

Package ‘goldilocks’

September 9, 2026

Title Adaptive Trial Designs for Survival and Binary Endpoints

Version 1.0.0

Description Implements Goldilocks adaptive trial designs for time-to-event and fixed-time binary endpoints. Outcomes are generated with a piecewise exponential model, with conjugate Gamma priors used for predictive imputation. Final analyses may use log-rank, Cox, or restricted mean survival time tests, Bayesian piecewise-exponential inference, frequentist risk differences, or Bayesian beta-binomial inference. The method closely follows Broglio and colleagues (2014) [<doi:10.1080/10543406.2014.888569>](https://doi.org/10.1080/10543406.2014.888569) and supports simulation of design operating characteristics.

License GPL-3

Encoding UTF-8

URL <https://graemeleehickey.github.io/goldilocks/>,
<https://github.com/graemeleehickey/goldilocks>,
<https://CRAN.R-project.org/package=goldilocks>

BugReports <https://github.com/graemeleehickey/goldilocks/issues>

Depends R (>= 4.1.0), survival

Imports dplyr, parallel, pbmcapply, PWEALL, Rcpp, rlang, stats

Suggests bench, survRM2, covr, DiagrammeR, testthat (>= 3.0.0), knitr, rmarkdown

LinkingTo BH, Rcpp

Language en-US

RoxygenNote 7.3.3

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation yes

Author Graeme L. Hickey [aut, cre] (ORCID:
 [<https://orcid.org/0000-0002-4989-0054>](https://orcid.org/0000-0002-4989-0054)),
Ying Wan [aut],

Thevaa Chandereng [aut] (ORCID:
<<https://orcid.org/0000-0003-4078-9176>>, bayesDP code as a
template),
Becton, Dickinson and Company [cph],
Tim Kacprowski [ctb] (For code from fastlogrank R package.)

Maintainer Graeme L. Hickey <graemeleehickey@gmail.com>

Repository CRAN

Date/Publication 2026-09-09 14:40:02 UTC

Contents

enrollment	2
evaluate_interim	4
goldilocks	10
plot_enrollment	11
plot_sim_decisions	13
plot_sim_ocs	14
plot_sim_stopping	15
plot_trial_trace	16
ppwe	17
print.goldilocks_calendar_summary	18
print.goldilocks_interim	18
print.goldilocks_trial	19
prop_to_haz	19
pwe_impute	20
pwe_sim	22
randomization	23
sim_comp_data	24
sim_trials	27
summarise_calendar_time	34
summarise_sims	35
summarise_trial_trace	36
survival_adapt	37

Index	49
--------------	-----------

enrollment	<i>Simulate exact continuous-time enrollment</i>
------------	--

Description

Simulates enrollment times from a Poisson process with a piecewise-constant rate.

Usage

```
enrollment(lambda = 1, N_total, lambda_time = NULL)
```

Arguments

lambda	A numeric vector of finite, positive enrollment rates per unit of calendar time. Supply one rate for each interval defined by lambda_time, so length(lambda) must equal length(lambda_time) + 1. The default is 1.
N_total	A required positive integer giving the total sample size.
lambda_time	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior times at which the enrollment rate changes. The initial boundary at time zero is implicit and must not be supplied. Use NULL for a constant enrollment rate.

Details

Major behavior change from goldilocks 0.5.0 and earlier. Versions through 0.5.0 generated Poisson counts in unit-time bins. `enrollment()` returned rebased integer bin times, after which `sim_comp_data()` added independent uniform jitter and sorted the result. From version 0.6.0, enrollment times are drawn directly from the exact continuous-time piecewise-constant Poisson process. Consequently:

- seeded simulations do not reproduce enrollment or downstream trial results obtained with version 0.5.0 or earlier;
- enrollment times and operating-characteristic estimates can change, particularly when rates are low or a rate change is not an integer time;
- the `lambda_time` argument now contains internal change times only: change `lambda_time = 0` to `lambda_time = NULL` for a constant rate, and change `lambda_time = c(0, t1, t2)` to `lambda_time = c(t1, t2)` for a piecewise rate.

`enrollment()` treats time zero as **first patient in**, matching the time origin used throughout goldilocks. The first returned enrollment time is therefore fixed at zero. The remaining `N_total - 1` arrivals form a continuous-time non-homogeneous Poisson process whose rate is constant between the supplied internal knots. Thus, `lambda` is measured in enrollments per unit of `lambda_time`; for example, when time is measured in months, `lambda = 5` means five enrollments per month on average.

Write the internal knots as $0 < \tau_1 < \dots < \tau_K$, with $\tau_0 = 0$ implicit. The enrollment intensity is

$$\lambda(t) = \lambda_j, \quad \tau_{j-1} \leq t < \tau_j,$$

for $j = 1, \dots, K$, and $\lambda(t) = \lambda_{K+1}$ after the final knot. The final rate therefore continues for as long as needed to reach `N_total`; there is no finite accrual horizon in this function. The value of the intensity at an isolated knot does not change the Poisson process. When assigning a realized enrollment time to an interval, goldilocks follows the survival counting-process convention: intervals are open on the left and closed on the right, so an arrival exactly at τ_j belongs to the interval ending at τ_j . The first patient at time zero is a fixed origin and is handled separately.

Arrivals are generated exactly by the time-rescaling theorem. If $E_2, \dots, E_N$ are independent unit-rate exponential variables and $S_i = \sum_{k=2}^i E_k$, then the enrollment times after first patient in are

$$T_1 = 0, \quad T_i = \Lambda^{-1}(S_i),$$

where $\Lambda(t) = \int_0^t \lambda(u) du$ is the cumulative enrollment intensity. This construction gives independent Poisson counts on disjoint calendar intervals, with expected count $\int_a^b \lambda(u) du$ over $(a, b]$. For a constant rate, successive enrollment gaps are independent `Exponential(lambda)` variables.

The rate-change times are measured from first patient in, not from an earlier site-opening or trial-activation date. If operational delays before first patient in are important, they must be modelled separately before using the returned relative times.

For example, `lambda = c(0.3, 0.7, 0.9, 1.2)` with `lambda_time = c(5, 10, 15)` specifies average enrollment rates of 0.3 over positive times in $(0, 5]$, 0.7 over $(5, 10]$, 0.9 over $(10, 15]$, and 1.2 after time 15. Fractional knots such as `lambda_time = 2.5` are handled exactly; no unit-time binning or post-hoc jitter is used.

Value

A non-decreasing numeric vector of `N_total` continuous enrollment times, measured from first patient in and expressed in the same time unit used for `lambda_time`. The first value is zero.

Examples

```
# Constant enrollment: first patient at zero, then exponential gaps.
enrollment(lambda = 0.7, N_total = 10)

# Three internal rate changes define four enrollment intervals.
enrollment(
  lambda = c(0.3, 0.7, 0.9, 1.2),
  N_total = 50,
  lambda_time = c(5, 10, 15)
)

# Fractional change times are supported exactly.
enrollment(
  lambda = c(0.25, 1),
  N_total = 20,
  lambda_time = 2.5
)
```

evaluate_interim

Evaluate an externally observed interim data cut

Description

Applies the same posterior predictive decision calculation used by `survival_adapt()` to subject-level data observed at one interim look. The function does not simulate a trial or modify data. Participants who could still be enrolled before the maximum sample size are represented according to `N_total`, the observed arm counts, and `rand_ratio`.

Usage

```

evaluate_interim(
  data,
  data_cut,
  look,
  N_total,
  end_of_study,
  cutpoints = NULL,
  prior_surv = c(0.1, 0.1),
  prior_bin = c(1, 1),
  bin_method = "mc",
  rand_ratio = c(control = 1, treatment = 1),
  single_arm = FALSE,
  alternative = "greater",
  h0 = 0,
  Fn = 0.05,
  Sn = 0.9,
  prob_ha = 0.95,
  N_impute = 500,
  N_mcmc = 1000,
  mc_conf_level = 0.95,
  empty_interval = c("prior", "propagate", "error"),
  method = "logrank",
  binary_imputation = c("event-time", "bernoulli"),
  seed = NULL,
  Qn = 1,
  rmst_tau = end_of_study,
  prior_surv_final = prior_surv
)

```

Arguments

<code>data</code>	A required data frame with one row per enrolled subject and columns <code>id</code> , <code>treatment</code> , <code>enrollment</code> , <code>time</code> , <code>event</code> , and <code>status</code> . Treatment is coded 1 for treatment and 0 for control; single-arm data use 1. <code>enrollment</code> is measured from first participant randomization, which must be zero, and <code>time</code> is follow-up from that subject's randomization. See Details for the permitted character values in <code>status</code> .
<code>data_cut</code>	A required single finite, non-negative numeric value giving the calendar time of the interim data cut, measured from the same origin and in the same units as <code>enrollment</code> , <code>time</code> , <code>end_of_study</code> , and <code>cutpoints</code> .
<code>look</code>	A required positive integer identifying the prespecified interim look.
<code>N_total</code>	A required positive integer giving the maximum total sample size.
<code>end_of_study</code>	A required single finite, positive numeric value giving the planned follow-up time for each subject.
<code>cutpoints</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior follow-up times defining the piecewise-exponential model used for interim posterior estimation, predictive imputation, and final analysis. The number of

	interval-specific prior columns must be one greater than the number of cutpoints. NULL specifies a constant-hazard analysis model.
prior_surv	A numeric vector, matrix, or named list specifying the Gamma prior for the piecewise-exponential hazards used to generate outcomes during interim prediction. A length-two vector supplies shape and rate and applies the same prior to every arm and interval. A 2 by length(cutpoints) + 1 matrix supplies interval-specific values shared by all arms, with shapes in row 1 and rates in row 2. For independent arm-specific priors, supply a list named control and treatment in a two-arm design, or treatment in a single-arm design. Each list element may be a length-two vector or an interval-specific matrix. Both arms must be supplied; no values are borrowed or filled from the other arm. Rates must use the same time unit as event times, exposure, and cutpoints. The default is <code>c(0.1, 0.1)</code> .
prior_bin	A length-two numeric vector of finite, positive shape parameters <code>c(a, b)</code> for the Beta(<code>a, b</code>) event-probability prior used when <code>method = "bayes-bin"</code> . The same prior is applied to both arms. The default is <code>c(1, 1)</code> , a uniform prior.
bin_method	A single character string selecting how to calculate the posterior probability for <code>method = "bayes-bin"</code> . It must be one of "mc" (Monte Carlo sampling), "normal" (normal approximation), or "quadrature" (numerical integration). The default is "mc". The normal approximation can be inaccurate with sparse events or non-events and posterior event probabilities near 0 or 1. It can change whether <code>prob_ha</code> is exceeded. Increasing <code>N_mcmc</code> does not improve this approximation; use "quadrature" or sufficiently precise "mc" instead.
rand_ratio	A length-two positive integer vector giving the control to treatment allocation ratio at the maximum sample size. The default is <code>c(control = 1, treatment = 1)</code> . Name the values control and treatment; either order is accepted. A legacy unnamed vector is interpreted as <code>c(control, treatment)</code> . The maximum sample size must divide exactly according to this ratio. Ignored for single-arm designs.
single_arm	A single logical value indicating whether the design has one treatment arm and no control arm. The default is FALSE.
alternative	A single character string specifying the alternative hypothesis. It must be one of "greater" (the default), "less", or "two.sided". One-sided alternatives ("greater" and "less") are supported for <code>method = "bayes-surv"</code> and <code>method = "bayes-bin"</code> . All three options are supported for <code>method = "logrank"</code> , <code>method = "cox"</code> , <code>method = "rmst"</code> , <code>method = "riskdiff-wald"</code> , and <code>method = "riskdiff-fm"</code> . For an adverse event, benefit is in the "greater" direction for RMST (longer event-free time) and the "less" direction for the other methods (lower hazard or event probability).
h0	A single finite numeric value specifying the null hypothesis or margin. The default is 0. For Bayesian analyses, <code>h0</code> must lie in <code>[0, 1]</code> for a single-arm design and <code>[-1, 1]</code> for a two-arm design. When <code>method = "bayes-surv"</code>, <code>h0</code> is the null value of $p_{\text{treatment}} - p_{\text{control}}$. In a single-arm design, <code>h0</code> is the external benchmark event probability, often referred to as a performance goal (PG) or objective performance criterion (OPC).

	• When <code>method = "bayes-bin"</code>, h_0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$ for a two-arm design, or the null event probability for a single-arm design. • When <code>method = "cox"</code>, h_0 is the null log hazard ratio for treatment versus control. Use $h_0 = 0$ for the usual hazard ratio of 1 null, or $h_0 = \log(\text{margin})$ for a non-inferiority margin specified as a hazard ratio. A Cox non-inferiority test should usually use <code>alternative = "less"</code>. • When <code>method = "rmst"</code>, h_0 is the null treatment-control RMST difference in time units and must lie in $[-\text{rmst_tau}, \text{rmst_tau}]$. For non-inferiority allowing a loss of m time units, use $h_0 = -m$ and <code>alternative = "greater"</code>. • When <code>method = "riskdiff-wald"</code> or <code>method = "riskdiff-fm"</code>, h_0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$ and must lie in $[-1, 1]$. • When <code>method = "logrank"</code>, only $h_0 = 0$ is supported; this denotes the usual equal-survival null. Nonzero values are rejected because the standard log-rank statistic does not implement a nonzero effect margin.
<code>Fn</code>	NULL, or a single numeric probability in $[0, 1]$ giving the threshold for stopping for futility at this look. Futility is declared when predictive success at the maximum sample size is strictly less than <code>Fn</code> . Set <code>Fn = 0</code> or NULL to disable the maximum-sample calculation. The default is <code>0.05</code> .
<code>Sn</code>	A single numeric probability in $[0, 1]$ giving the threshold for stopping for expected success at this look. Expected success is declared when predictive success among the currently enrolled participants is strictly greater than <code>Sn</code> . The default is <code>0.9</code> .
<code>prob_ha</code>	A single numeric probability in $[0, 1]$ defining success in each completed-data analysis. For Bayesian methods this is compared with the posterior probability of the alternative; for frequentist methods it is compared with $1 - P$. The default is <code>0.95</code> .
<code>N_impute</code>	A positive integer giving the number of predictive imputations used at each interim look and, when requested, for final multiple imputation. The default is <code>500</code> . An imputed Cox, RMST, or risk-difference final analysis requires at least two.
<code>N_mcmc</code>	A positive integer giving the number of posterior draws used within each method = <code>"bayes-surv"</code> and by method = <code>"bayes-bin"</code> when <code>bin_method = "mc"</code> . The default is <code>1000</code> .
<code>mc_conf_level</code>	A single numeric probability strictly between <code>0.5</code> and <code>1</code> , giving the confidence level for one-sided exact binomial bounds reported as diagnostics of finite Monte Carlo uncertainty. The bounds do not alter completed-data success classifications or interim decisions, which use strict point-estimate comparisons with <code>prob_ha</code> , <code>Qn</code> , <code>Sn</code> , and <code>Fn</code> . The default is <code>0.95</code> .
<code>empty_interval</code>	A single character string specifying how to handle empty piecewise-exponential intervals when updating Gamma hazard models for predictive imputation and Bayesian survival analysis. An empty interval is an interval with no exposed subjects in a treatment arm at the analysis time. <code>"prior"</code> (the default) leaves the interval at zero exposure time and zero events, so its posterior is driven only by its assigned survival prior. <code>"propagate"</code> is a legacy heuristic that copies exposure time and event counts from the nearest non-empty interval in the same treatment arm and emits a warning. <code>"error"</code> stops when any empty interval is found.

method	A single character string specifying the completed-data and final analysis. Available choices are a log-rank (method = "logrank") test, Cox proportional hazards regression model Wald test (method = "cox"), a restricted mean survival time difference Wald test (method = "rmst"), a fully-Bayesian piecewise-exponential analysis (method = "bayes-surv"), a Bayesian beta-binomial analysis of complete binary outcomes (method = "bayes-bin"), a frequentist risk-difference Wald test (method = "riskdiff-wald"), or a Farrington-Manning score test (method = "riskdiff-fm") of complete binary outcomes. The deprecated method = "riskdiff" is accepted as an alias for "riskdiff-wald" with a warning. The default is "logrank". See Details.
binary_imputation	A single character string selecting the predictive imputation approach for method = "bayes-bin", method = "riskdiff-wald", or method = "riskdiff-fm". "event-time" (the default) draws a conditional piecewise-exponential event time and reduces it to event status at end_of_study. "bernoulli" draws the endpoint status directly from its conditional event probability. This argument is ignored for time-to-event analysis methods.
seed	NULL (the default), or a single non-negative integer used for the predictive Monte Carlo calculation. A supplied seed makes the result reproducible and leaves the existing random-number state unchanged. With seed = NULL, the call uses and advances the current random-number state.
Qn	A single numeric probability in $[0, 1]$ giving the upper threshold for declaring immediate trial success at this look. Immediate success is declared when predictive success among the currently enrolled participants is strictly greater than Qn. Qn must be greater than or equal to Sn. The default, 1, disables immediate-success stopping.
rmst_tau	A single finite positive restriction time for method = "rmst", in the same units as end_of_study. Defaults to end_of_study and must not exceed it. Prespecify the same horizon for all looks, imputations, and simulations. It may precede analysis cutpoints and does not shorten the planned follow-up or imputation horizon. Ignored for other methods.
prior_surv_final	A numeric vector, matrix, or named list specifying the Gamma analysis prior for each hypothetical completed trial when method = "bayes-surv". It accepts the same shared, interval-specific, and arm-specific forms as prior_surv and defaults to prior_surv. It should match the prior for the actual final analysis. Other methods do not use it in this interim calculation.

Details

The arm-specific maximum enrollment is $N_{\text{total}} * \text{rand_ratio} / \text{sum}(\text{rand_ratio})$. Potential future accrual in each arm is its maximum enrollment minus its observed enrollment. Randomization block information and concealed future assignment order are unnecessary: the maximum-sample predictive calculation requires only the number of potential future participants in each arm.

The `data$status` values distinguish observed follow-up: "event" for an observed endpoint event, "complete" for event-free completion of end_of_study, "pending" for a subject still under follow-up, and "censored" for permanent early censoring. Pending and censored outcomes are predictively imputed conditional on time.

With `method = "bayes-surv"`, `prior_surv` generates predictive outcomes and `prior_surv_final` is the analysis prior used to test each hypothetical completed trial for success, at both the current and maximum sample sizes. Supply the same final prior as in the prespecified trial design. The default `prior_surv_final = prior_surv` uses one prior for both roles. Other methods do not use `prior_surv_final` in this interim calculation; Bayesian binary completed-data analyses use `prior_bin`.

Q_n , S_n , and F_n are scalar thresholds for this look. Immediate success is declared when the estimated probability of completed-data success among the current participants is strictly greater than Q_n . Otherwise, expected success stops accrual for planned follow-up when that probability is strictly greater than S_n . If neither success rule applies, futility is declared when the corresponding maximum-sample probability is strictly less than F_n . Set $F_n = 0$ or `NULL` to disable futility. Exact one-sided Monte Carlo bounds are returned as diagnostics and do not drive any decision.

The function requires treatment assignments to perform the arm-specific posterior and completed-data analyses. In a blinded trial, an independent unblinded statistician or service should join the treatment assignments, run this function, and return the aggregate decision without distributing the subject-level input.

Value

An object of class `goldilocks_interim`, containing:

- `decision`: a one-row decision summary;
- `probabilities`: current- and maximum-sample predictive probabilities;
- `monte_carlo`: estimates, standard errors, and diagnostic bounds;
- `diagnostics`: observed status counts, potential accruals, warnings, imputation diagnostics, and resolved Gamma prior and posterior parameters by arm and interval;
- `trace`: a one-row decision trace compatible with `plot_trial_trace()` and `summarise_trial_trace()`;
- `metadata`: the evaluated design, resolved prior design, package version, time-origin, data-cut, and random-number policy. For `method = "bayes-surv"`, both Gamma priors are retained in `metadata$design` and `metadata$prior_design`; posterior diagnostics describe the predictive model based on the observed interim data. For `method = "bayes-bin"`, `metadata$design` retains the normalized imputation prior (`prior_surv`), completed-data analysis prior (`prior_bin`), and imputation horizon (`end_of_study`).

Predictive and analysis priors

For `method = "bayes-surv"`, `prior_surv_final` is used **during interim calculations as well as at the actual final analysis**. The two arguments specify different roles, not simply different calendar stages:

Calculation	Gamma prior used
At interim, generate outstanding outcomes for enrolled and future participants	<code>prior_surv</code>
At interim, test each hypothetical completed trial at the current or maximum sample size	<code>prior_surv_final</code>
At final analysis, impute missing outcomes if <code>imputed_final = TRUE</code>	<code>prior_surv_final</code>
Analyze the actual final trial data	<code>prior_surv_final</code>

Within one interim predictive replicate, first update `prior_surv` with the observed events and exposure, draw hazards, and generate outstanding outcomes. Then start a fresh analysis posterior using `prior_surv_final` and the completed dataset's events and exposure. Compare its posterior probability of the alternative with `prob_ha`. The proportion of replicates that pass is the predictive probability used by `Qn`, `Sn`, and `Fn`.

To incorporate external evidence in prediction while using a weak analysis prior, explicitly supply an informative `prior_surv` and the chosen weak `prior_surv_final`. **Omitting `prior_surv_final` uses `prior_surv` for both roles; the package does not automatically weaken the analysis prior.** The predictive prior can still affect the selected sample size and stopping decision, so calibrate the design using both prespecified priors.

This table describes Bayesian survival analysis. For `method = "bayes-bin"`, completed-data success tests at interim and final use `prior_bin`; `prior_surv_final` governs only optional final imputation. Frequentist completed-data tests use no analysis prior. `evaluate_interim()` performs the two interim calculations; use the same prior arguments as in the simulated design.

Examples

```
interim_data <- data.frame(
  id = 1:6,
  treatment = c(0, 1, 0, 1, 0, 1),
  enrollment = c(0, 1, 2, 3, 4, 5),
  time = c(6, 5, 4, 3, 2, 1),
  event = c(1, 0, 0, 1, 0, 0),
  status = c("event", "pending", "pending", "event", "pending", "pending")
)

evaluate_interim(
  data = interim_data,
  data_cut = 6,
  look = 1,
  N_total = 10,
  end_of_study = 12,
  rand_ratio = c(control = 1, treatment = 1),
  alternative = "less",
  N_impute = 5,
  seed = 2026
)
```

Description

Implements the Goldilocks Bayesian adaptive design proposed by Broglio et al. (2014) for single-arm and randomized two-arm trials. Outcomes are generated with an underlying piecewise-exponential event-time model. Final analyses may retain the time-to-event outcome (`method = "logrank"`,

"cox", "rmst", or "bayes-surv") or reduce complete follow-up to event status at a fixed endpoint time (method = "riskdiff-wald", "riskdiff-fm", or "bayes-bin").

The method can be used for a confirmatory trial to select a sample size based on accumulating data. During accrual, predictive probabilities are used to determine whether the current sample size is sufficient, whether continuing accrual would be futile, or whether enrollment should continue. The algorithm explicitly accounts for completion of planned follow-up before the primary analysis. Broglio et al. (2014) refer to this as a Goldilocks trial design, as it is constantly asking the question, **“Is the sample size too big, too small, or just right?”**

References

Broglio KR, Connor JT, Berry SM. Not too big, not too small: a Goldilocks approach to sample size selection. *Journal of Biopharmaceutical Statistics*, 2014; **24(3)**: 685–705.

plot_enrollment	<i>Plot an enrollment projection</i>
-----------------	--------------------------------------

Description

Draws the expected cumulative enrollment curve for a Goldilocks trial design, together with optional random enrollment trajectories and projected interim and maximum-sample-size milestones.

Usage

```
plot_enrollment(
  x = NULL,
  lambda = NULL,
  N_total = NULL,
  lambda_time = NULL,
  interim_look = NULL,
  end_of_study = NULL,
  n_sim = 20L,
  seed = NULL,
  time_unit = NULL,
  xlab = NULL,
  ylab = "Cumulative number of enrolled patients",
  main = NULL,
  annotate = TRUE,
  projection_col = "#276E9B",
  simulation_col = "#777777",
  milestone_col = "#C8682A"
)
```

Arguments

x	NULL (the default), or a result returned by survival_adapt() or sim_trials() . Results created by current versions of goldilocks retain the evaluated enrollment design needed by this function.
lambda	NULL (the default), or a numeric vector of finite, positive enrollment rates per unit of calendar time. It is required when x = NULL and otherwise overrides the rates stored in x.
N_total	NULL (the default), or a positive integer giving the maximum total sample size. It is required when x = NULL and otherwise overrides the value stored in x.
lambda_time	NULL (the default), or a numeric vector of finite, positive, strictly increasing calendar times at which the enrollment rate changes. See enrollment() .
interim_look	NULL (the default), or a strictly increasing positive integer vector giving the cumulative enrollment at each interim look. All values must be less than N_total.
end_of_study	NULL (the default), or a single finite, positive numeric value giving the planned follow-up time for each subject. When available and annotate = TRUE, it is reported beneath the plot.
n_sim	A single non-negative integer giving the number of random enrollment trajectories to draw. The default is 20L; use 0 to show only the expected enrollment curve.
seed	NULL (the default), or a single integer between 0 and .Machine\$integer.max for the random trajectories. A supplied seed gives reproducible trajectories and leaves the existing random-number state unchanged.
time_unit	NULL (the default), or a non-empty character string naming the design's unit of time, such as "months" or "days".
xlab	NULL (the default), or a character string for the horizontal axis label. When NULL, the label is constructed from time_unit.
ylab	A character string for the vertical axis label. The default is "Cumulative number of enrolled patients".
main	NULL (the default), or a character string for the main title.
annotate	A single logical value indicating whether follow-up and simulation notes should appear beneath the plot. The default is TRUE.
projection_col	A character string specifying the colour of the expected enrollment curve. The default is "#276E9B".
simulation_col	A character string specifying the colour of the random enrollment trajectories. The default is "#777777".
milestone_col	A character string specifying the colour of the interim and maximum-sample-size guides. The default is "#C8682A".

Details

The blue projection is $1 + \Lambda(t)$, where $\Lambda(t)$ is the cumulative intensity of the piecewise-constant Poisson enrollment process. The first patient is fixed at time zero, consistently with [enrollment\(\)](#). A milestone's projected time solves $1 + \Lambda(t) = N$. With a constant enrollment rate this is also the

mean arrival time, $(N - 1)/\lambda$. With a piecewise rate it is an expected-count projection rather than the mean of the corresponding arrival-time distribution.

If `x` supplies a stored design, explicitly supplied design arguments override the corresponding stored values. This makes it possible, for example, to compare a fitted design with a different enrollment rate.

Value

Invisibly, a list containing the evaluated design, the projection data frame, the milestones data frame, and the simulated enrollment-time vectors in simulations.

Examples

```
plot_enrollment(
  lambda = 20,
  N_total = 600,
  interim_look = 400,
  end_of_study = 12,
  n_sim = 20,
  seed = 20260727,
  time_unit = "months"
)

# Piecewise enrollment rates are supported.
plot_enrollment(
  lambda = c(8, 20),
  lambda_time = 6,
  N_total = 200,
  interim_look = c(100, 150),
  n_sim = 5,
  seed = 1,
  time_unit = "months"
)
```

plot_sim_decisions	<i>Plot predictive-probability decision maps</i>
--------------------	--

Description

Draws one decision map per interim look across simulated trials. The horizontal axis is the predictive probability of success if enrollment continues to the maximum sample size, and the vertical axis is the predictive probability of success if enrollment stops at the current sample size. Shaded regions and dashed lines show the futility, continuation, stopping-accrual-for-expected-success, and immediate-success rules. Identical points are aggregated; point size indicates their frequency.

Usage

```
plot_sim_decisions(x)
```

Arguments

x A required result returned by `sim_trials()` with `return_trace = TRUE`, or the corresponding traces data frame. The trace must contain interim-look numbers, predictive probabilities, decision thresholds, and decisions.

Details

The upper shaded region represents $\text{ppp_stop_now} > \text{immediate_success_threshold}$; the middle region represents $\text{success_threshold} < \text{ppp_stop_now} \leq \text{immediate_success_threshold}$. Futility applies below or at the expected-success threshold when the maximum-sample-size prediction is strictly below its threshold. A legacy trace without `immediate_success_threshold` is treated as if that threshold were one, which disables immediate success.

Value

The simulation traces, invisibly.

Examples

```
traces <- data.frame(
  trial = 1:6,
  look = rep(c(1, 2), each = 3),
  planned_N = rep(c(40, 60), each = 3),
  ppp_stop_now = c(0.96, 0.4, 0.2, 0.92, 0.5, 0.1),
  success_threshold = rep(c(0.95, 0.9), each = 3),
  immediate_success_threshold = 0.99,
  ppp_success_at_max = c(0.8, 0.5, 0.02, 0.85, 0.4, 0.01),
  futility_threshold = 0.05,
  decision = c(
    "stop_expected_success", "continue", "stop_futility",
    "stop_expected_success", "continue", "stop_futility"
  )
)
plot_sim_decisions(traces)
```

plot_sim_ocs

Plot operating characteristics across simulation scenarios

Description

Draws operating-characteristic curves across a series of true treatment-effect scenarios. The first panel shows final success and stopping probabilities. The second panel shows mean enrolled sample size.

Usage

```
plot_sim_ocs(x, effect, xlab = "True treatment effect")
```

Arguments

x	A required data frame returned by <code>summarise_sims()</code> , with one row per simulation scenario and columns power, stop_success, stop_futility, stop_max_N, and mean_N. When present, stop_immediate_success is plotted separately from stop_success, which retains its historical expected-success meaning.
effect	A required numeric vector containing one finite treatment-effect value per row of x, or a single character string naming such a numeric column in x.
xlab	A single character string for the treatment-effect axis label. The default is "True treatment effect".

Value

x, invisibly.

Examples

```
operating_characteristics <- data.frame(
  scenario = c("null", "small", "target"),
  effect = c(1, 0.85, 0.7),
  power = c(0.025, 0.55, 0.9),
  stop_success = c(0.01, 0.35, 0.75),
  stop_futility = c(0.7, 0.25, 0.05),
  stop_max_N = c(0.29, 0.4, 0.2),
  mean_N = c(180, 250, 210),
  sd_N = c(45, 60, 55),
  stop_and_fail = c(0.001, 0.01, 0.02)
)
plot_sim_ocs(
  operating_characteristics,
  effect = "effect",
  xlab = "True hazard ratio"
)
```

plot_sim_stopping

Plot stopping outcomes from trial simulations

Description

Draws a stacked bar chart of stopping outcomes by enrolled sample size, with colours distinguishing immediate success, stopping accrual for expected success, futility, and maximum-sample-size outcomes. The type argument controls whether the function draws marginal, conditional, or cumulative bars, or a flowchart through successive interim looks. Bar-chart subtitles state the denominator used by the selected view. The input can be the sims element returned by `sim_trials()` or the complete `sim_trials()` result.

Usage

```
plot_sim_stopping(
  x,
  type = c("marginal", "conditional", "cumulative", "flowchart")
)
```

Arguments

x	A required simulation-result data frame, or the complete list returned by sim_trials() .
type	A single character string specifying the percentages to plot. "marginal" (the default) shows the percentage of all simulated trials ending at each sample size; its bars sum to 100 percent across sample sizes. "conditional" shows the percentage stopping at each look among trials still active at the start of that look. "cumulative" shows the status of all simulated trials after each look; every bar sums to 100 percent and includes trials continuing to the next look. "flowchart" starts with all simulated trials and branches at each look into futility, continued enrollment, expected-success, and immediate-success nodes labelled with trial counts.

Details

The marginal view uses terminal sample sizes observed in `N_enrolled`. When the complete result from `sim_trials(return_trace = TRUE)` is supplied, the conditional, cumulative, and flowchart views also include sample sizes recorded in traces, so reached looks at which no trial stopped still appear. The flowchart requires the `N_max` column and is rendered with [DiagrammeR::grViz\(\)](#).

Value

For bar-chart types, the simulation result data frame, invisibly. For `type = "flowchart"`, a [DiagrammeR](#) `grViz` htmlwidget.

<code>plot_trial_trace</code>	<i>Plot predictive probabilities and enrollment at interim looks</i>
-------------------------------	--

Description

Draws three base-R panels showing the predictive probability of success if accrual stops now, the predictive probability of success at the maximum sample size, and enrollment and observed events by treatment arm. Thresholds and early stopping decisions are marked on the probability panels.

Usage

```
plot_trial_trace(x)
```

Arguments

x	A required <code>goldilocks_trial</code> result from survival_adapt() , a <code>goldilocks_interim</code> result from evaluate_interim() , or an interim trace data frame.
---	--

Value

The trace data frame, invisibly.

ppwe	<i>Calculate endpoint event probabilities from piecewise hazards</i>
------	--

Description

Calculates the cumulative event probability at a fixed follow-up time for one or more sets of piecewise-constant hazard rates.

Usage

```
ppwe(hazard, end_of_study, cutpoints = NULL)
```

Arguments

hazard	A required numeric matrix of finite, non-negative hazard rates. Rows represent parameter sets, such as posterior draws, and columns represent the intervals defined by cutpoints. The number of columns must equal <code>length(cutpoints) + 1</code> , and at least one row is required.
end_of_study	A required single finite, positive numeric time at which the cumulative event probability is evaluated. It must be greater than every cutpoint.
cutpoints	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior times at which the hazard rate changes. The number of hazard rates must be one greater than the number of cutpoints. Use NULL for a constant hazard.

Details

The cumulative probability depends on interval durations, so the value assigned to an isolated cutpoint has no effect. PWEALL represents its generating hazard with pieces closed on the left and open on the right. When goldilocks assigns realized event times to analysis intervals, it instead uses the survival counting-process convention, open on the left and closed on the right.

Value

A numeric vector of event probabilities in $[0, 1]$, with one value for each row of hazard.

```
print.goldilocks_calendar_summary
```

Print a calendar-time operating-characteristic summary

Description

Print a calendar-time operating-characteristic summary

Usage

```
## S3 method for class 'goldilocks_calendar_summary'
print(x, digits = 1, ...)
```

Arguments

x	A goldilocks_calendar_summary result returned by summarise_calendar_time() .
digits	A single non-negative integer giving the number of digits after the decimal point in displayed values. The default is 1.
...	Additional arguments; currently ignored.

Value

x, invisibly.

```
print.goldilocks_interim
```

Print an externally evaluated interim analysis

Description

Prints the predictive probabilities and decision from an interim analysis returned by [evaluate_interim\(\)](#).

Usage

```
## S3 method for class 'goldilocks_interim'
print(x, ...)
```

Arguments

x	A goldilocks_interim result returned by evaluate_interim() .
...	Additional arguments; currently ignored.

Value

x, invisibly.

```
print.goldilocks_trial
```

Print a Goldilocks adaptive trial result

Description

Prints the final trial summary and reports how many interim looks were completed for a `goldilocks_trial` object returned by `survival_adapt()`.

Usage

```
## S3 method for class 'goldilocks_trial'
print(x, ...)
```

Arguments

<code>x</code>	A <code>goldilocks_trial</code> result returned by <code>survival_adapt()</code> when <code>return_trace = TRUE</code> .
<code>...</code>	Additional arguments passed to <code>base::print.data.frame()</code> when printing the final trial summary.

Value

The input object, invisibly.

<code>prop_to_haz</code>	<i>Derive piecewise-constant hazard rates from cumulative event probabilities</i>
--------------------------	---

Description

Converts cumulative event probabilities at specified follow-up times into the corresponding piecewise-constant hazard rates. This is useful for expressing data-generation assumptions in terms of clinically interpretable event probabilities.

Usage

```
prop_to_haz(probs, cutpoints = NULL, endtime)
```

Arguments

probs	A required numeric vector of finite cumulative event probabilities in $[0, 1)$ at each cutpoint and at endtime, in that order. Its length must be one greater than the number of cutpoints. With no cutpoints, supply a single probability at endtime. Values must be non-decreasing and are not recycled.
cutpoints	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior times at which the event hazard changes. NULL corresponds to a simple (non-piecewise) exponential model.
endtime	A required single finite, positive numeric value giving the follow-up time corresponding to the final element of probs. It must be later than every cutpoint and use the same time unit.

Details

Given $J - 1$ interior cutpoints, then there are J intervals defined as: $[s_0, s_1), [s_1, s_2), \dots, [s_{J-1}, s_J)$, with conditions that $s_0 = 0$ and $s_J = \infty$. Each interval corresponds to constant hazard λ_j . This is the PWEALL representation of the continuous generating hazard. Changing the value at an isolated cutpoint does not alter the cumulative probabilities calculated here. When observed event times are assigned to analysis intervals, goldilocks uses $(s_{j-1}, s_j]$, matching the survival counting-process convention, so an event at s_j belongs to the interval ending there.

Value

A numeric vector of non-negative hazard rates, with one value for each interval defined by cutpoints and endtime.

Examples

```
lambda <- prop_to_haz(0.15, endtime = 36) # 15% probability at 36-months
all.equal(pexp(36, lambda), 0.15)

# 15% probability at 12-months, and 30% at 24-months
prop_to_haz(c(0.15, 0.30), 12, 24)
PWEALL::pwe(12, prop_to_haz(c(0.15, 0.30), 12, 24), c(0, 12))$dist
PWEALL::pwe(24, prop_to_haz(c(0.15, 0.30), 12, 24), c(0, 12))$dist
```

pwe_impute

Impute piecewise exponential time-to-event outcomes

Description

Draws an event time from a piecewise-exponential distribution conditional on a subject remaining event-free through the observed follow-up time, with optional administrative censoring.

Usage

```
pwe_impute(time, hazard, cutpoints = NULL, maxtime = NULL)
```

Arguments

time	A required numeric vector of finite, non-negative event-free follow-up times for subjects who have not had an event. When maxtime is supplied, no value may exceed it. A zero-length vector returns a zero-row data frame. Values are not recycled against other arguments.
hazard	A required numeric vector of finite, non-negative event rates, with one value per interval defined by cutpoints. If the final rate is zero, maxtime must be supplied.
cutpoints	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior times at which the hazard rate changes. The number of hazard rates must be one greater than the number of cutpoints. Use NULL for a constant hazard.
maxtime	NULL (the default), or a single finite, positive numeric administrative censoring time. When supplied, it must be later than every cutpoint.

Details

If a subject is event-free at time $s < t$, then the conditional probability is

$$F_{T|s}(t|s) = P(T \leq t | T > s) = \frac{F(t) - F(s)}{1 - F(s)}$$

where $F(\cdot)$ is the cumulative distribution function of the piecewise exponential (PWE) distribution. Equivalently, $F(t) = 1 - S(t)$, where $S(t)$ is the survival function. If $U \sim Unif(0, 1)$, then we can generate an event time (conditional on being event free up until s) as

$$F^{-1}(U(1 - F(s)) + F(s))$$

If $s = 0$, this is equivalent to a direct unconditional sample from the PWE distribution.

PWEALL represents the generating hazard with pieces closed on the left and open on the right. Its cumulative distribution is continuous at every cutpoint, so this endpoint choice does not affect imputation. For assigning realized event times to analysis intervals, goldilocks uses the survival counting-process convention, open on the left and closed on the right; an event exactly at a cutpoint belongs to the interval ending there.

Value

A data frame with one row per subject and columns time, the imputed event or censoring time, and event, coded 1 for an event and 0 for administrative censoring.

Examples

```
pwe_impute(time = c(3, 4, 5), hazard = c(0.002, 0.01), cutpoints = 12)
pwe_impute(time = c(3, 4, 5), hazard = c(0.002, 0.01), cutpoints = 12,
            maxtime = 36)
pwe_impute(time = 19.621870008, hazard = c(2.585924e-02, 3.685254e-09),
            cutpoints = 12, maxtime = 36)
```

pwe_sim	<i>Simulate piecewise exponential time-to-event outcomes</i>
---------	--

Description

Simulates event times from a piecewise-exponential distribution, with optional administrative censoring at a fixed follow-up time.

Usage

```
pwe_sim(n = 1, hazard = 1, cutpoints = NULL, maxtime = NULL)
```

Arguments

n	A single non-negative integer giving the number of event times to simulate. The default is 1; $n = 0$ returns a zero-row data frame.
hazard	A numeric vector of finite, non-negative event rates, with one value per interval defined by cutpoints. The default is 1, giving a constant unit rate. If at least one outcome is requested and the final rate is zero, <code>maxtime</code> must be supplied so that subjects without an event can be administratively censored.
cutpoints	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior times at which the hazard rate changes. The number of hazard rates must be one greater than the number of cutpoints. Use NULL for a constant hazard.
maxtime	NULL (the default), or a single finite, positive numeric administrative censoring time. When supplied, it must be later than every cutpoint.

Details

PWEALL represents the generating hazard with pieces closed on the left and open on the right. Because the event-time distribution is continuous, the value of the hazard at an isolated cutpoint does not alter the cumulative hazard, distribution, or generated samples. When realized event times are later assigned to analysis intervals, `goldilocks` follows the survival counting-process convention, open on the left and closed on the right, so an event exactly at a cutpoint belongs to the interval ending there. See [pwe_impute\(\)](#) for the conditional sampling details.

Value

A data frame with one row per simulated subject and columns `time`, the event or censoring time, and `event`, coded 1 for an event and 0 for administrative censoring.

Examples

```
pwe_sim(10, hazard = c(0.005, 0.001), cutpoints = 3, maxtime = 36)
y <- pwe_sim(n = 1, hazard = c(2.585924e-02, 3.685254e-09),
             cutpoints = 12)
```

randomization	<i>Generate a block-randomized treatment sequence</i>
---------------	---

Description

Generates a randomized treatment assignment sequence for control and treatment arms using a fixed allocation ratio and one or more permitted block sizes.

Usage

```
randomization(N_total, block = 2, allocation = c(control = 1, treatment = 1))
```

Arguments

<code>N_total</code>	A required positive integer giving the total number of treatment assignments.
<code>block</code>	A positive integer vector containing one or more permitted block sizes. Every block size must be a multiple of <code>sum(allocation)</code> . The default is 2.
<code>allocation</code>	A length-two positive integer vector giving the control to treatment allocation ratio. The default is <code>c(control = 1, treatment = 1)</code> . Name the values <code>control</code> and <code>treatment</code> ; either supplied order is accepted and matched by name. A legacy unnamed vector remains accepted in <code>c(control, treatment)</code> order. Unequal unnamed values produce a warning because names may be required in a future major release.

Details

Complete randomization may not always be ideal due to the chance of drawing a large block assigned to one treatment group, potentially impacting the time to enrollment completion. Therefore, a block randomization allocation may be preferable. The block randomization allocation specification allows for different two-arm randomization ratios, but they must be given in integer form. For every value `b` in `block`, the required relationship is `b %% sum(allocation) == 0`; see the equal- and unequal-allocation examples below.

Value

An integer treatment assignment vector, coded 0 for control and 1 for treatment.

Examples

```
# Implementing treatment allocation for control to treatment with 1:1.5
# randomization ratio
randomization(
  N_total = 100,
  block = 5,
  allocation = c(control = 2, treatment = 3)
)

# Treatment allocation with 2:1 for control to treatment
```

```

randomization(
  N_total = 70,
  block = 9,
  allocation = c(treatment = 1, control = 2)
)

# Treatment allocation for control to treatment with 1:2 for control
# to treatment with multiple block sizes c(3, 9, 6)
randomization(
  N_total = 100,
  block = c(3, 9, 6),
  allocation = c(control = 1, treatment = 2)
)

# For complete randomization set the N_total to block size
randomization(
  N_total = 100,
  block = 100,
  allocation = c(control = 1, treatment = 1)
)

# randomization() is a two-arm helper; a multi-arm allocation is rejected.
try(randomization(
  N_total = 60,
  block = 6,
  allocation = c(1, 1, 1)
), silent = TRUE)

```

sim_comp_data

Simulate complete trial data under piecewise-exponential event rates

Description

Simulates enrollment, treatment allocation, event or censoring times, and loss to follow-up for a single-arm or randomized two-arm trial. Event times follow a piecewise-exponential distribution within each arm.

Usage

```

sim_comp_data(
  hazard_treatment,
  hazard_control = NULL,
  generation_cutpoints = NULL,
  N_total,
  lambda = 0.3,
  lambda_time = NULL,
  end_of_study,
  block = 2,
  rand_ratio = c(control = 1, treatment = 1),

```

```
    prop_loss = 0
  )
```

Arguments

<code>hazard_treatment</code>	A required numeric vector of finite, non-negative event rates for the treatment arm. Supply one rate per interval defined by <code>generation_cutpoints</code> ; a single value specifies a constant event rate.
<code>hazard_control</code>	NULL (the default) for a single-arm trial, or a numeric vector of finite, non-negative event rates for the control arm in a two-arm trial. It must contain one rate per interval defined by <code>generation_cutpoints</code> .
<code>generation_cutpoints</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior follow-up times at which the data-generating hazard changes. The number of hazards for each arm must be one greater than the number of generation cutpoints. NULL specifies a constant-hazard data-generating model.
<code>N_total</code>	A required positive integer giving the maximum total sample size.
<code>lambda</code>	A numeric vector of finite, positive enrollment rates per unit of calendar time. Supply one rate for each interval defined by <code>lambda_time</code> . The default is 0.3. See enrollment() for the continuous-time enrollment model and time origin.
<code>lambda_time</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing calendar times at which the enrollment rate changes. Time zero is implicit, and <code>length(lambda)</code> must equal <code>length(lambda_time) + 1</code> .
<code>end_of_study</code>	A required finite, positive numeric value giving the planned follow-up time for each subject. It must be later than the final <code>generation_cutpoints</code> value and use the same time unit.
<code>block</code>	A positive integer vector of permitted randomization block sizes. Every value must be a multiple of <code>sum(rand_ratio)</code> . The default is 2 and the argument is ignored for a single-arm trial.
<code>rand_ratio</code>	A length-two positive integer vector giving the control to treatment randomization ratio. The default is <code>c(control = 1, treatment = 1)</code> . Name the values <code>control</code> and <code>treatment</code> ; either supplied order is accepted and matched by name. A legacy unnamed vector remains accepted in <code>c(control, treatment)</code> order. Unequal unnamed values produce a warning because names may be required in a future major release. See randomization() for more details.
<code>prop_loss</code>	A numeric vector containing one or two probabilities in $[0, 1)$. Each value is the dropout-time CDF at <code>end_of_study</code> : $P(D \leq \tau) = p$, where τ is the planned follow-up duration per subject. Independently of event time and enrollment, each subject's dropout time D is exponentially distributed with rate $-\log(1 - p)/\tau$. The observed time is the minimum of event time, dropout time, and <code>end_of_study</code> ; an event occurring before dropout is retained. Thus, <code>prop_loss</code> is not the expected proportion actually censored by dropout: that proportion can be lower because events occur first, and the realized number of dropouts varies between trials. A single value applies the same dropout distribution to every arm. For a two-arm design, supply a length-two vector named

control and treatment for arm-specific probabilities; supplied order does not matter. Single-arm designs require one probability. The default 0 sets dropout time to infinity without drawing random numbers. A value of 1 is rejected because it requires an infinite exponential rate.

Details

Enrollment is simulated directly in continuous time by `enrollment()`. The first patient is placed at time zero and all subsequent enrollment times are measured from first patient in. No uniform jitter is added in `sim_comp_data()`.

`lambda_time` and `generation_cutpoints` both contain internal change times, but they describe different clocks. `lambda_time` describes changes in the trial's calendar-time enrollment rate measured from first patient in. `generation_cutpoints` describes changes in an individual subject's event hazard measured from that subject's enrollment. They need not have the same values or length. All time quantities supplied to a simulation should nevertheless use one common unit, such as days or months.

PWEALL represents the continuous generating hazard with pieces closed on the left and open on the right. This differs from the package's open-left, closed-right convention for assigning realized times only at the cutpoints themselves, which have probability zero under the continuous model. The cumulative hazard, event-time distribution, and generated simulations are therefore unchanged.

Dropout is independent censoring conditional on treatment arm. For event time T , `loss_to_fu` is true only when $D < \min(T, \tau)$. Administrative censoring and dropout after an observed event are not counted as loss to follow-up. For example, `prop_loss = 0.05` with `end_of_study = 12` specifies a 5% dropout CDF at 12 months if the time unit is months; it does not force five losses in a 100-subject trial. Equal dropout probabilities in arms with different event hazards need not yield equal observed dropout proportions.

To express a dropout probability q supplied at a different reference time t_0 , use $p = 1 - (1 - q)^{\tau/t_0}$ at `end_of_study` to preserve the same exponential dropout hazard. Treatment discontinuation is not separately modeled and should not be treated as loss to follow-up if endpoint collection continues.

This independent exponential mechanism replaces selection of `ceiling(prop_loss * arm size)` subjects followed by censoring uniformly before each selected subject's potential event or administrative time. Positive `prop_loss` values therefore change seeded results and design operating characteristics relative to the previous mechanism. Simulations with `prop_loss = 0` are unchanged.

Value

A data frame with one row per subject and columns:

- `time`: Numeric event or censoring time.
- `treatment`: Numeric treatment indicator, coded 1 for the treatment arm and 0 for the control arm. Single-arm designs have `treatment = 1` for every subject.
- `event`: Numeric event indicator, coded 1 for an event and 0 for right-censoring.
- `enrollment`: Numeric time of subject enrollment relative to first patient in. The package treats enrollment and randomization as occurring at the same time.
- `id`: Integer subject identifier.
- `loss_to_fu`: Logical indicator that dropout occurred before both the event and the administrative follow-up horizon.

sim_trials

*Estimate operating characteristics by trial simulation***Description**

Repeats `survival_adapt()` under fixed design and data-generating assumptions, returning trial-level results from which operating characteristics can be estimated.

Usage

```
sim_trials(
  hazard_treatment,
  hazard_control = NULL,
  cutpoints = NULL,
  N_total,
  lambda = 0.3,
  lambda_time = NULL,
  interim_look = NULL,
  end_of_study,
  prior_surv = c(0.1, 0.1),
  prior_bin = c(1, 1),
  bin_method = "mc",
  block = 2,
  rand_ratio = c(control = 1, treatment = 1),
  prop_loss = 0,
  alternative = "greater",
  h0 = 0,
  Fn = 0.05,
  Sn = 0.9,
  prob_ha = 0.95,
  N_impute = 500,
  N_mcmc = 1000,
  mc_conf_level = 0.95,
  N_trials = 10,
  method = "logrank",
  imputed_final = FALSE,
  empty_interval = c("prior", "propagate", "error"),
  return_trace = FALSE,
  ncores = 1L,
  backend = c("auto", "fork", "psock", "sequential"),
  seed = NULL,
  binary_imputation = c("event-time", "bernoulli"),
  prior_surv_final = prior_surv,
  generation_cutpoints = cutpoints,
  Qn = 1,
  rmst_tau = end_of_study
)
```

Arguments

<code>hazard_treatment</code>	A required numeric vector of finite, non-negative event rates for the treatment arm. Supply one rate per interval defined by <code>generation_cutpoints</code> ; a single value specifies a constant event rate.
<code>hazard_control</code>	NULL (the default) for a single-arm trial, or a numeric vector of finite, non-negative event rates for the control arm in a two-arm trial. It must contain one rate per interval defined by <code>generation_cutpoints</code> .
<code>cutpoints</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior follow-up times defining the piecewise-exponential model used for interim posterior estimation, predictive imputation, and final analysis. The number of interval-specific prior columns must be one greater than the number of cutpoints. NULL specifies a constant-hazard analysis model.
<code>N_total</code>	A required positive integer giving the maximum total sample size.
<code>lambda</code>	A numeric vector of finite, positive enrollment rates per unit of calendar time. Supply one rate for each interval defined by <code>lambda_time</code> . The default is 0.3. See enrollment() for the continuous-time enrollment model and time origin.
<code>lambda_time</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing calendar times at which the enrollment rate changes. Time zero is implicit, and <code>length(lambda)</code> must equal <code>length(lambda_time) + 1</code> .
<code>interim_look</code>	NULL (the default) for no interim analyses, or a strictly increasing positive integer vector giving the cumulative sample size at each interim look. Do not include the maximum sample size. For two-arm designs, each interim look must be at least the (largest) block size (see <code>block</code>), ensuring both treatment groups are present at every interim analysis; a smaller look could enroll subjects from one treatment group only, leaving the interim posterior undefined for the missing group.
<code>end_of_study</code>	A required finite, positive numeric value giving the planned subject-level follow-up time. It must be greater than the final value in both <code>cutpoints</code> and <code>generation_cutpoints</code> , when supplied, and use the same time unit.
<code>prior_surv</code>	A numeric vector, matrix, or named list specifying the Gamma prior for the piecewise-exponential hazards used to generate outcomes during interim prediction. A length-two vector supplies shape and rate and applies the same prior to every arm and interval. A 2 by <code>length(cutpoints) + 1</code> matrix supplies interval-specific values shared by all arms, with shapes in row 1 and rates in row 2. For independent arm-specific priors, supply a list named <code>control</code> and <code>treatment</code> in a two-arm design, or <code>treatment</code> in a single-arm design. Each list element may be a length-two vector or an interval-specific matrix. Both arms must be supplied; no values are borrowed or filled from the other arm. Rates must use the same time unit as event times, exposure, and cutpoints. The default is <code>c(0.1, 0.1)</code> .
<code>prior_bin</code>	A length-two numeric vector of finite, positive shape parameters <code>c(a, b)</code> for the Beta(<code>a</code> , <code>b</code>) event-probability prior used when <code>method = "bayes-bin"</code> . The same prior is applied to both arms. The default is <code>c(1, 1)</code> , a uniform prior.

bin_method	A single character string selecting how to calculate the posterior probability for method = "bayes-bin". It must be one of "mc" (Monte Carlo sampling), "normal" (normal approximation), or "quadrature" (numerical integration). The default is "mc". The normal approximation can be inaccurate with sparse events or non-events and posterior event probabilities near 0 or 1. It can change whether prob_h0 is exceeded. Increasing N_mcmc does not improve this approximation; use "quadrature" or sufficiently precise "mc" instead.
block	A positive integer vector of permitted randomization block sizes. Every value must be a multiple of sum(rand_ratio). The default is 2 and the argument is ignored for a single-arm trial.
rand_ratio	A length-two positive integer vector giving the control to treatment randomization ratio. The default is c(control = 1, treatment = 1). Name the values control and treatment; either supplied order is accepted and matched by name. A legacy unnamed vector remains accepted in c(control, treatment) order. Unequal unnamed values produce a warning because names may be required in a future major release. See randomization() for more details.
prop_loss	A numeric vector containing one or two probabilities in $[0, 1)$. Each value is the dropout-time CDF at end_of_study: $P(D \leq \tau) = p$, where τ is the planned follow-up duration per subject. Independently of event time and enrollment, each subject's dropout time D is exponentially distributed with rate $-\log(1 - p)/\tau$. The observed time is the minimum of event time, dropout time, and end_of_study; an event occurring before dropout is retained. Thus, prop_loss is not the expected proportion actually censored by dropout: that proportion can be lower because events occur first, and the realized number of dropouts varies between trials. A single value applies the same dropout distribution to every arm. For a two-arm design, supply a length-two vector named control and treatment for arm-specific probabilities; supplied order does not matter. Single-arm designs require one probability. The default 0 sets dropout time to infinity without drawing random numbers. A value of 1 is rejected because it requires an infinite exponential rate.
alternative	A single character string specifying the alternative hypothesis. It must be one of "greater" (the default), "less", or "two.sided". One-sided alternatives ("greater" and "less") are supported for method = "bayes-surv" and method = "bayes-bin". All three options are supported for method = "logrank", method = "cox", method = "rmst", method = "riskdiff-wald", and method = "riskdiff-fm". For an adverse event, benefit is in the "greater" direction for RMST (longer event-free time) and the "less" direction for the other methods (lower hazard or event probability).
h0	A single finite numeric value specifying the null hypothesis or margin. The default is 0. For Bayesian analyses, h0 must lie in $[0, 1]$ for a single-arm design and $[-1, 1]$ for a two-arm design. • When method = "bayes-surv", h0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$. In a single-arm design, h0 is the external benchmark event probability, often referred to as a performance goal (PG) or objective performance criterion (OPC). • When method = "bayes-bin", h0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$ for a two-arm design, or the null event probability for a single-arm design.

	• When <code>method = "cox"</code>, h_0 is the null log hazard ratio for treatment versus control. Use $h_0 = 0$ for the usual hazard ratio of 1 null, or $h_0 = \log(\text{margin})$ for a non-inferiority margin specified as a hazard ratio. A Cox non-inferiority test should usually use <code>alternative = "less"</code>. • When <code>method = "rmst"</code>, h_0 is the null treatment-control RMST difference in time units and must lie in $[-\text{rmst_tau}, \text{rmst_tau}]$. For non-inferiority allowing a loss of m time units, use $h_0 = -m$ and <code>alternative = "greater"</code>. • When <code>method = "riskdiff-wald"</code> or <code>method = "riskdiff-fm"</code>, h_0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$ and must lie in $[-1, 1]$. • When <code>method = "logrank"</code>, only $h_0 = 0$ is supported; this denotes the usual equal-survival null. Nonzero values are rejected because the standard log-rank statistic does not implement a nonzero effect margin.
<code>Fn</code>	NULL, or a numeric vector of probabilities in $[0, 1]$. Each value is the predictive-probability threshold to stop at the i -th look early for futility. If there are no interim looks (i.e. <code>interim_look = NULL</code>), then <code>Fn</code> is not used in the simulations or analysis. Set <code>Fn = 0</code> to disable futility monitoring; <code>Fn = NULL</code> has the same effect. Supply either one value, which is repeated at every interim look, or exactly one value per <code>interim_look</code> . Other lengths are rejected rather than recycled. The default is <code>0.05</code> .
<code>Sn</code>	A numeric vector of probabilities in $[0, 1]$. Each value is the predictive-probability threshold to stop accrual at the i -th look for expected success. If there are no interim looks (i.e. <code>interim_look = NULL</code>), then <code>Sn</code> is not used in the simulations or analysis. Supply either one value, which is repeated at every interim look, or exactly one value per <code>interim_look</code> . Other lengths are rejected rather than recycled. The default is <code>0.9</code> .
<code>prob_ha</code>	A single numeric probability in $[0, 1]$ defining success in each completed-data analysis. For Bayesian methods this is compared with the posterior probability of the alternative; for frequentist methods it is compared with $1 - P$. The default is <code>0.95</code> .
<code>N_impute</code>	A positive integer giving the number of predictive imputations used at each interim look and, when requested, for final multiple imputation. The default is <code>500</code> . An imputed Cox, RMST, or risk-difference final analysis requires at least two.
<code>N_mcmc</code>	A positive integer giving the number of posterior draws used within each method = "bayes-surv" and by method = "bayes-bin" when <code>bin_method = "mc"</code> . The default is <code>1000</code> .
<code>mc_conf_level</code>	A single numeric probability strictly between <code>0.5</code> and <code>1</code> , giving the confidence level for one-sided exact binomial bounds reported as diagnostics of finite Monte Carlo uncertainty. The bounds do not alter completed-data success classifications or interim decisions, which use strict point-estimate comparisons with <code>prob_ha</code> , <code>Qn</code> , <code>Sn</code> , and <code>Fn</code> . The default is <code>0.95</code> .
<code>N_trials</code>	A positive integer giving the number of independent trials to simulate. The default is <code>10</code> .
<code>method</code>	A single character string specifying the completed-data and final analysis. Available choices are a log-rank (method = "logrank") test, Cox proportional hazards regression model Wald test (method = "cox"), a restricted mean survival

time difference Wald test (method = "rmst"), a fully-Bayesian piecewise-exponential analysis (method = "bayes-surv"), a Bayesian beta-binomial analysis of complete binary outcomes (method = "bayes-bin"), a frequentist risk-difference Wald test (method = "riskdiff-wald"), or a Farrington-Manning score test (method = "riskdiff-fm") of complete binary outcomes. The deprecated method = "riskdiff" is accepted as an alias for "riskdiff-wald" with a warning. The default is "logrank". See Details.

- imputed_final** A single logical value indicating whether the final analysis should be based on imputed outcomes for subjects who were LTFU (i.e. right-censored with time less than end_of_study). The default is FALSE, which uses the observed-data analysis. If no outcomes require imputation, the selected complete-data test is used directly with either flag. With missing outcomes and method = "cox", "rmst", or "riskdiff-wald", setting this to TRUE pools the scalar treatment effects and variances using Rubin's rules; this requires N_impute >= 2 and positive total variance. Genuine final imputation is unsupported for method = "riskdiff-fm" because no validated FM pooling rule is implemented. Simulations combining FM and imputed_final = TRUE therefore require prop_loss = 0 in both arms. Imputed final analyses remain unavailable for method = "logrank".
- empty_interval** A single character string specifying how to handle empty piecewise-exponential intervals when updating Gamma hazard models for predictive imputation and Bayesian survival analysis. An empty interval is an interval with no exposed subjects in a treatment arm at the analysis time. "prior" (the default) leaves the interval at zero exposure time and zero events, so its posterior is driven only by its assigned survival prior. "propagate" is a legacy heuristic that copies exposure time and event counts from the nearest non-empty interval in the same treatment arm and emits a warning. "error" stops when any empty interval is found.
- return_trace** A single logical value indicating whether to retain the compact interim decision trace from every simulated trial. The default, FALSE, preserves the compact output. When TRUE, the returned list also contains a traces data frame with a trial column linking each trace row to the corresponding original simulated trial.
- ncores** A positive integer giving the maximum number of processor cores to use. The default is 1L, which runs trials sequentially. The number actually used cannot exceed N_trials; with backend = "auto", at least two trials are required per core to justify the parallel-processing overhead.
- backend** A single character string selecting the computational method. "auto" (the default) runs sequentially when ncores = 1 or fewer than four trials are requested; otherwise it uses fork-based parallelization on Unix-like systems and a PSOCK cluster on Windows. "fork", "psock", and "sequential" select a method explicitly. Forking is unavailable on Windows.
- seed** NULL (the default), or a single integer between 0 and .Machine\$integer.max. A supplied seed gives reproducible simulations, including when trials are run in parallel, and leaves the pre-existing random-number state unchanged.
- binary_imputation** A single character string selecting the predictive imputation approach for method = "bayes-bin", method = "riskdiff-wald", or method = "riskdiff-fm". "event-time"

(the default) draws a conditional piecewise-exponential event time and reduces it to event status at `end_of_study`. "bernoulli" draws the endpoint status directly from its conditional event probability. This argument is ignored for time-to-event analysis methods.

prior_surv_final

A numeric vector, matrix, or named list specifying the Gamma prior used for final-stage piecewise-exponential imputation and, for `method = "bayes-surv"`, both the analysis of each hypothetical completed trial at interim looks and the actual final analysis. It accepts the same shared or arm-specific forms as `prior_surv` and defaults to `prior_surv`. An informative `prior_surv` can therefore predict outstanding outcomes while a weak `prior_surv_final` defines the Bayesian survival success criterion. To use different priors for these roles, supply `prior_surv_final` explicitly; an informative predictive prior is otherwise also the default analysis prior. See **Predictive and analysis priors** below.

generation_cutpoints

NULL, or a numeric vector of finite, positive, strictly increasing interior follow-up times defining the piecewise-exponential model used to generate event times. `hazard_treatment` and `hazard_control` must each have one value per resulting interval. Defaults to cutpoints, preserving the historical behavior in which generation and analysis used one partition.

Qn

A numeric vector of probabilities in $[0, 1]$. Each value is the upper predictive-probability threshold for declaring immediate trial success at the i -th look. If there are no interim looks (i.e. `interim_look = NULL`), then `Qn` is not used in the simulations or analysis. Supply either one value, which is repeated at every interim look, or exactly one value per `interim_look`; other lengths are rejected. `Qn` must be greater than or equal to `Sn` at every look. The default, 1, disables immediate-success stopping.

rmst_tau

A single finite positive restriction time for `method = "rmst"`, in the same units as `end_of_study`. Defaults to `end_of_study` and must not exceed it. Prespecify the same horizon for all looks, imputations, and simulations. It may precede analysis cutpoints and does not shorten the planned follow-up or imputation horizon. Ignored for other methods.

Details

This function is a wrapper for `survival_adapt()` that repeatedly simulates independent trials under the same design parameters and assumed treatment effect.

To use multiple cores (where available), the argument `ncores` can be increased from the default of 1. The default `backend = "auto"` stays sequential for fewer than four trials and otherwise uses no more than one core per two trials. This avoids parallel-processing overhead for small simulation studies. On Unix-like systems parallel trials use forked R processes; on Windows they use PSOCK processes. Set `backend` explicitly when a particular computational method is required.

Errors raised by an individual `survival_adapt()` call are isolated so other trials can finish. Failed trials are excluded from `sims`, recorded in `failures` with their trial number, error class, and message, and reported together in one warning. If every requested trial fails, `sim_trials()` stops and attaches the same failure table to the error as `failures`. With a supplied seed, the original call and failed trial number reproduce the same per-trial random-number stream.

With a supplied seed, each trial receives an independent random-number stream. The resulting trial-level simulations are identical whether they are run sequentially or with a supported parallel method, and the pre-existing R random-number state is restored afterward. With `seed = NULL`, the current random-number state is used and advanced.

Value

A list containing `sims`, a data frame with one row per successfully simulated trial; `failures`, a data frame with columns `trial`, `error_class`, and `message`; and `call`. When `return_trace = TRUE`, the list also contains `traces`, a data frame with one row per completed interim look and a trial identifier. Per-trial calendar-time metrics are always retained in `sims`; traces additionally retain calendar time and active follow-up at each look. See [survival_adapt\(\)](#) for details of the summary and trace columns, and [summarise_calendar_time\(\)](#) for wide operating-characteristic tables. The returned object also retains the evaluated `decision_design` and resolved `prior_design` attributes from [survival_adapt\(\)](#). An `rng_metadata` attribute records the random-number generator, computational method, and seed policy. A `parallel_metadata` attribute records the requested and actual computational method and number of cores. An `arguments` attribute contains a named list of all evaluated argument values, including defaults. Its `prop_loss` element contains a named value for every simulated arm, and its `rand_ratio` element is stored in `control`, treatment order for two-arm designs. Its `cutpoints` and `generation_cutpoints` elements retain the analysis and data-generation partitions, respectively. For `method = "bayes-bin"`, it also retains the imputation priors (`prior_surv` and `prior_surv_final`), completed-data analysis prior (`prior_bin`), and imputation horizon (`end_of_study`). The attribute can be saved with [saveRDS\(\)](#) and supplied to a later call with `do.call(sim_trials, attr(result, "arguments"))`.

Predictive and analysis priors

For `method = "bayes-surv"`, `prior_surv_final` is used **during interim calculations as well as at the actual final analysis**. The two arguments specify different roles, not simply different calendar stages:

Calculation	Gamma prior used
At interim, generate outstanding outcomes for enrolled and future participants	<code>prior_surv</code>
At interim, test each hypothetical completed trial at the current or maximum sample size	<code>prior_surv_final</code>
At final analysis, impute missing outcomes if <code>imputed_final = TRUE</code>	<code>prior_surv_final</code>
Analyze the actual final trial data	<code>prior_surv_final</code>

Within one interim predictive replicate, first update `prior_surv` with the observed events and exposure, draw hazards, and generate outstanding outcomes. Then start a fresh analysis posterior using `prior_surv_final` and the completed dataset's events and exposure. Compare its posterior probability of the alternative with `prob_ha`. The proportion of replicates that pass is the predictive probability used by Q_n , S_n , and F_n .

To incorporate external evidence in prediction while using a weak analysis prior, explicitly supply an informative `prior_surv` and the chosen weak `prior_surv_final`. **Omitting `prior_surv_final` uses `prior_surv` for both roles; the package does not automatically weaken the analysis prior.** The predictive prior can still affect the selected sample size and stopping decision, so calibrate the design using both prespecified priors.

This table describes Bayesian survival analysis. For method = "bayes-bin", completed-data success tests at interim and final use prior_bin; prior_surv_final governs only optional final imputation. Frequentist completed-data tests use no analysis prior. `evaluate_interim()` performs the two interim calculations; use the same prior arguments as in the simulated design.

Examples

```
hc <- prop_to_haz(c(0.20, 0.30), 12, 36)
ht <- prop_to_haz(c(0.05, 0.15), 12, 36)

out <- sim_trials(
  hazard_treatment = ht,
  hazard_control = hc,
  cutpoints = 12,
  N_total = 600,
  lambda = 20,
  lambda_time = NULL,
  interim_look = c(400, 500),
  end_of_study = 36,
  prior_surv = c(0.1, 0.1),
  block = 2,
  rand_ratio = c(control = 1, treatment = 1),
  prop_loss = 0.30,
  alternative = "two.sided",
  h0 = 0,
  Fn = 0.05,
  Sn = 0.9,
  prob_ha = 0.975,
  N_impute = 5,
  N_mcmc = 5,
  method = "logrank",
  N_trials = 2,
  ncores = 1,
  backend = "auto",
  seed = 123)
```

summarise_calendar_time

Summarize operating characteristics on the calendar-time scale

Description

Summarizes trial duration, follow-up burden, and (when retained) interim-look timing without adding any new simulation arguments. Calendar time is measured from the first patient's enrollment at time zero. analysis_ready_time is the time at which the last enrolled subject's observed event or censoring becomes available; it does not include external data-cleaning or database-lock delays.

Pass a complete result from `sim_trials()` to retain requested, analyzed, and failed simulation counts. Interim timing requires simulations run with `return_trace = TRUE`. A simulation data frame can also be supplied, but only the trial-duration table can then be calculated.

Usage

```
summarise_calendar_time(data)
```

Arguments

data A required complete result returned by `sim_trials()`, a simulation data.frame, or a list of either form. Named list elements identify scenarios. Existing scenario columns and grouping variables are preserved.

Value

An object of class `goldilocks_calendar_summary`, containing two wide data frames:

- **trial_duration**: one row per scenario and stopping reason, plus an overall row. It reports simulation denominators, stopping counts, sample size, accrual-stop time, analysis-ready time, planned completion time, total person-time under follow-up, and peak concurrent follow-up. Newer results also report the time at which the terminal decision was made.
- **interim_timing**: one row per scenario and interim look. It reports how often the look was reached, its calendar time, and the number of subjects actively under follow-up. This table has zero rows when traces are not available.

Continuous quantities are reported as means, Monte Carlo standard errors, and the 10th, 50th, and 90th percentiles. Percentages use the requested number of simulations as their denominator, so excluded failures remain visible.

summarise_sims	<i>Estimate operating characteristics from trial simulations</i>
----------------	--

Description

Estimates success, stopping, and sample-size operating characteristics from a collection of simulated trials and quantifies their Monte Carlo uncertainty.

Usage

```
summarise_sims(data, max_mcse = NULL)
```

Arguments

data A required complete result returned by `sim_trials()`, a simulation data.frame, or a list of either form. Named list elements identify scenarios. Existing scenario columns and grouping variables are preserved.

max_mcse NULL (the default), or a named numeric vector of finite, positive values giving the largest acceptable Monte Carlo standard error for selected estimands. Supported names are `power`, `stop_immediate_success`, `stop_success`, `stop_any_success`, `stop_futility`, `stop_max_N`, `mean_N`, `stop_and_fail`, and `failure_rate`. A warning identifies every scenario and estimand whose Monte Carlo standard error exceeds its target.

Value

A data frame reporting the operating characteristics, including the power (which will be equal to the type I error in the null case); the proportion of trials that declared immediate success, stopped accrual for expected success, stopped for futility, or went to the maximum sample size. `stop_success` retains its historical meaning of stopping accrual for expected success; `stop_any_success` combines both success-stopping decisions. The average stopping sample size (and standard deviation) are also recorded. The proportion of trials that stopped accrual for expected success, yet went on to fail, is also reported. Each probability and mean is accompanied by its Monte Carlo standard error and 95% Monte Carlo confidence limits, with columns ending in `_mcse`, `_mc_lower`, and `_mc_upper`. Probability intervals use the Wilson method; the mean sample-size interval uses a *t* distribution.

These intervals describe uncertainty from using a finite number of simulated trials under fixed design and data-generating assumptions. They are **not clinical confidence intervals**, treatment-effect intervals, or measures of model uncertainty.

The output always reports `n_used`, the number of successfully analyzed simulations used by the operating-characteristic estimands. When complete `sim_trials()` results are supplied, it also reports requested, analyzed, and failed counts, the failure rate, computational method, and seed. For a raw simulation data frame, requested and failed counts are unknown and are reported as NA. Details of the call, random-number generation, parallel computation, timing, failures, and design assumptions are retained in the `simulation_metadata` attribute.

`summarise_trial_trace` *Summarize an interim decision path*

Description

Creates a compact one-row summary of the final interim look and stopping decision. Pass a `goldilocks_trial` object to include final analysis information, or pass its trace element to summarize the path only.

Usage

```
summarise_trial_trace(x)
```

Arguments

`x` A required `goldilocks_trial` result from `survival_adapt()`, a `goldilocks_interim` result from `evaluate_interim()`, or an interim trace data frame.

Value

A one-row data frame.

survival_adapt

*Simulate and analyze one Goldilocks adaptive trial***Description**

Simulates one single-arm or randomized two-arm trial under a Goldilocks sample-size design. At each planned interim look, posterior predictive probabilities determine whether to declare immediate success, stop accrual for expected success, stop for futility, or continue toward the maximum sample size.

Usage

```
survival_adapt(
  hazard_treatment,
  hazard_control = NULL,
  cutpoints = NULL,
  N_total,
  lambda = 0.3,
  lambda_time = NULL,
  interim_look = NULL,
  end_of_study,
  prior_surv = c(0.1, 0.1),
  prior_bin = c(1, 1),
  bin_method = "mc",
  block = 2,
  rand_ratio = c(control = 1, treatment = 1),
  prop_loss = 0,
  alternative = "greater",
  h0 = 0,
  Fn = 0.05,
  Sn = 0.9,
  prob_ha = 0.95,
  N_impute = 500,
  N_mcmc = 1000,
  mc_conf_level = 0.95,
  empty_interval = c("prior", "propagate", "error"),
  method = "logrank",
  imputed_final = FALSE,
  return_trace = FALSE,
  binary_imputation = c("event-time", "bernoulli"),
  prior_surv_final = prior_surv,
  generation_cutpoints = cutpoints,
  Qn = 1,
  rmst_tau = end_of_study
)
```

Arguments

<code>hazard_treatment</code>	A required numeric vector of finite, non-negative event rates for the treatment arm. Supply one rate per interval defined by <code>generation_cutpoints</code> ; a single value specifies a constant event rate.
<code>hazard_control</code>	NULL (the default) for a single-arm trial, or a numeric vector of finite, non-negative event rates for the control arm in a two-arm trial. It must contain one rate per interval defined by <code>generation_cutpoints</code> .
<code>cutpoints</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing interior follow-up times defining the piecewise-exponential model used for interim posterior estimation, predictive imputation, and final analysis. The number of interval-specific prior columns must be one greater than the number of cutpoints. NULL specifies a constant-hazard analysis model.
<code>N_total</code>	A required positive integer giving the maximum total sample size.
<code>lambda</code>	A numeric vector of finite, positive enrollment rates per unit of calendar time. Supply one rate for each interval defined by <code>lambda_time</code> . The default is 0.3. See <code>enrollment()</code> for the continuous-time enrollment model and time origin.
<code>lambda_time</code>	NULL (the default), or a numeric vector of finite, positive, strictly increasing calendar times at which the enrollment rate changes. Time zero is implicit, and <code>length(lambda)</code> must equal <code>length(lambda_time) + 1</code> .
<code>interim_look</code>	NULL (the default) for no interim analyses, or a strictly increasing positive integer vector giving the cumulative sample size at each interim look. Do not include the maximum sample size. For two-arm designs, each interim look must be at least the (largest) block size (see <code>block</code>), ensuring both treatment groups are present at every interim analysis; a smaller look could enroll subjects from one treatment group only, leaving the interim posterior undefined for the missing group.
<code>end_of_study</code>	A required finite, positive numeric value giving the planned subject-level follow-up time. It must be greater than the final value in both <code>cutpoints</code> and <code>generation_cutpoints</code> , when supplied, and use the same time unit.
<code>prior_surv</code>	A numeric vector, matrix, or named list specifying the Gamma prior for the piecewise-exponential hazards used to generate outcomes during interim prediction. A length-two vector supplies shape and rate and applies the same prior to every arm and interval. A 2 by <code>length(cutpoints) + 1</code> matrix supplies interval-specific values shared by all arms, with shapes in row 1 and rates in row 2. For independent arm-specific priors, supply a list named <code>control</code> and <code>treatment</code> in a two-arm design, or <code>treatment</code> in a single-arm design. Each list element may be a length-two vector or an interval-specific matrix. Both arms must be supplied; no values are borrowed or filled from the other arm. Rates must use the same time unit as event times, exposure, and cutpoints. The default is <code>c(0.1, 0.1)</code> .
<code>prior_bin</code>	A length-two numeric vector of finite, positive shape parameters <code>c(a, b)</code> for the Beta(a, b) event-probability prior used when <code>method = "bayes-bin"</code> . The same prior is applied to both arms. The default is <code>c(1, 1)</code> , a uniform prior.

bin_method	A single character string selecting how to calculate the posterior probability for method = "bayes-bin". It must be one of "mc" (Monte Carlo sampling), "normal" (normal approximation), or "quadrature" (numerical integration). The default is "mc". The normal approximation can be inaccurate with sparse events or non-events and posterior event probabilities near 0 or 1. It can change whether prob_h0 is exceeded. Increasing N_mcmc does not improve this approximation; use "quadrature" or sufficiently precise "mc" instead.
block	A positive integer vector of permitted randomization block sizes. Every value must be a multiple of sum(rand_ratio). The default is 2 and the argument is ignored for a single-arm trial.
rand_ratio	A length-two positive integer vector giving the control to treatment randomization ratio. The default is c(control = 1, treatment = 1). Name the values control and treatment; either supplied order is accepted and matched by name. A legacy unnamed vector remains accepted in c(control, treatment) order. Unequal unnamed values produce a warning because names may be required in a future major release. See randomization() for more details.
prop_loss	A numeric vector containing one or two probabilities in $[0, 1)$. Each value is the dropout-time CDF at end_of_study: $P(D \leq \tau) = p$, where τ is the planned follow-up duration per subject. Independently of event time and enrollment, each subject's dropout time D is exponentially distributed with rate $-\log(1 - p)/\tau$. The observed time is the minimum of event time, dropout time, and end_of_study; an event occurring before dropout is retained. Thus, prop_loss is not the expected proportion actually censored by dropout: that proportion can be lower because events occur first, and the realized number of dropouts varies between trials. A single value applies the same dropout distribution to every arm. For a two-arm design, supply a length-two vector named control and treatment for arm-specific probabilities; supplied order does not matter. Single-arm designs require one probability. The default 0 sets dropout time to infinity without drawing random numbers. A value of 1 is rejected because it requires an infinite exponential rate.
alternative	A single character string specifying the alternative hypothesis. It must be one of "greater" (the default), "less", or "two.sided". One-sided alternatives ("greater" and "less") are supported for method = "bayes-surv" and method = "bayes-bin". All three options are supported for method = "logrank", method = "cox", method = "rmst", method = "riskdiff-wald", and method = "riskdiff-fm". For an adverse event, benefit is in the "greater" direction for RMST (longer event-free time) and the "less" direction for the other methods (lower hazard or event probability).
h0	A single finite numeric value specifying the null hypothesis or margin. The default is 0. For Bayesian analyses, h0 must lie in $[0, 1]$ for a single-arm design and $[-1, 1]$ for a two-arm design. • When method = "bayes-surv", h0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$. In a single-arm design, h0 is the external benchmark event probability, often referred to as a performance goal (PG) or objective performance criterion (OPC). • When method = "bayes-bin", h0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$ for a two-arm design, or the null event probability for a single-arm design.

	• When <code>method = "cox"</code>, h_0 is the null log hazard ratio for treatment versus control. Use $h_0 = 0$ for the usual hazard ratio of 1 null, or $h_0 = \log(\text{margin})$ for a non-inferiority margin specified as a hazard ratio. A Cox non-inferiority test should usually use <code>alternative = "less"</code>. • When <code>method = "rmst"</code>, h_0 is the null treatment-control RMST difference in time units and must lie in $[-\text{rmst_tau}, \text{rmst_tau}]$. For non-inferiority allowing a loss of m time units, use $h_0 = -m$ and <code>alternative = "greater"</code>. • When <code>method = "riskdiff-wald"</code> or <code>method = "riskdiff-fm"</code>, h_0 is the null value of $p_{\text{treatment}} - p_{\text{control}}$ and must lie in $[-1, 1]$. • When <code>method = "logrank"</code>, only $h_0 = 0$ is supported; this denotes the usual equal-survival null. Nonzero values are rejected because the standard log-rank statistic does not implement a nonzero effect margin.
<code>Fn</code>	NULL, or a numeric vector of probabilities in $[0, 1]$. Each value is the predictive-probability threshold to stop at the i -th look early for futility. If there are no interim looks (i.e. <code>interim_look = NULL</code>), then <code>Fn</code> is not used in the simulations or analysis. Set <code>Fn = 0</code> to disable futility monitoring; <code>Fn = NULL</code> has the same effect. Supply either one value, which is repeated at every interim look, or exactly one value per <code>interim_look</code> . Other lengths are rejected rather than recycled. The default is <code>0.05</code> .
<code>Sn</code>	A numeric vector of probabilities in $[0, 1]$. Each value is the predictive-probability threshold to stop accrual at the i -th look for expected success. If there are no interim looks (i.e. <code>interim_look = NULL</code>), then <code>Sn</code> is not used in the simulations or analysis. Supply either one value, which is repeated at every interim look, or exactly one value per <code>interim_look</code> . Other lengths are rejected rather than recycled. The default is <code>0.9</code> .
<code>prob_ha</code>	A single numeric probability in $[0, 1]$ defining success in each completed-data analysis. For Bayesian methods this is compared with the posterior probability of the alternative; for frequentist methods it is compared with $1 - P$. The default is <code>0.95</code> .
<code>N_impute</code>	A positive integer giving the number of predictive imputations used at each interim look and, when requested, for final multiple imputation. The default is <code>500</code> . An imputed Cox, RMST, or risk-difference final analysis requires at least two.
<code>N_mcmc</code>	A positive integer giving the number of posterior draws used within each method = <code>"bayes-surv"</code> and by method = <code>"bayes-bin"</code> when <code>bin_method = "mc"</code> . The default is <code>1000</code> .
<code>mc_conf_level</code>	A single numeric probability strictly between <code>0.5</code> and <code>1</code> , giving the confidence level for one-sided exact binomial bounds reported as diagnostics of finite Monte Carlo uncertainty. The bounds do not alter completed-data success classifications or interim decisions, which use strict point-estimate comparisons with <code>prob_ha</code> , <code>Qn</code> , <code>Sn</code> , and <code>Fn</code> . The default is <code>0.95</code> .
<code>empty_interval</code>	A single character string specifying how to handle empty piecewise-exponential intervals when updating Gamma hazard models for predictive imputation and Bayesian survival analysis. An empty interval is an interval with no exposed subjects in a treatment arm at the analysis time. <code>"prior"</code> (the default) leaves the interval at zero exposure time and zero events, so its posterior is driven only

by its assigned survival prior. "propagate" is a legacy heuristic that copies exposure time and event counts from the nearest non-empty interval in the same treatment arm and emits a warning. "error" stops when any empty interval is found.

method	A single character string specifying the completed-data and final analysis. Available choices are a log-rank (method = "logrank") test, Cox proportional hazards regression model Wald test (method = "cox"), a restricted mean survival time difference Wald test (method = "rmst"), a fully-Bayesian piecewise-exponential analysis (method = "bayes-surv"), a Bayesian beta-binomial analysis of complete binary outcomes (method = "bayes-bin"), a frequentist risk-difference Wald test (method = "riskdiff-wald"), or a Farrington-Manning score test (method = "riskdiff-fm") of complete binary outcomes. The deprecated method = "riskdiff" is accepted as an alias for "riskdiff-wald" with a warning. The default is "logrank". See Details.
imputed_final	A single logical value indicating whether the final analysis should be based on imputed outcomes for subjects who were LTFU (i.e. right-censored with time less than end_of_study). The default is FALSE, which uses the observed-data analysis. If no outcomes require imputation, the selected complete-data test is used directly with either flag. With missing outcomes and method = "cox", "rmst", or "riskdiff-wald", setting this to TRUE pools the scalar treatment effects and variances using Rubin's rules; this requires N_impute >= 2 and positive total variance. Genuine final imputation is unsupported for method = "riskdiff-fm" because no validated FM pooling rule is implemented. Simulations combining FM and imputed_final = TRUE therefore require prop_loss = 0 in both arms. Imputed final analyses remain unavailable for method = "logrank".
return_trace	A single logical value indicating whether the interim decision path should be returned in addition to the usual final summary. The default, FALSE, returns the historical one-row data frame. When TRUE, the result is a goldilocks_trial object with summary, trace, prior and posterior diagnostics, and call elements.
binary_imputation	A single character string selecting the predictive imputation approach for method = "bayes-bin", method = "riskdiff-wald", or method = "riskdiff-fm". "event-time" (the default) draws a conditional piecewise-exponential event time and reduces it to event status at end_of_study. "bernoulli" draws the endpoint status directly from its conditional event probability. This argument is ignored for time-to-event analysis methods.
prior_surv_final	A numeric vector, matrix, or named list specifying the Gamma prior used for final-stage piecewise-exponential imputation and, for method = "bayes-surv", both the analysis of each hypothetical completed trial at interim looks and the actual final analysis. It accepts the same shared or arm-specific forms as prior_surv and defaults to prior_surv. An informative prior_surv can therefore predict outstanding outcomes while a weak prior_surv_final defines the Bayesian survival success criterion. To use different priors for these roles, supply prior_surv_final explicitly; an informative predictive prior is otherwise also the default analysis prior. See Predictive and analysis priors below.
generation_cutpoints	NULL, or a numeric vector of finite, positive, strictly increasing interior follow-

	up times defining the piecewise-exponential model used to generate event times. hazard_treatment and hazard_control must each have one value per resulting interval. Defaults to cutpoints, preserving the historical behavior in which generation and analysis used one partition.
Qn	A numeric vector of probabilities in $[0, 1]$. Each value is the upper predictive-probability threshold for declaring immediate trial success at the i -th look. If there are no interim looks (i.e. interim_look = NULL), then Qn is not used in the simulations or analysis. Supply either one value, which is repeated at every interim look, or exactly one value per interim_look; other lengths are rejected. Qn must be greater than or equal to Sn at every look. The default, 1, disables immediate-success stopping.
rmst_tau	A single finite positive restriction time for method = "rmst", in the same units as end_of_study. Defaults to end_of_study and must not exceed it. Prespecify the same horizon for all looks, imputations, and simulations. It may precede analysis cutpoints and does not shorten the planned follow-up or imputation horizon. Ignored for other methods.

Details

Implements the Goldilocks design method described in Broglio et al. (2014). At each interim analysis, two probabilities are computed:

1. **The posterior predictive probability of eventual success.** This is calculated as the proportion of imputed datasets at the *current* sample size that satisfy the completed-data success criterion. At each interim analysis this proportion is first compared to the corresponding element of Qn. If it is strictly greater than Qn, the trial stops and declares immediate success. Otherwise, if it is strictly greater than Sn, accrual/enrollment is suspended and the outstanding follow-up is allowed to complete before conducting the pre-specified final analysis.
2. **The posterior predictive probability of success at the maximum sample size.** This is calculated as the proportion of imputed datasets at the *maximum* sample size that satisfy the completed-data success criterion. It is compared to the corresponding element of Fn, and if it is below the threshold, accrual/enrollment is suspended and the trial terminated. Typically this would be a binding decision. If it is not a binding decision, then one should also explore the simulations with $F_n = \emptyset$.

Hence, each interim look has four possible decisions, applied in this order:

1. **Stop and declare immediate success** when $P_{n,l} > Q_l$.
2. **Stop accruing for expected success and follow** when $S_l < P_{n,l} \leq Q_l$.
3. **Stop for futility** when $P_{n_{max},l} < F_l$.
4. **Continue to enroll** new subjects, or if at maximum sample size, proceed to final analysis.

The following completed-data analysis methods are available at interim and final analyses:

- Log-rank test (method = "logrank"). Each (imputed) dataset with both treatment and control arms can be compared using a standard log-rank test. The output is a P -value, and there is no treatment effect reported. The function returns $1 - P$, which is reported in post_prob_ha. Whilst not a posterior probability, it can be contrasted in the same manner. For example, if the success threshold is $P < 0.05$, then one requires $\text{post_prob_ha} > 0.95$. The reason for this

is to enable simple switching between Bayesian and frequentist paradigms for analysis. When `alternative = "less"` or `"greater"`, a one-sided P -value is computed from the log-rank z-statistic.

- Cox proportional hazards regression Wald test (`method = "cox"`). Similar to the log-rank test, a P -value is calculated and $1 - P$ is reported in `post_prob_ha`. When `alternative = "two.sided"`, the standard two-sided Wald P -value is used when $h_0 = 0$. For other values of h_0 , the Wald test is centered on the specified null log hazard ratio. When `alternative = "less"` or `"greater"`, a one-sided P -value is derived from the Wald z-statistic relative to h_0 . The treatment effect (log hazard ratio) is also reported. When `imputed_final = TRUE` and outcomes are missing, the Cox model is fitted to each of at least two imputed datasets. The log hazard ratios and their within-imputation variances are combined using Rubin's rules; the pooled Wald test uses Rubin's large-sample degrees of freedom. When `imputed_final = FALSE`, the existing single Cox model is fitted directly to the observed right-censored data.
- Restricted mean survival time (`method = "rmst"`). Estimates the area under each Kaplan-Meier survival curve from zero through the prespecified `rmst_tau`. The effect is treatment minus control RMST, measured in time units. The Wald test uses the sum of the independent arm-specific Greenwood variances and reports $1 - P$. All three alternatives and nonzero margins are supported. For an adverse event, a positive difference favors treatment. Observed censored subjects remain in the analysis. An arm whose last follow-up precedes `rmst_tau` with positive estimated survival makes the analysis non-estimable; the horizon is never reduced automatically. A curve reaching zero earlier is allowed. The test requires positive total variance, including after Rubin pooling for `imputed_final = TRUE`. This unadjusted two-arm analysis does not assume proportional hazards. Interim prediction and final imputation still depend on the specified piecewise-exponential model. See `vignette("rmst", package = "goldilocks")`.
- Bayesian difference in cumulative event probability (`method = "bayes-surv"`). Each imputed dataset is used to update the conjugate Gamma analysis prior `prior_surv_final`, at both interim looks and the final stage, yielding a posterior distribution for the piecewise exponential rate parameters. In turn, the posterior distribution of the cumulative incidence function ($1 - S(t)$, where $S(t)$ is the survival function) evaluated at time `end_of_study` is calculated. In a single-arm study, inference concerns the treatment-arm event probability. In a two-arm study, the independent arm-specific posteriors define the posterior distribution of the treatment-minus-control difference. The reported posterior probability is determined by `alternative` and h_0 .

For piecewise-exponential analyses, an interim or final dataset may contain intervals with no exposed subjects in one treatment arm, especially when later cutpoints occur after the available follow-up at early looks. The `empty_interval` argument controls this case. The default, `"prior"`, leaves an empty interval prior-driven, making the absence of interval data explicit. The legacy `"propagate"` option borrows sufficient statistics from the nearest non-empty interval within the same treatment arm. It is operationally stable but statistically consequential because adjacent observed data then inform the empty interval's posterior. `"error"` is strict and stops the simulation or analysis when an empty interval is encountered.

- Bayesian beta-binomial analysis (`method = "bayes-bin"`). Each complete or imputed dataset is reduced to binary event outcomes at `end_of_study`. A conjugate $\text{Beta}(a, b)$ prior, specified with `prior_bin`, is updated with the number of events and non-events in each arm. In a single-arm study, inference is based on the posterior event probability. In a two-arm study, inference is based on $p_{\text{treatment}} - p_{\text{control}}$. This posterior probability can be calculated using

Monte Carlo beta draws (`bin_method = "mc"`), a normal approximation (`"normal"`), or numerical quadrature (`"quadrature"`). Like the risk-difference test, this method requires complete binary outcomes: censored subjects must either be followed to `end_of_study`, imputed, or excluded when `imputed_final = FALSE`.

The normal approximation matches posterior moments but can misrepresent tail probabilities when a Beta posterior is skewed, particularly near event-probability boundaries of 0 or 1. This can change success and interim stopping decisions. Use `bin_method = "quadrature"` or sufficiently precise `"mc"` when the approximation is unsuitable; increasing `N_mcmc` does not correct normal-approximation error.

Two equivalent predictive imputation approaches are available through `binary_imputation`. With `"event-time"`, the package samples a future event time conditional on the available event-free follow-up and then records whether it falls by `end_of_study`. With `"bernoulli"`, it calculates the same endpoint probability directly. If T is the observed event-free follow-up, T^* is `end_of_study`, $S(t)$ is the survival function, and $H(t)$ is the cumulative hazard, that probability is

$$\Pr(X = 1 \mid T_{\text{event}} > T) = \frac{S(T) - S(T^*)}{S(T)} = 1 - \exp\{-[H(T^*) - H(T)]\}.$$

A Bernoulli outcome is drawn with this probability. For a subject not yet enrolled, $T = 0$; observed events are retained unchanged. Because no precise event time is generated, the imputed time is set to `end_of_study` and only the binary event status is analyzed. Each imputation still uses a sampled posterior hazard draw, so uncertainty in the piecewise-exponential model is retained.

- Frequentist risk difference (`method = "riskdiff-wald"` or `"riskdiff-fm"`). Each complete or predictively imputed dataset is reduced to binary event outcomes at `end_of_study`. The estimated treatment effect is $p_{\text{treatment}} - p_{\text{control}}$. `"riskdiff-wald"` uses the observed arm risks in an unpooled Wald variance. `"riskdiff-fm"` instead uses maximum likelihood arm risks constrained by the null difference h_0 in a Farrington-Manning score variance. The latter remains defined for common sparse tables, including equal-arm all-zero and all-one outcomes. Both methods report $1 - P$ in `post_prob_ha` and support all three alternatives. Because they require complete binary outcomes, lost-to-follow-up subjects are excluded when `imputed_final = FALSE`. With genuinely missing final outcomes, only `"riskdiff-wald"` supports `imputed_final = TRUE`: estimates and variances from at least two completed datasets are pooled using Rubin's rules, requiring positive total variance. FM final imputation is rejected rather than converted to a Wald test. With complete final outcomes, either flag preserves the selected test.
- Imputed final analysis (`imputed_final`). The overall final analysis conducted after accrual is suspended and follow-up is complete can be analyzed on imputed datasets for Bayesian methods (`"bayes-surv"` and `"bayes-bin"`), Cox regression, RMST, and the frequentist risk-difference Wald analysis, or on the non-imputed dataset. Interim prediction completes outcomes that are not yet observed, whereas final imputation applies only to subjects right-censored because of loss to follow-up before `end_of_study`. Design evaluations should pre-specify whether the final analysis imputes these outcomes and assess sensitivity to that choice, particularly when appreciable attrition is expected. Simulated dropout is independent of event time within each arm, as described under `prop_loss`. This supports survival analyses with right-censoring, but does not make complete-case binary analysis unbiased: early events can be observed before dropout, whereas later endpoint outcomes can be missing. Binary designs

with dropout should assess model-based final imputation and its assumptions. For Cox regression and RMST the final estimates and variances are pooled with Rubin's rules. It cannot be used with `method = "logrank"`.

When imputation is involved, either at interim analyses or through `imputed_final = TRUE`, the package uses a two-stage impute-then-analyze procedure. First, the piecewise-exponential model is fitted to the *observed* time-to-event data and used to complete pending outcomes. Second, each completed dataset is analyzed using the model selected by `method`.

For `method = "bayes-bin"`, these are deliberately separate models. The imputation model has piecewise hazards with Gamma prior `prior_surv` (or `prior_surv_final` during final imputation), whereas the completed-data analysis has an event probability at `end_of_study` with Beta prior `prior_bin`. The Beta prior is not derived from the Gamma prior, and the two stages are not one joint Bayesian model. Both prior specifications can therefore affect predictive decisions when outcomes require imputation. The "event-time" and "bernoulli" binary-imputation options use the same piecewise-exponential prediction model and do not change this separation.

For `method = "bayes-surv"`, the second analysis instead forms a fresh piecewise-exponential posterior from the completed data and `prior_surv_final`. At interim looks the first stage uses `prior_surv`, allowing predictive borrowing to differ from the final success criterion. For frequentist methods ("`logrank`", "`cox`", "`rmst`", "`riskdiff-wald`", and "`riskdiff-fm`"), each completed dataset uses a standard test rather than a posterior. Imputed Cox, RMST, and risk-difference final analyses pool estimates and variances using Rubin's rules.

At each interim look, follow-up times are masked (censored) to reflect the calendar time of the analysis. The package treats enrollment and randomization as occurring at the same time. Subjects enrolled at the exact interim boundary have zero follow-up time. These times are clamped to `.Machine$double.eps` (approximately 2.2×10^{-16}) so that they contribute negligible but non-zero exposure to the interim posterior. This affects at most one subject per interim look.

Value

With `return_trace = FALSE` (the default), a one-row data frame containing the evaluated design and final trial results, including:

- `N_treatment`: Number of patients enrolled in the treatment arm.
- `N_control`: Number of patients enrolled in the control arm.
- `est_final`: Treatment effect estimated at the final analysis. For RMST this is treatment minus control RMST through `rmst_tau`, in time units. The final analysis occurs when either the maximum sample size is reached and follow-up is complete, or the interim analysis triggered early stopping of enrollment/accrual and follow-up for those subjects is complete. It is NA after an immediate-success decision, for which no later analysis is required, or when the optional diagnostic analysis after binding futility cannot be computed.
- `post_prob_ha`: Posterior probability from the final analysis. If a Bayesian method uses `imputed_final = TRUE`, this is calculated for each imputed final-analysis dataset and averaged over `N_impute` imputations. For an imputed Cox analysis it is $1 - P$ from the Rubin-pooled Wald test. The same interpretation applies to imputed RMST and risk-difference analyses. For non-imputed frequentist analyses it is $1 - P$ from the corresponding test. It is NA after an immediate-success decision, for which no later analysis is required, or when the optional diagnostic analysis after binding futility cannot be computed.

- `stop_futility`: Logical indicator of whether the trial stopped early for binding futility.
- `stop_immediate_success`: Logical indicator of whether the trial stopped and declared immediate success at an interim look.
- `stop_expected_success`: Logical indicator of whether the trial stopped accruing for expected success and continued planned follow-up.
- `trial_success`: Logical indicator of the trial's official success outcome. An immediate-success decision is final at the interim look; binding futility is a final failure.
- `stopping_reason`: One of "immediate_success", "expected_success", "futility", or "maximum_sample_size".
- `decision_time`: Calendar time at which the trial decision becomes final. This is the interim look time for immediate success or futility, and the analysis-ready time otherwise.
- `accrual_stop_time`: Calendar time of the last enrollment in the trial.
- `analysis_ready_time`: Calendar time at which the last enrolled subject's observed event or censoring becomes available. This excludes external data-cleaning and database-lock delays.
- `planned_completion_time`: Calendar time at which the last enrolled subject would complete the full planned follow-up.
- `followup_person_time`: Sum of observed follow-up times across enrolled subjects.
- `peak_active_followup`: Largest number of enrolled subjects concurrently under follow-up.

Calendar time is measured from the first patient's enrollment at time zero. Times use the same units as `lambda_time`, `cutpoints`, `generation_cutpoints`, and `end_of_study`.

The returned object has a `decision_design` attribute containing `interim_look`, `Fn`, `Sn`, `Qn`, and the Monte Carlo settings. Thresholds in this information are stored as one value per interim look (and have length zero when no interim looks are planned). A `prior_design` attribute contains the resolved Gamma shape, rate, mean hazard, and standard deviation for every stage, arm, and interval.

Both return forms have an `arguments` attribute containing a named list of the evaluated argument values, including defaults. It can be saved with `saveRDS()` and supplied to a later call with `do.call(survival_adapt, attr(result, "arguments"))`. Its `prop_loss` element contains a named value for every simulated arm. Its `rand_ratio` element is stored in `control`, `treatment` order for two-arm designs. Its `cutpoints` and `generation_cutpoints` elements retain the analysis and data-generation partitions, respectively. For `method = "bayes-bin"`, this metadata explicitly retains the imputation priors (`prior_surv` and `prior_surv_final`), completed-data analysis prior (`prior_bin`), and imputation horizon (`end_of_study`). The separate `prior_design` attribute gives the resolved Gamma parameters by stage, arm, and interval.

With `return_trace = TRUE`, a `goldilocks_trial` object is returned. Its `summary` element is the same data frame and its `trace` element has one row per interim look. `prior_diagnostics` contains the resolved interim and final priors. `posterior_diagnostics` reports observed and effective sufficient statistics and conjugate posterior parameters by completed look, arm, and interval. The trace records calendar time, the number of subjects actively under follow-up, enrollment and observed events by arm, predictive probabilities, diagnostic Monte Carlo standard errors and exact bounds, draw counts, thresholds, the decision and reason, empty-interval fallback diagnostics, and warnings raised during that look. It deliberately excludes imputed data sets and posterior draws to keep the output compact.

Predictive and analysis priors

For method = "bayes-surv", prior_surv_final is used **during interim calculations as well as at the actual final analysis**. The two arguments specify different roles, not simply different calendar stages:

Calculation	Gamma prior used
At interim, generate outstanding outcomes for enrolled and future participants	prior_surv
At interim, test each hypothetical completed trial at the current or maximum sample size	prior_surv_final
At final analysis, impute missing outcomes if imputed_final = TRUE	prior_surv_final
Analyze the actual final trial data	prior_surv_final

Within one interim predictive replicate, first update prior_surv with the observed events and exposure, draw hazards, and generate outstanding outcomes. Then start a fresh analysis posterior using prior_surv_final and the completed dataset's events and exposure. Compare its posterior probability of the alternative with prob_ha. The proportion of replicates that pass is the predictive probability used by Qn, Sn, and Fn.

To incorporate external evidence in prediction while using a weak analysis prior, explicitly supply an informative prior_surv and the chosen weak prior_surv_final. **Omitting prior_surv_final uses prior_surv for both roles; the package does not automatically weaken the analysis prior.** The predictive prior can still affect the selected sample size and stopping decision, so calibrate the design using both prespecified priors.

This table describes Bayesian survival analysis. For method = "bayes-bin", completed-data success tests at interim and final use prior_bin; prior_surv_final governs only optional final imputation. Frequentist completed-data tests use no analysis prior. [evaluate_interim\(\)](#) performs the two interim calculations; use the same prior arguments as in the simulated design.

References

Broglia KR, Connor JT, Berry SM. Not too big, not too small: a Goldilocks approach to sample size selection. *Journal of Biopharmaceutical Statistics*, 2014; 24(3): 685–705.

Examples

```
# RCT with exponential hazard (no piecewise breaks)
# Note: the number of imputations is small to enable this example to run
#       quickly on CRAN tests. In practice, much larger values are needed.
survival_adapt(
  hazard_treatment = -log(0.85) / 36,
  hazard_control = -log(0.7) / 36,
  cutpoints = NULL,
  N_total = 600,
  lambda = 20,
  lambda_time = NULL,
  interim_look = 400,
  end_of_study = 36,
  prior_surv = c(0.1, 0.1),
  block = 2,
  rand_ratio = c(control = 1, treatment = 1),
```

```
prop_loss = 0.30,  
alternative = "less",  
h0 = 0,  
Fn = 0.05,  
Sn = 0.9,  
prob_ha = 0.975,  
N_impute = 10,  
N_mcmc = 10,  
method = "bayes-surv")
```

Index

`base::print.data.frame()`, [19](#)

`DiagrammeR::grViz()`, [16](#)

`enrollment`, [2](#)

`enrollment()`, [12](#), [25](#), [26](#), [28](#), [38](#)

`evaluate_interim`, [4](#)

`evaluate_interim()`, [10](#), [16](#), [18](#), [34](#), [36](#), [47](#)

`goldilocks`, [10](#)

`plot_enrollment`, [11](#)

`plot_sim_decisions`, [13](#)

`plot_sim_ocs`, [14](#)

`plot_sim_stopping`, [15](#)

`plot_trial_trace`, [16](#)

`plot_trial_trace()`, [9](#)

`ppwe`, [17](#)

`print.goldilocks_calendar_summary`, [18](#)

`print.goldilocks_interim`, [18](#)

`print.goldilocks_trial`, [19](#)

`prop_to_haz`, [19](#)

`pwe_impute`, [20](#)

`pwe_impute()`, [22](#)

`pwe_sim`, [22](#)

`randomization`, [23](#)

`randomization()`, [25](#), [29](#), [39](#)

`saveRDS()`, [33](#), [46](#)

`sim_comp_data`, [24](#)

`sim_trials`, [27](#)

`sim_trials()`, [12](#), [14–16](#), [34–36](#)

`summarise_calendar_time`, [34](#)

`summarise_calendar_time()`, [18](#), [33](#)

`summarise_sims`, [35](#)

`summarise_sims()`, [15](#)

`summarise_trial_trace`, [36](#)

`summarise_trial_trace()`, [9](#)

`survival_adapt`, [37](#)

`survival_adapt()`, [4](#), [12](#), [16](#), [19](#), [27](#), [32](#), [33](#),
[36](#)