time, status = lung$status,
      arm = lung$sex - 1, tau = 600)
```

11.26.6 Output & Results

`rmst2()` returns the RMST for each arm, the difference and ratio between arms, and 95 % confidence intervals derived from the asymptotic theory. The output is structured for direct inclusion in clinical-trial reports.

11.26.7 Interpretation

A reporting sentence: "Through 600 days of follow-up, female patients accumulated an average of 380 days alive vs 310 days for male patients; the RMST difference was 70 days (95 % CI 20 to 120, $p = 0.005$). Because the hazards were not proportional (Schoenfeld test $p = 0.04$), RMST is the preferred summary." Always state τ and justify it on clinical grounds; the same data can yield different RMST conclusions at different horizons.

11.26.8 Practical Tips

- Choose τ based on clinical relevance *and* on data adequacy: τ should be smaller than the longest follow-up time across both arms so the KM estimate is stable.
- RMST handles non-proportional hazards, treatment-effect waning, and crossing curves where Cox HR fails; for any trial with these features, RMST is preferable.
- Report RMST on the absolute time scale (days, months, years); clinicians and patients reason about absolute survival, not log hazard ratios.
- For multi-arm trials, run pairwise RMST comparisons with Bonferroni or Hochberg correction; `survRM2::rmst2()` handles two-arm comparisons natively.
- Pair RMST with KM curves and a hazard-ratio sensitivity analysis; reviewers expect to see all three when proportional-hazards assumptions are in question.
- For very small samples or sparse late-time data, switch to bootstrap confidence intervals via `bootRMST` or hand-coded `boot::boot()` resampling on the patient-level data.

11.26.9 R Packages Used

`survRM2` for `rmst2()` and the canonical RMST workflow; `survival` for the underlying Kaplan-Meier estimation; `pseudo` for pseudo-observation-based RMST regression with covariates; `flexsurv` for parametric models that integrate analytically over $S(t)$; `survminer::ggsurvplot()` for KM curves that complement the RMST report.

11.26.10 For Reviewers

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

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

11.26.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.27 Schoenfeld Residuals

11.27.1 Introduction

Schoenfeld residuals are the standard diagnostic for the proportional-hazards (PH) assumption in Cox regression. They are computed at each event time as the difference between the failing subject's covariate value and the expected covariate value across the risk set. Under PH, Schoenfeld residuals are uncorrelated with time; a trend across time signals that the hazard ratio is itself time-varying, which violates the central assumption of the Cox model and invalidates a single-number hazard-ratio interpretation.

11.27.2 Prerequisites

A working understanding of Cox proportional-hazards regression, the partial likelihood, and the difference between the proportional-hazards and accelerated-failure-time framings of survival regression.

11.27.3 Theory

At event time t_i in a Cox model, the Schoenfeld residual for covariate x is

$$r_i = x_i - \hat{\mathbb{E}}[x \mid \text{risk set at } t_i],$$

with the expectation weighted by the partial-likelihood weights $\hat{w}_j = \exp(X_j\hat{\beta}) / \sum_k \exp(X_k\hat{\beta})$ over the risk set. Grambsch and Therneau (1994) developed scaled versions whose correlation with a function of time $g(t)$ — typically $\log t$ or rank-time — yields a χ^2_1 test of PH for each covariate, plus a global χ^2_p test for the joint null. The function `cox.zph()` implements this test and returns smoothed-residual plots that visualise the time-varying effect when one is present.

11.27.4 Assumptions

A Cox model has been fitted, with sufficient events to estimate the smoothed Schoenfeld residual function.

11.27.5 R Implementation

```r
library(survival)

data(lung)
fit ← coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)

zph ← cox.zph(fit)
print(zph)

# Visualise
par(mfrow = c(1, 3))
plot(zph)
par(mfrow = c(1, 1))
```

11.27.6 Output & Results

`print(zph)` returns per-covariate χ^2 statistics, degrees of freedom, and p-values, plus a global test against the joint null that all hazard ratios are constant. `plot(zph)` produces one diagnostic panel per covariate showing the smoothed Schoenfeld residual against transformed time, with a horizontal reference line at the constant-hazard-ratio assumption.

11.27.7 Interpretation

A reporting sentence: "Schoenfeld residual plots for age, sex, and ECOG performance status showed no systematic trend (per-covariate p all > 0.20, global $p = 0.55$); the proportional-hazards assumption is supported." When PH fails: "The hazard ratio for treatment varied with time (Schoenfeld $p < 0.001$, decreasing trend); we report restricted-mean survival difference rather than a single Cox HR."

11.27.8 Practical Tips

- Always run `cox.zph()` after fitting a Cox model; reporting an HR without a PH check is no longer acceptable in clinical-trial papers.
- When PH fails for one covariate, options include: stratifying by it (`+ strata(var)`), modelling its effect as time-varying with `tt()` or `survSplit()`, or switching to RMST as the summary of treatment effect.
- Categorical covariates with k levels produce $k - 1$ Schoenfeld tests; PH may hold for some contrasts and fail for others.
- Small samples have low power to detect PH violations; pair the formal test with a visual inspection of `plot(zph)` rather than relying on p-values alone.
- For strictly time-varying effects, switch to flexible parametric models (`flexsurv::flexsurvreg` with splines on log time) or include `tt(...)` time-transform functions in `coxph()`.
- A failed PH test in itself is not reason to abandon the Cox model; it is a reason to refine the model, change the summary, or stratify.

11.27.9 R Packages Used

`survival` for `coxph()` and `cox.zph()`; `survminer::ggcoxzph()` for ggplot-style Schoenfeld diagnostic panels; `timereg` for additive hazards and structured time-varying-coefficient models; `flexsurv` for parametric alternatives when PH fails.

11.27.10 For Reviewers

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

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

11.27.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.28 Stratified Cox Models

11.28.1 Introduction

A stratified Cox model is the standard fix when the proportional-hazards assumption fails across the levels of a categorical covariate but holds within them. Stratification estimates a separate baseline hazard $h_{0s}(t)$ for each stratum while constraining the covariate effects β to be common across strata. The cost is that you cannot estimate the effect of the stratifier itself — its baselines are absorbed — but the gain is honest hazard-ratio estimation for the rest of the covariates without forcing PH across the problematic dimension.

11.28.2 Prerequisites

A working understanding of Cox proportional-hazards regression, the diagnostic role of Schoenfeld residuals, and the concept of stratification as an alternative to direct adjustment.

11.28.3 Theory

The stratified Cox model is

$$h(t \mid x, \text{stratum} = s) = h_{0s}(t) \exp(x^\top \beta).$$

The partial likelihood is summed across strata: each stratum contributes the standard Cox partial-likelihood term over its own risk sets, and the resulting score equations determine $\hat{\beta}$. Because the stratum baselines are unrestricted, no assumption is made about how $h_{0s}(t)$ relates across s. The trade-off is that the effect of being in stratum s versus another stratum cannot be estimated within the model — that contrast is precisely the term absorbed into the separate baselines.

11.28.4 Assumptions

Proportional hazards within each stratum (so the common β is meaningful), independent observations, and non-informative censoring.

11.28.5 R Implementation

```
library(survival)

data(lung)
# If ph.ecog violates PH, stratify on it
fit ← coxph(Surv(time, status) ~ age + sex + strata(ph.ecog), data = lung)
summary(fit)

cox.zph(fit)
```

11.28.6 Output & Results

`summary(fit)` lists hazard ratios for non-stratified covariates with confidence intervals and Wald tests; the stratifier appears nowhere because its effect is absorbed into the per-stratum baselines. `cox.zph()` checks PH for the remaining covariates within strata.

11.28.7 Interpretation

A reporting sentence: “After Schoenfeld residuals indicated a PH violation for ECOG performance status (per-covariate $p < 0.01$), we fit a Cox model stratified on ECOG; the resulting hazard ratios for age (HR 1.02 per year, 95 % CI 1.01 to 1.03) and sex (HR 0.60 for female vs male, 95 % CI 0.42 to 0.85) are pooled across strata.” When the stratifier is itself of clinical interest, supplement the model with stratum-specific Kaplan-Meier curves so readers can see the absorbed effect.

11.28.8 Practical Tips

- Stratify on a covariate when its PH fails and you do not need its hazard-ratio estimate; if you do need it, switch to a time-varying coefficient via `tt()` or `survSplit()` instead.
- Many small strata reduce the effective sample size for $\hat{\beta}$; pool levels (e.g. ECOG 0 vs 1+) when raw stratification leaves few events per stratum.
- Interactions between stratum and other covariates allow different β per stratum, generalising the model toward fully separate fits per stratum at the cost of degrees of freedom.

- Predictions and survival curves are stratum-specific; report which baseline is used for any predicted survival probability.
- Always verify that PH still holds for the unstratified covariates after stratification; failure there motivates additional stratification or a switch to time-varying effects.
- For continuous covariates that violate PH, stratification requires categorisation (typically into quartiles), which loses information; time-varying coefficients are usually preferable.

11.28.9 R Packages Used

`survival` for `coxph()` with `strata()`; `survminer::ggcoxzph()` for PH diagnostics within strata; `flexsurv` for parametric alternatives that can accommodate non-proportional hazards directly; `timereg` for additive hazards models when stratification is too coarse.

11.28.10 For Reviewers

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

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

11.28.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.29 Simulating Survival Data

11.29.1 Introduction

Simulating censored time-to-event data is essential for three classes of work: power and sample-size calculations for trials with non-standard designs, validation of new statistical methods (does the estimator recover the truth?), and pedagogy. The inverse-hazard method generates event times from any specified hazard function and lets you build realistic datasets that match a target survival pattern, censoring distribution, and covariate-effect structure.

11.29.2 Prerequisites

A working understanding of hazard and cumulative hazard functions, the inverse-CDF method for random-number generation, and the relationship between proportional-hazards models and exponentiated covariate effects.

11.29.3 Theory

For a hazard $h(t \mid x)$ with cumulative hazard $H(t \mid x)$ and a uniform variate $U \sim \mathrm{Uniform}(0, 1)$,

$$T = H^{-1}\big(-\log U \mid x\big)$$

has the target distribution. Under proportional hazards $H(t \mid x) = H_0(t)\exp(x^\top \beta)$, so

$$T_i = H_0^{-1}\big(-\log U_i / \exp(x_i^\top \beta)\big)\,.$$

Censoring is added by drawing an independent censoring time C_i from a chosen distribution (administrative, exponential, or empirical from a real cohort) and recording $\min(T_i, C_i)$ as the observation time and $\mathbf{1}\{T_i \le C_i\}$ as the event indicator.

11.29.4 Assumptions

The specified baseline hazard and covariate-effect structure are the truth being simulated; the censoring distribution is independent of T_i unless the simulation explicitly handles informative censoring.

11.29.5 R Implementation

```r
library(simsurv)

set.seed(2026)
n <- 300
x <- data.frame(id = 1:n, arm = factor(rep(c("A", "B"), each = n/2)))
x$arm_num <- as.numeric(x$arm) - 1

# Weibull baseline: lambda = 0.1, gamma = 1.2
sim <- simsurv(lambdas = 0.1, gammas = 1.2,
               x = x, betas = c(arm_num = -0.5),
               maxt = 10)

# Merge with covariates
d <- merge(sim, x, by = "id")
head(d)

# Kaplan-Meier
library(survival)
fit_km <- survfit(Surv(eventtime, status) ~ arm, data = d)
plot(fit_km)
```

11.29.6 Output & Results

`simsurv()` returns a tibble with one row per subject containing the simulated event time, the censoring indicator (here taken at `maxt = 10`), and the subject id. Merging back with the covariate frame gives a complete simulated dataset ready for analysis. The Kaplan-Meier plot shows the two arms separating with the designed effect size.

11.29.7 Interpretation

A reporting sentence: "Simulated Weibull data with shape 1.2, baseline scale 0.1, and a treatment hazard ratio of $\exp(-0.5) = 0.61$; a Cox model on the simulated data recovered an HR of 0.59 (95 % CI 0.46 to 0.76), confirming the simulation is correctly calibrated." Always confirm that the analysis pipeline recovers the simulation parameters before using the simulator for power calculations.

11.29.8 Practical Tips

- `simsurv` supports Weibull and Gompertz baselines plus user-defined hazard functions; for arbitrary parametric shapes, supply `hazard` or `loghazard` directly.
- For competing risks, simulate each cause's event time independently and record the cause that occurred first; the resulting data have the correct cumulative incidence functions.
- Match the censoring distribution to the real-study target: administrative censoring at study end, plus loss-to-follow-up exponentials with rates calibrated to historical drop-out.
- For power calculations, run thousands of simulated datasets, fit the planned model on each, and record the proportion of times the test rejects at the chosen alpha; this is the standard Monte Carlo power estimator.
- For time-varying covariates, simulate via piecewise-constant hazards across pre-specified intervals; `simsurv` supports `tdefunction` for time-dependent covariate effects.
- For validation of new methods, re-simulate under several scenarios (PH violation, heavy censoring, sparse events) and check robustness; a method that works only on its design assumptions is brittle.

11.29.9 R Packages Used

`simsurv` for the canonical inverse-hazard simulator with Weibull, Gompertz, and user-defined baselines; `flexsurv` for `rflexsurv()` simulation from fitted flexible models; `coxed` for Cox-model-based simulations; `survSim` for additional simulation extensions including frailty and time-dependent effects.

11.29.10 For Reviewers

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

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

11.29.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias

- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.30 Time-Dependent ROC

11.30.1 Introduction

Time-dependent ROC extends classical ROC analysis to right-censored survival data. At each chosen evaluation time t, the analysis treats subjects observed to fail before t as cases and subjects known to survive past t as controls; subjects censored before t are reweighted via inverse-probability-of-censoring weights. The resulting time-specific AUC summarises how well a continuous risk score (such as a Cox linear predictor) discriminates failures from survivors at t. It is the natural extension of the C-index to "what is the AUC at exactly one year" reporting.

11.30.2 Prerequisites

A working understanding of ROC curves, the C-index, Cox regression linear predictors, and inverse-probability-of-censoring weighting.

11.30.3 Theory

Two definitions of cases and controls compete:

- **Incident / dynamic** (I/D): cases are subjects who fail at time t exactly; controls are those still at risk past t. Useful for hazard-style discrimination.
- **Cumulative / dynamic** (C/D): cases are subjects who fail at any time $\leq t$; controls are those still at risk past t. The more common choice for clinical risk-score reporting.

Inverse-probability-of-censoring weights compensate for subjects censored before the comparison can be made; the censoring distribution is typically estimated by Kaplan-Meier on the censoring times. The result is a time-dependent AUC at each requested t with bootstrap or analytical confidence intervals.

11.30.4 Assumptions

Non-informative censoring (or correctly conditional on covariates if `weighting = "cox"` or `"aalen"` is used) and a continuous risk marker that is meaningful as a ranking score.

11.30.5 R Implementation

```r
library(timeROC); library(survival)

data(lung)
fit ← coxph(Surv(time, status) ~ age + sex + ph.ecog, data = lung)
lp ← predict(fit, type = "lp")

tROC ← timeROC(T = lung$time, delta = lung$status, marker = lp,
               times = c(180, 365, 730), cause = 1,
               weighting = "marginal")
tROC
plot(tROC, time = 365, col = "steelblue")
```

11.30.6 Output & Results

`timeROC` returns time-specific AUCs with their standard errors at each requested horizon, along with empirical sensitivity-specificity pairs that can be plotted as a ROC curve. The plot shows the ROC curve at one selected time; running over multiple times reveals how discrimination evolves through follow-up.

11.30.7 Interpretation

A reporting sentence: “The Cox linear predictor’s time-dependent AUC was 0.72 (95 % CI 0.65 to 0.79) at one year and 0.69 (95 % CI 0.62 to 0.76) at two years; discrimination remained stable across the two-year window relevant to clinical decision-making.” Always report AUCs at clinically meaningful horizons, not just the global C-index, because discrimination can change substantially across follow-up.

11.30.8 Practical Tips

- Report AUCs at the horizons that drive clinical decisions; a five-year AUC is irrelevant for a six-month treatment decision and vice versa.
- The cumulative/dynamic definition is more common in clinical reporting; incident/dynamic is more useful when modelling hazards directly.
- `timeROC()` supports competing risks via the `cause` argument; the same workflow applies with the failure of interest specified.
- AUC alone does not capture calibration; pair time-dependent AUCs with calibration plots and Brier scores at the same times.
- For external validation, apply the development-cohort risk score to the validation cohort and compute time-dependent AUCs there; in-sample AUCs are optimistic.
- Use `weighting = "cox"` or `weighting = "aalen"` when censoring depends on covariates; the marginal weighting assumes censoring is uninformative given baseline covariates.

11.30.9 R Packages Used

`timeROC` for `timeROC()` with marginal, Cox, and Aalen-based weighting; `risksetROC` for an alternative kernel-smoothed implementation; `pec` for compatible prediction-error curves; `survAUC` for additional AUC variants including Uno’s C-statistic.

11.30.10 For Reviewers

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

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

11.30.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.31 Time-Varying Covariates

11.31.1 Introduction

Real survival studies routinely include covariates whose value changes during follow-up: treatment initiation, laboratory measurements repeated at each visit, secondary events that alter prognosis. Including these as time-fixed baseline values misrepresents the exposure history; pretending the baseline value applied throughout follow-up biases hazard-ratio estimates and is a leading cause of immortal-time errors. The standard remedy is the counting-process data layout, which splits each subject's follow-up into intervals during which covariate values are constant and lets the partial likelihood combine the contributions correctly.

11.31.2 Prerequisites

A working understanding of Cox regression, the partial likelihood, and the distinction between baseline (time-fixed) and time-varying covariates.

11.31.3 Theory

In counting-process format each row represents one interval $(\tau_{ij}^{\text{start}}, \tau_{ij}^{\text{stop}}]$ during which covariate values for subject i are constant. The columns are:

- `tstart`: interval start.
- `tstop`: interval end.
- `event`: 1 if the event occurred at `tstop`, otherwise 0.
- Covariates with their values during the interval.

The partial likelihood evaluates each subject's risk-set contribution using whichever interval includes the event time, so each event time naturally compares subjects with their then-current covariate values. Robust ("sandwich") variance estimators, requested via `cluster = id`, account for the within-subject correlation induced by multiple rows per subject.

11.31.4 Assumptions

Covariate values are constant within each interval and accurately measured. The covariate is "internal" if its trajectory could be affected by the event of interest (e.g., a biomarker that changes after treatment), or "external" if it is determined exogenously (e.g., scheduled treatment dose); inference from internal covariates requires careful causal interpretation.

11.31.5 R Implementation

```
library(survival)

data(cgd, package = "survival")
head(cgd)

# Multi-row-per-subject: counting-process format
fit ← coxph(Surv(tstart, tstop, status) ~ treat + inherit,
            data = cgd, cluster = id)
summary(fit)
```

11.31.6 Output & Results

`summary(fit)` produces the usual Cox-regression output with hazard ratios, confidence intervals, and Wald tests for each covariate, plus robust standard errors that respect the within-subject clustering. The number of subjects (one per cluster) and the number of events are reported alongside the number of intervals.

11.31.7 Interpretation

A reporting sentence: "After accounting for the time-varying treatment status across each subject's intervals, prophylactic treatment reduced the hazard of CGD infection by 60 % (HR 0.40, 95 % CI 0.25 to 0.63); robust standard errors were clustered on subject identifier to account for repeated intervals." When the time-varying covariate is treatment status, always describe whether subjects could move between exposure groups and how often.

11.31.8 Practical Tips

- Use `survSplit()` (or `tmerge()` for multi-event data) to convert wide-format clinical data into the counting-process layout; doing this by hand is error-prone.
- Always specify `cluster = id` (or `cluster(id)` in the formula) so robust standard errors account for multiple rows per subject; ignoring clustering produces SEs that are too small.
- Check for overlapping intervals: the same subject should never have two rows whose intervals share any time.

- For external time-varying covariates (treatment delivered at fixed planned times), landmark analysis is a cleaner alternative that avoids the data-management overhead.
- For internal time-varying covariates (biomarkers that respond to the disease course), joint longitudinal-survival models (`JM`, `joineRML`) are preferable because they correctly handle the feedback loop.
- Be explicit in the methods section about which covariates were treated as time-varying and how often they were updated; readers cannot reproduce the analysis without this information.

11.31.9 R Packages Used

`survival` for `coxph()` with counting-process input, `survSplit()`, and `tmerge()`; `dynpred` for landmark and dynamic-prediction alternatives; `JM` and `joineRML` for joint longitudinal-survival models; `mstate` for multi-state extensions when intervals correspond to state transitions.

11.31.10 For Reviewers

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

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

11.31.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.32 Weighted Log-Rank Tests

11.32.1 Introduction

The standard log-rank test gives every event time equal weight in the comparison between groups. That is the optimal weighting under proportional hazards but loses power when the difference between groups is concentrated at one end of follow-up — early differences that wane (typical of immunotherapy run-in effects), late differences that emerge after a delay (typical of immune-checkpoint inhibitors), or crossing-curve patterns. Weighted log-rank tests recover power in these settings by emphasising the time region where the difference is expected.

11.32.2 Prerequisites

A working understanding of the log-rank test, the proportional-hazards assumption, and the relationship between hazard ratios and survival curves at different follow-up times.

11.32.3 Theory

The general form of a weighted log-rank test is

$$\chi^2 = \frac{\left(\sum_j w(t_j)[O_j - E_j]\right)^2}{\sum_j w(t_j)^2 V_j},$$

where $O_j - E_j$ is the observed-minus-expected event count at t_j and V_j is its variance. Common weighting schemes:

- **Log-rank**: $w(t) = 1$. Optimal under proportional hazards.
- **Wilcoxon / Gehan-Breslow**: $w(t_j) = n_j$ (size of risk set). Emphasises early differences.
- **Peto-Peto**: $w(t) = \hat{S}(t)$. Similar to Wilcoxon but uses the pooled survival estimator.
- **Fleming-Harrington** $G^{\rho,\gamma}$: $w(t) = \hat{S}(t^-)^\rho\,[1 - \hat{S}(t^-)]^\gamma$. $G^{1,0}$ emphasises early differences; $G^{0,1}$ emphasises late differences; $G^{1,1}$ emphasises mid-follow-up differences.

Combination tests (max-combo) compute several weights simultaneously and apply a multiple-testing correction; this preserves Type I control while gaining power across a range of alternative shapes.

11.32.4 Assumptions

Independent non-informative censoring. The chosen weight reflects the hazard pattern actually expected; post-hoc weight selection inflates Type I error and is statistical malpractice.

11.32.5 R Implementation

```r
library(survival)

data(lung)
# Standard log-rank (rho = 0)
survdiff(Surv(time, status) ~ sex, data = lung, rho = 0)

# Peto-Peto (rho = 1): weights early differences
survdiff(Surv(time, status) ~ sex, data = lung, rho = 1)

# Fleming-Harrington via nphRCT or similar package (conceptually)
```

11.32.6 Output & Results

`survdiff()` reports the chi-squared statistic, degrees of freedom, p-value, and observed-vs-expected counts under the chosen weight (`rho` argument). The `nph` and `nphRCT` packages add Fleming-Harrington $G^{\rho,\gamma}$ tests and combination tests with formal Type I control.

11.32.7 Interpretation

A reporting sentence: "The pre-specified Peto-Peto weighted log-rank test rejected the null of equal survival ($\chi^2_1 = 12.8$, $p < 0.001$); the standard log-rank produced a similar conclusion ($\chi^2_1 = 10.3$, $p = 0.001$). The Peto-Peto weighting was chosen a priori based on prior evidence that the treatment effect is concentrated in the first 18 months." Always state the weight in advance and justify it; weight-shopping is the most common abuse of these tests.

11.32.8 Practical Tips

- Pre-specify the weighting scheme in the protocol; switching weights after seeing the data inflates Type I error far above the nominal level.
- For trials with expected delayed treatment effects (immunotherapy), Fleming-Harrington $G^{0,1}$ or a max-combo test is increasingly the standard.
- Restricted-mean survival time (RMST) is a complementary approach for non-proportional hazards; report both when proportional hazards is in question.
- Combination tests (max-combo: standard log-rank, $G^{1,0}$, $G^{0,1}$, $G^{1,1}$) preserve Type I control via Hochberg-style adjustment; `nph::nphRCT` implements them.
- Inspect the Kaplan-Meier curve before choosing a weight; a clearly delayed effect rules out the standard log-rank as the primary test.
- Modern immuno-oncology trials routinely pre-specify a max-combo or RMST primary analysis; the era of one-size-fits-all log-rank reporting is closing.

11.32.9 R Packages Used

`survival` for `survdiff()` with `rho` weighting; `nph` and `nphRCT` for Fleming-Harrington $G^{\rho,\gamma}$ tests and max-combo statistics; `survRM2` for RMST as an alternative summary; `coin` for permutation-based exact weighted log-rank tests.

11.32.10 For Reviewers

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

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

11.32.11 See also — labs in this chapter

- Survival primer: KM, log-rank, Cox PH
- Time-varying covariates and immortal-time bias
- Competing risks and multistate models
- Survival ML (random survival forest, DeepSurv)
- Time-dependent Brier, IPA, external validation

11.33 Week 3, Session 2 — Competing risks and multistate models

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.33.1 Learning objectives

- Estimate cause-specific cumulative incidence with `tidycmprsk`.
- Contrast cause-specific hazards with the Fine-Gray subdistribution hazard.
- Set up a three-state illness-death multistate model with `mstate`.

11.33.2 Prerequisites

Survival analysis.

11.33.3 Background

Competing risks are events that preclude the event of interest: death from cancer competes with death from other causes. Kaplan- Meier treats the competing event as censoring, which is wrong when the censoring is informative. Cumulative incidence functions (CIFs) give the probability that the event of interest has occurred by time *t in the presence of* competing risks.

Two hazard models are common. Cause-specific Cox models treat each cause in turn, censoring at other causes; they answer an aetiologic question (“does the covariate affect the rate of this cause?”). The Fine-Gray subdistribution hazard model fixes subjects with competing events in the risk set indefinitely; it answers a prognostic question (“does the covariate affect the cumulative incidence?”).

Multistate models generalise survival to several possible transitions. The illness-death model has three states: healthy, ill, dead, with transitions healthy → ill, healthy → dead, and ill → dead. `mstate` builds these objects from tabular data.

Fine-Gray is descriptive; cause-specific Cox is causal (in the simple, aetiological sense). A covariate can increase the Fine-Gray subdistribution hazard of cause 1 either by increasing cause 1 itself or by decreasing cause 2 — it is a net effect on the cumulative incidence.

11.33.4 Setup

```
library(tidyverse)
library(survival)
library(tidycmprsk)
library(ggsurvfit)
library(mstate)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.33.5 1. Hypothesis

In the `survival::colon` data (recurrence and death as competing events), a treatment covariate affects both events. We estimate the cumulative incidence of each.

11.33.6 2. Visualise

```
data(colon, package = "survival")
# Keep death records (etype == 2) + recurrence merged; for teaching
# we build a two-event dataset:
dat ← colon ▷
  filter(etype == 1) ▷                        # recurrence row per subject
  transmute(id, time, status_rec = status,
            rx, age, sex) ▷
  left_join(
    colon ▷ filter(etype == 2) ▷
      transmute(id, time_d = time, status_d = status),
    by = "id"
  ) ▷
  mutate(
    # event type: 0 censored, 1 recurrence, 2 death without recurrence
    etype = case_when(
      status_rec == 1 ~ 1L,
      status_d   == 1 ~ 2L,
      TRUE            ~ 0L
    ),
    etime = pmin(time, time_d, na.rm = TRUE),
    etype_f = factor(etype, levels = 0:2,
                     labels = c("censored", "recurrence", "death"))
  ) ▷
  drop_na(etime, etype_f, rx)

ggplot(dat, aes(etime, fill = etype_f)) +
  geom_histogram(bins = 40, alpha = 0.7, position = "identity") +
  labs(x = "Time (days)", y = "Count", fill = NULL)
```

11.33.7 3. Assumptions

Non-informative censoring within cause; proportional hazards for Fine-Gray; for multistate, Markov transitions (future depends only on current state).

11.33.8 4. Conduct

```
cif ← cuminc(Surv(etime, etype_f) ~ rx, data = dat)
cif
```

```r
ggsurvfit::ggcuminc(cif, outcome = c("recurrence", "death")) +
  labs(x = "Days", y = "Cumulative incidence")
```

```r
# Fine-Gray subdistribution hazard for recurrence
fg ← crr(Surv(etime, etype_f) ~ rx + age + sex, data = dat,
         failcode = "recurrence")
fg
```

```r
# Multistate: illness-death using mstate on a simple simulated dataset
n ← 300
sim ← tibble(
  id = seq_len(n),
  t_ill   = rexp(n, rate = 0.02),
  t_death = rexp(n, rate = 0.01)
) ▷
  mutate(
    t1 = pmin(t_ill, t_death, 50),
    s1 = if_else(t_ill < t_death & t_ill ⩽ 50, 1L, 0L),
    s2 = if_else(t_death ⩽ 50 & t_death < t_ill, 1L, 0L)
  )

tmat ← transMat(x = list(c(2, 3), c(3), c()),
                names = c("healthy", "ill", "dead"))
tmat
```

11.33.9 5. Concluding statement

> Cause-specific cumulative incidence functions differed by treatment arm in the colon data, with the Fine-Gray subdistribution hazard estimated above. The three-state illness- death transition matrix (healthy → ill → dead) shows how `mstate` represents the problem; the full fit is straightforward once the long-format dataset is built via `msprep`.

11.33.10 Common pitfalls

- Reporting 1 − KM as the cumulative incidence in the presence of competing risks.
- Using Fine-Gray when the question is aetiological.
- Reporting both cause-specific and Fine-Gray hazard ratios without distinguishing them.
- Assuming Markov transitions in a multistate model when history matters.

11.33.11 Further reading

- Fine JP, Gray RJ (1999), *A proportional hazards model for the subdistribution of a competing risk*.
- Putter H, Fiocco M, Geskus RB (2007), *Tutorial in biostatistics: competing risks and multi-state models*.
- Austin PC, Fine JP (2017), *Practical recommendations for reporting Fine-Gray model analyses*.

11.33.12 Session info

```
sessionInfo()
```

11.33.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.34 Week 3, Session 4 — Survival ML (random survival forest, DeepSurv)

Workflow labs use the variant template: Goal → Approach → Execution → Check → Report.

11.34.1 Learning objectives

- Fit a random survival forest and interpret its variable importance and risk stratification.
- Describe the DeepSurv architecture conceptually and when it is preferred over the forest.
- Compare against a Cox model on the same features.

11.34.2 Prerequisites

Cox regression and censoring.

11.34.3 Background

Survival analysis deals with time-to-event data in the presence of censoring. The Cox proportional-hazards model is the workhorse linear method; random survival forests (RSF) extend the idea of random forests to right-censored outcomes, using a log-rank split rule and Harrell's concordance as the default fit criterion. They handle nonlinearities and interactions automatically and are a good first nonparametric benchmark.

DeepSurv is a neural-network generalisation of the Cox partial likelihood: the linear predictor is replaced by a feed-forward network. In small biomedical datasets it rarely beats a well-tuned RSF or a Cox model with splines, but the approach scales to large cohorts and to inputs where a representation must be learned.

Evaluation of survival models uses time-integrated concordance (Harrell's C), the integrated Brier score, and — increasingly — decision curves at a clinically relevant horizon. Reporting only C on a single test set is weak evidence.

11.34.4 Setup

```
library(tidyverse)
library(survival)
library(randomForestSRC)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.34.5 1. Goal

Fit an RSF on `survival::pbc` and compare to a Cox model using Harrell's concordance.

11.34.6 2. Approach

```
d ← survival::pbc ▷
  as_tibble() ▷
  drop_na(bili, albumin, age, edema, protime, stage) ▷
  mutate(status = ifelse(status == 2, 1, 0))

ggplot(d, aes(time / 365.25, fill = factor(status))) +
  geom_histogram(bins = 30, alpha = 0.7, position = "identity") +
  labs(x = "follow-up (years)", y = "count", fill = "event")
```

11.34.7 3. Execution

```
fit_cox ← coxph(Surv(time, status) ~ bili + albumin + age + edema +
                  protime + stage, data = d)
fit_rsf ← rfsrc(Surv(time, status) ~ bili + albumin + age + edema +
                  protime + stage,
                data = as.data.frame(d), ntree = 500,
                importance = TRUE)
fit_rsf
```

```
tibble(var = names(fit_rsf$importance),
       imp = as.numeric(fit_rsf$importance)) ▷
  arrange(desc(imp)) ▷
  ggplot(aes(reorder(var, imp), imp)) +
  geom_col() + coord_flip() + labs(x = NULL, y = "VIMP")
```

11.34.8 4. Check

Compare concordance on held-out data.

```
idx ← sample(nrow(d), 0.7 * nrow(d))
tr ← d[idx, ]; te ← d[-idx, ]
cox2 ← coxph(Surv(time, status) ~ bili + albumin + age + edema +
               protime + stage, data = tr)
rsf2 ← rfsrc(Surv(time, status) ~ bili + albumin + age + edema +
               protime + stage,
             data = as.data.frame(tr), ntree = 500)
c_cox ← survival::concordance(cox2, newdata = te)$concordance
p_rsf ← predict(rsf2, newdata = as.data.frame(te))$predicted
c_rsf ← survival::concordance(Surv(te$time, te$status) ~ p_rsf,
                              reverse = TRUE)$concordance
c(cox = c_cox, rsf = c_rsf)
```

DeepSurv conceptual sketch.

```
library(torch)
# A DeepSurv-style loss computes the partial log-likelihood on the
# sorted risk set. The network output h(x) replaces the Cox linear
# predictor; the loss is  -(sum_{i:event}  h(x_i) - log sum_{j in R_i} exp(h(x_j))).
deepsurv_loss <- function(risk, time, event) {
  ord <- order(time, decreasing = TRUE)
  risk <- risk[ord]; event <- event[ord]
  log_cumsum <- torch_logcumsumexp(risk, dim = 1)
  -torch_sum(event * (risk - log_cumsum)) / torch_sum(event)
}
```

11.34.9 5. Report

> On `survival::pbc`, a random survival forest achieved test concordance `round(c_rsf, 3)` compared with `round(c_cox, 3)` for a multivariable Cox model. Bilirubin, albumin, and age dominated VIMP. A DeepSurv network with the partial-likelihood loss would be the natural next step on a larger cohort.

On this sample size, the RSF and Cox model are close; on larger datasets with nonlinearities, the forest often pulls ahead.

11.34.10 Common pitfalls

- Treating the Cox proportional-hazards assumption as always met; it often is not, and the RSF is more robust to violation.
- Comparing models without the same censoring distribution in the test set.
- Reporting only C without a calibration or decision curve at the horizon of interest.

11.34.11 Further reading

- Ishwaran H et al. (2008), *Random survival forests*.
- Katzman JL et al. (2018), *DeepSurv: Personalized treatment recommender system using a Cox proportional hazards deep neural network*.

11.34.12 Session info

```
sessionInfo()
```

11.34.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.35 Week 4, Session 3 — Survival primer: KM, log-rank,

Cox PH

Inference lab: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.35.1 Learning objectives

- Estimate and plot a Kaplan-Meier survival curve for a two-group comparison.
- Run and interpret a log-rank test.
- Fit a Cox proportional-hazards model, check the PH assumption, and read a hazard ratio honestly.

11.35.2 Prerequisites

Comfort with GLMs and with reading a regression table.

11.35.3 Background

Survival analysis handles outcomes that are times to an event, where some subjects have not yet had the event when observation ends — the familiar right-censoring problem. Ignoring censoring by throwing out the censored observations biases the estimator; pretending the censored observations are event-free biases it the other way. The survival machinery handles censoring correctly by reasoning at each event time about who is still at risk.

Three tools cover most clinical applications: the Kaplan-Meier estimator for a non-parametric survival curve, the log-rank test for a two-group comparison, and Cox proportional-hazards regression for adjustment and multivariable modelling. The PH assumption — that the hazard ratio is constant over time — deserves explicit checking; when it fails, a time-varying coefficient or a stratified model is usually the right fix.

11.35.4 Setup

```
library(tidyverse)
library(survival)
library(ggsurvfit)
library(broom)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.35.5 1. Hypothesis

Using `survival::lung`, compare survival between male and female patients. H_0: hazard ratio = 1. H_1: hazard ratio ≠ 1. $\alpha = 0.05$.

11.35.6 2. Visualise

```
lung2 ← lung ▷
  mutate(sex = factor(sex, levels = 1:2, labels = c("male", "female")))

km ← survfit2(Surv(time, status) ~ sex, data = lung2)

km ▷
  ggsurvfit() +
  add_confidence_interval() +
  add_risktable() +
  labs(x = "Days", y = "Survival probability")
```

11.35.7 3. Assumptions

Two assumptions are worth checking: non-informative censoring (patients censored at a given time are representative of those still at risk) and proportional hazards.

```
cox_fit ← coxph(Surv(time, status) ~ sex + age, data = lung2)
cox.zph(cox_fit)
```

A large *p*-value on each row and a non-systematic Schoenfeld residual plot support the PH assumption; a small *p* calls for a time-varying coefficient or stratification.

11.35.8 4. Conduct

```
lr ← survdiff(Surv(time, status) ~ sex, data = lung2)
lr

tidy(cox_fit, exponentiate = TRUE, conf.int = TRUE)
```

11.35.9 5. Concluding statement

```
hr_row ← tidy(cox_fit, exponentiate = TRUE, conf.int = TRUE) ▷
  filter(term == "sexfemale")
```

> Female sex was associated with a lower risk of death relative to male sex (HR `round(hr_row$estimate, 2)`; 95% CI `round(hr_row$conf.low, 2)` to `round(hr_row$conf.high, 2)`; *p* = `signif(hr_row$p.value, 2)`), adjusted for age. The proportional-hazards assumption was not clearly violated (Schoenfeld test, all *p* > 0.05).

11.35.10 Common pitfalls

- Reporting a median survival that is not reached and calling it “undefined” in the results. Say so explicitly.
- Ignoring the PH check because the output looks clean.
- Comparing Kaplan-Meier curves by eye alone — add the log-rank test and the risk table.

- Collapsing a competing event (e.g. death from another cause) into censoring without justification.

11.35.11 Further reading

- Harrell FE. *Regression Modeling Strategies*, ch. 20.
- Therneau TM & Grambsch PM. *Modeling Survival Data.*

11.35.12 Session info

```
sessionInfo()
```

11.35.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.36 Week 3, Session 5 — Time-dependent Brier, IPA, external validation

Testing labs use the main template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.36.1 Learning objectives

- Compute a time-dependent Brier score and the Index of Prediction Accuracy (IPA).
- Simulate an external-validation cohort with distribution shift and evaluate a fitted model on it.
- Describe why a model can have a high internal AUC and a bad external IPA.

11.36.2 Prerequisites

Survival ML; calibration.

11.36.3 Background

Validation is the point at which a prediction model either earns its keep or does not. Internal validation — CV or the bootstrap on the training data — controls optimism from the fitting step. External validation applies the fixed model to an independent cohort and measures discrimination, calibration, and clinical utility. Time- dependent Brier scores and IPA (1 − Brier/Brier_null) summarise the cost of a probabilistic prediction at a given horizon, combining discrimination and calibration into a single number per time.

Real biomedical models routinely lose performance on external cohorts. Distribution shift (different age, comorbidities, measurement protocols) is the usual reason; miscalibration is the usual failure mode. This lab simulates both and shows the diagnostic signature each leaves behind.

TRIPOD-AI asks you to report all three — discrimination, calibration, clinical utility — on both internal and external data, and to disclose the shift between them. A reader who only sees AUC on internal data has no basis for deploying the model.

11.36.4 Setup

```
library(tidyverse)
library(survival)
library(pROC)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.36.5 1. Hypothesis

A Cox model fit on a development cohort retains its calibration and discrimination on an external cohort with shifted covariate distribution.

11.36.6 2. Visualise

```
make_cohort ← function(n, mu_x = 0, beta = 0.6) {
  x ← rnorm(n, mu_x)
  lp ← beta * x
  t ← rexp(n, rate = exp(lp - 1))
  c ← rexp(n, rate = 0.1)
  time ← pmin(t, c)
  event ← as.integer(t ⩽ c)
  tibble(x = x, time = time, event = event)
}
dev  ← make_cohort(500, mu_x = 0)
ext  ← make_cohort(500, mu_x = 1.0)   # distribution shift
bind_rows(dev ▷ mutate(cohort = "dev"),
          ext ▷ mutate(cohort = "ext")) ▷
  ggplot(aes(x, fill = cohort)) +
  geom_histogram(bins = 30, alpha = 0.7, position = "identity")
```

11.36.7 3. Assumptions

Independent right censoring; Cox proportional hazards in the development cohort; no time-varying covariates.

11.36.8 4. Conduct

Fit on dev; predict survival at a fixed horizon on both cohorts.

```r
fit <- coxph(Surv(time, event) ~ x, data = dev)
horizon <- 3

predict_surv <- function(fit, newdata, t) {
  bh <- basehaz(fit, centered = FALSE)
  H0 <- approx(bh$time, bh$hazard, xout = t, rule = 2)$y
  lp <- predict(fit, newdata = newdata, type = "lp")
  exp(-H0 * exp(lp))
}

s_dev <- predict_surv(fit, dev, horizon)
s_ext <- predict_surv(fit, ext, horizon)

# Observed indicator (for simple Brier w/out censoring at horizon)
obs_dev <- dev$event == 1 & dev$time <= horizon
obs_ext <- ext$event == 1 & ext$time <= horizon

brier <- function(p_event, obs) mean((p_event - obs)^2)
brier_null <- function(obs) mean((mean(obs) - obs)^2)
b_dev <- brier(1 - s_dev, obs_dev); b0_dev <- brier_null(obs_dev)
b_ext <- brier(1 - s_ext, obs_ext); b0_ext <- brier_null(obs_ext)
ipa_dev <- 1 - b_dev / b0_dev
ipa_ext <- 1 - b_ext / b0_ext

auc_dev <- as.numeric(auc(roc(obs_dev, 1 - s_dev, quiet = TRUE)))
auc_ext <- as.numeric(auc(roc(obs_ext, 1 - s_ext, quiet = TRUE)))

tibble(cohort = c("dev", "ext"),
       auc = c(auc_dev, auc_ext),
       brier = c(b_dev, b_ext),
       ipa  = c(ipa_dev, ipa_ext))
```

A calibration plot at the horizon.

```r
mk_cal <- function(p, y, lab) {
  tibble(p = p, y = y, cohort = lab) |>
    mutate(bin = cut(p, quantile(p, seq(0, 1, by = 0.1)),
                     include.lowest = TRUE)) |>
    group_by(cohort, bin) |>
    summarise(mean_pred = mean(p), obs = mean(y), .groups = "drop")
}
bind_rows(mk_cal(1 - s_dev, obs_dev, "dev"),
          mk_cal(1 - s_ext, obs_ext, "ext")) |>
  ggplot(aes(mean_pred, obs, colour = cohort)) +
  geom_point() + geom_line() +
  geom_abline(slope = 1, intercept = 0, colour = "grey50") +
  labs(x = "predicted event prob at horizon",
       y = "observed event proportion")
```

11.36.9 5. Concluding statement

On a simulated external cohort with shifted covariate distribution, a Cox model fit on the development cohort retained discrimination (AUC `round(auc_ext, 2)`) but lost calibration, with IPA falling from `round(ipa_dev, 2)` (development) to `round(ipa_ext, 2)` (external). The calibration plot showed systematic over-prediction in the shifted cohort.

Distribution shift does not always show up in AUC; it often shows up first in calibration. Reporting Brier/IPA alongside AUC catches this failure mode.

11.36.10 Common pitfalls

- Reporting AUC as the only discrimination metric on an external cohort.
- Ignoring censoring when computing a Brier score at a horizon; `pec` or `riskRegression` handle this properly.
- Recalibrating on external data and then reporting the resulting performance as external validation.

11.36.11 Further reading

- Kattan MW, Gerds TA (2018), *The index of prediction accuracy: an intuitive measure useful for evaluating risk prediction models*.
- Steyerberg EW, *Clinical Prediction Models*, ch. 15–17.

11.36.12 Session info

```
sessionInfo()
```

11.36.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)

11.37 Week 3, Session 1 — Time-varying covariates and immortal-time bias

Inference lab using the five-step template: Hypothesis → Visualise → Assumptions → Conduct → Conclude.

11.37.1 Learning objectives

- Recognise immortal-time bias in a survival analysis.
- Correct it using either a landmark analysis or a time-dependent Cox model.
- Describe time-varying covariates in counting-process notation.

11.37.2 Prerequisites

Survival basics; `coxph` syntax.

11.37.3 Background

Immortal-time bias arises when a period during which the outcome cannot occur (by definition of the exposure) is misclassified as follow-up for one of the exposure groups. A classic example: “users of drug X” are defined as anyone who ever received a

prescription; these people must by definition have survived until their first prescription, and if you start their follow-up at baseline rather than at the prescription, you give them "immortal" time that biases the hazard ratio toward protection.

Two standard fixes exist. Landmark analysis picks a time after baseline and classifies exposure based on what happened before that time; follow-up starts at the landmark and the bias disappears. Time-dependent Cox models treat exposure as a function of time and let patients change exposure group when they actually start the drug, using counting-process (start, stop, event) data.

Time-dependent covariates can be external (calendar time, drug availability) or internal (a lab value that evolves with the disease). Internal time-dependent covariates can introduce their own bias when conditioning on them — they may be mediators of the very effect you are studying.

11.37.4 Setup

```
library(tidyverse)
library(survival)
library(ggsurvfit)
set.seed(42)
theme_set(theme_minimal(base_size = 12))
```

11.37.5 1. Hypothesis

We simulate a cohort where every patient has the same true hazard. Some patients start a drug at a random later time. A naive analysis that counts drug users from baseline should produce a spurious protective effect.

11.37.6 2. Visualise

```
n ← 500
dat ← tibble(
  id = seq_len(n),
  t_event = rexp(n, rate = 0.02),        # true event time
  t_drug  = rexp(n, rate = 0.05),        # time of drug start
  ever_drug = t_drug < t_event,          # did they start drug before event?
  t_drug_obs = pmin(t_drug, t_event),
  t_obs   = pmin(t_event, 100),          # admin censoring at 100
  event   = as.integer(t_event ⩽ 100)
)

dat ▷
  ggplot(aes(t_obs, fill = ever_drug)) +
  geom_histogram(bins = 30, alpha = 0.6,
                 position = "identity") +
  labs(x = "Observed time", y = "Count", fill = "Ever drug?")
```

11.37.7 3. Assumptions

Proportional hazards (for Cox), non-informative censoring, and independence of event times across patients. The "ever drug" split is deliberately biased by construction.

11.37.8 4. Conduct

```
# Naive (biased) analysis
fit_naive ← coxph(Surv(t_obs, event) ~ ever_drug, data = dat)
summary(fit_naive)
```

```
# Landmark at time 20: only include patients still at risk at 20,
# classify by drug status as of 20.
landmark ← 20
dat_lm ← dat ▷
  filter(t_obs > landmark) ▷
  mutate(drug_lm = t_drug < landmark,
         t_obs_lm = t_obs - landmark)

fit_lm ← coxph(Surv(t_obs_lm, event) ~ drug_lm, data = dat_lm)
summary(fit_lm)
```

```
# Time-dependent Cox: counting-process format
td ← dat ▷
  transmute(
    id,
    tstart = 0,
    tstop  = if_else(ever_drug, t_drug_obs, t_obs),
    event  = if_else(ever_drug, 0L, event),
    drug   = 0L
  ) ▷
  bind_rows(
    dat ▷
      filter(ever_drug) ▷
      transmute(id,
                tstart = t_drug_obs,
                tstop  = t_obs,
                event  = event,
                drug   = 1L)
  ) ▷
  filter(tstop > tstart) ▷
  arrange(id, tstart)

fit_td ← coxph(Surv(tstart, tstop, event) ~ drug, data = td)
summary(fit_td)
```

11.37.9 5. Concluding statement

The naive Cox analysis produced a hazard ratio of `round(exp(coef(fit_naive)), 2)` for ever-drug — spuriously protective, because immortal time was counted as exposed follow-up. A landmark analysis at time 20 gave HR `round(exp(coef(fit_lm)), 2)`, and a time-dependent Cox model gave HR `round(exp(coef(fit_td)), 2)`, both close to the null as expected from the simulated truth.

11.37.10 Common pitfalls

- Defining exposure over the whole follow-up based on events that occur during it.
- Picking a landmark time after visually inspecting event curves.
- Conditioning on a post-baseline mediator via a time-dependent covariate and still calling it a total effect.

- Forgetting to split the risk set at the exposure change time in counting-process data.

11.37.11 Further reading

- Suissa S (2008), *Immortal time bias in pharmaco-epidemiology*.
- Therneau TM, Grambsch PM. *Modeling Survival Data: Extending the Cox Model*.
- Dafni U (2011), *Landmark analysis at the 25-year landmark point*.

11.37.12 Session info

```
sessionInfo()
```

11.37.13 See also — chapter index

- Methods in this chapter
- Find your method (Chapter 0)