Package 'goldilocks'

October 13, 2022

Title Goldilocks Adaptive Trial Designs for Time-to-Event Endpoints

Version 0.3.0

Description Implements the Goldilocks adaptive trial design for a time to event outcome using a piecewise exponential model and conjugate Gamma prior distributions. The method closely follows the article by Broglio and colleagues <doi:10.1080/10543406.2014.888569>, which allows users to explore the operating characteristics of different trial designs.

License GPL-3

Encoding UTF-8

Depends R (>= 3.6.0), survival

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

Suggests covr, testthat (>= 3.0.0), knitr, rmarkdown

Language en-US

RoxygenNote 7.1.1

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Graeme L. Hickey [aut, cre] (<https://orcid.org/0000-0002-4989-0054>),
Ying Wan [aut],
Thevaa Chandereng [aut] (<https://orcid.org/0000-0003-4078-9176>,
bayesDP code as a template),
Becton, Dickinson and Company [cph]

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

Repository CRAN

Date/Publication 2021-05-10 08:20:11 UTC

R topics documented:

`enrollment` *Simulate enrollment times*

Description

Simulate enrollment time using a piecewise Poisson distribution.

Usage

```
enrollment(lambda = 1, N_total, lambda_time = 0)
```

Arguments

`lambda`	vector. Rate parameter(s) for Poisson distribution.
`N_total`	integer. Value of total sample size.
`lambda_time`	vector. Knots (of `length(lambda)`) indicating regions where a specific hazard rate (`lambda`) applies. The first element is always `lambda_time = 0`, denoting the trial start time. Note: final element of `lambda` is assumed to be constant as `lambda_time` tends to infinity.

Details

Subject recruitment is assumed to follow a (piecewise stationary) Poisson process. We assume trial recruitment to be an independent process, thus the 'memoryless' property modelling of subject recruitment is used. Since the subject recruitment rate can vary over time, we can account for differential rates over time. Note that the first trial enrollment is assumed to occur at time zero.

To illustrate, suppose we use a piecewise function to specify the change in enrollment rate over time:

$$\lambda = \begin{cases} 0.3 & \text{time} \in [0, 5) \\ 0.7 & \text{time} \in [5, 10) \\ 0.9 & \text{time} \in [10, 15) \\ 1.2 & \text{time} \in [15, \infty) \end{cases}$$

Then, to simulate individual patient enrollment dates with a sample size (`N_total`) of 50, we use

```
enrollment(lambda = c(0.3, 0.7, 0.9, 1.2), N_total = 50, lambda_time = c(0, 5, 10, 15))
```

Value

A vector of enrollment times (from time of first patient enrollment) in unit time (e.g. days).

See Also

This function is based on the `enrollment` function from the `bayesCT` R package.

Examples

```
enrollment(lambda = c(0.003, 0.7), N_total = 100, lambda_time = c(0, 10))
enrollment(lambda = c(0.3, 0.5, 0.9, 1.2, 2.1), N_total = 200,
           lambda_time = c(0, 20, 30, 40, 60))
```

`goldilocks` *goldilocks*

Description

The goal of `goldilocks` is to implement the Goldilocks Bayesian adaptive design proposed by Broglio et al. (2014) for time-to-event endpoint trials, both one- and two-arm, with an underlying piecewise exponential hazard model. The method can be used for a confirmatory trial to select a trial's sample size based on accumulating data. During accrual, frequent sample size selection analyses are made and predictive probabilities are used to determine whether the current sample size is sufficient or whether continuing accrual would be futile. The algorithm explicitly accounts for complete follow-up of all patients before the primary analysis is conducted. 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.

`ppwe` *Cumulative distribution function of the PWE for a vectorized hazard rate parameter*

Description

Extends the `pwe` function to allow for vectorization over the hazard rates.

Usage

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

Arguments

hazard	matrix. A matrix of hazard rate parameters with number of columns equal to the length of the cutpoints vector. The number of rows can be anything, and is typically dictated by the number of MCMC draws.
end_of_study	scalar. Length of the study; i.e. time at which endpoint will be evaluated.
cutpoints	vector. The change-point vector indicating time when the hazard rates change. Note the first element of cutpoints should always be 0.

Value

A vector of (0, 1) probabilities from evaluation of the PWE cumulative distribution function. Length of the vector matches the number of rows of the hazard matrix parameter.

---

prop_to_haz *Estimate plausible piecewise constant hazard rates from summary summary event proportions*

Description

Given estimates of the event probability at one or more fixed times, the corresponding piecewise hazard rates can be determined through closed-form formulae. This utility function can be useful when simulating trial datasets with plausible event rates.

Usage

```
prop_to_haz(probs, cutpoints = 0, endtime)
```

Arguments

probs	vector. Probabilities of the event (i.e. cumulative incidence probabilities) at one or more time point. If only a single value is given, then it is assumed that this is the probability at the endtime.
cutpoints	vector. Times at which the baseline hazard changes. Default is cutpoints = 0, which corresponds to a simple (non-piecewise) exponential model.
endtime	scalar. Time at which final element in probs corresponds to. Typically this would be the study endpoint time.

Details

Given $J-1$ internal cut-points, then there are J intervals defined as: $[s_0, s_1)$, $[s_1, s_2)$, ..., $[s_{J-1}, s_J)$, with conditions that $s_0 = 0$ and $s_J = \infty$. Each interval corresponds to constant hazard λ_j.

Value

Vector of constant hazard rates for each time piece defined by cutpoints.

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), c(0, 12), 24)
PWEALL::pwe(12, prop_to_haz(c(0.15, 0.30), c(0, 12), 24), c(0, 12))$dist
PWEALL::pwe(24, prop_to_haz(c(0.15, 0.30), c(0, 12), 24), c(0, 12))$dist
```

`pwe_impute` *Impute piecewise exponential time-to-event outcomes*

Description

Imputation of time-to-event outcomes using the piecewise constant hazard exponential function conditional on observed exposure.

Usage

```
pwe_impute(time, hazard, cutpoints = 0, maxtime = NULL)
```

Arguments

`time`	vector. The observed time for patient that have had no event or passed `maxtime`.
`hazard`	vector. The constant hazard rates for exponential failures.
`cutpoints`	vector. The change-point vector indicating time when the hazard rates change. Note the first element of `cutpoints` should always be 0.
`maxtime`	scalar. Maximum time before end of study.

Details

If a subject is event-free at time $s < t$, then the conditional probability $F_{T\|s}|(t\|s) = P[T \leq \|T > s] = (F(t) - F(s))/(1 - F(s))$, 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 $F^{-1}(U(1 - F(s)) + F(s))$. Note: if $s = 0$, then this is the equivalent of a direct (unconditional) sample from the PWE distribution.

Value

A data frame with simulated follow-up times (`time`) and respective event indicator (`event`, 1 = event occurred, 0 = censoring).

Examples

```
pwe_impute(time = c(3, 4, 5), hazard = c(0.002, 0.01), cutpoints = c(0, 12))
pwe_impute(time = c(3, 4, 5), hazard = c(0.002, 0.01), cutpoints = c(0, 12),
           maxtime = 36)
pwe_impute(time = 19.621870008, hazard = c(2.585924e-02, 3.685254e-09),
           cutpoints = c(0, 12), maxtime = 36)
```

---

pwe_sim *Simulate piecewise exponential time-to-event outcomes*

Description

Simulate time-to-event outcomes using the piecewise constant hazard exponential function.

Usage

```
pwe_sim(n = 1, hazard = 1, cutpoints = 0, maxtime = NULL)
```

Arguments

n	integer. The number of random samples to generate. Default is n=1.
hazard	vector. The constant hazard rates for exponential failures.
cutpoints	vector. The change-point vector indicating time when the hazard rates change. Note the first element of cutpoints should always be 0.
maxtime	scalar. Maximum time before end of study.

Details

See pwe_impute for details.

Value

A data frame with simulated follow-up times (time) and respective event indicator (event, 1 = event occurred, 0 = censoring).

Examples

```
pwe_sim(10, hazard = c(0.005, 0.001), cutpoints = c(0, 3), maxtime = 36)
y <- pwe_sim(n = 1, hazard = c(2.585924e-02, 3.685254e-09),
             cutpoints = c(0, 12))
```

`randomization` *Randomization allocation*

Description

Implements a randomization allocation for control and treatment arms with different randomization ratios and block sizes.

Usage

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

Arguments

`N_total`	integer. Total sample size for randomization allocation.
`block`	vector. Block size for randomization. Note that it needs to be a multiple of the sum of `allocation`.
`allocation`	vector. The randomization allocation in the order `c(control, treatment)`.

Details

Complete randomization may not always be ideal due to the chance of drawing a large block of a single treatment arm, potentially impacting the time to enrollment completion. Therefore, a block randomization allocation may be preferable. The block randomization allocation specification allows for different randomization ratios, but they must be given in integer form. Additionally, the block size should be an integer that is divisible by the sum of the randomization allocation; see the examples.

Value

The randomization allocation with 0, 1 for control and treatment, respectively.

Examples

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

# Treatment allocation with 2:1 for control to treatment
randomization(N_total = 70, block = 9, allocation = c(2, 1))

# 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(1, 2))

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

`sim_trials` *Simulate one or more clinical trials subject to known design parameters and treatment effect*

Description

Simulate multiple clinical trials with fixed input parameters, and tidily extract the relevant data to generate operating characteristics.

Usage

```
sim_trials(
  hazard_treatment,
  hazard_control = NULL,
  cutpoints = 0,
  N_total,
  lambda = 0.3,
  lambda_time = 0,
  interim_look = NULL,
  end_of_study,
  prior = c(0.1, 0.1),
  block = 2,
  rand_ratio = c(1, 1),
  prop_loss = 0,
  alternative = "two.sided",
  h0 = 0,
  Fn = 0.1,
  Sn = 0.9,
  prob_ha = 0.95,
  N_impute = 10,
  N_mcmc = 10,
  N_trials = 10,
  method = "logrank",
  imputed_final = FALSE,
  ncores = 1L
)
```

Arguments

`hazard_treatment`	vector. Constant hazard rates under the treatment arm.
`hazard_control`	vector. Constant hazard rates under the control arm.
`cutpoints`	vector. Times at which the baseline hazard changes. Default is `cutpoints = 0`, which corresponds to a simple (non-piecewise) exponential model.
`N_total`	integer. Maximum sample size allowable

lambda	vector. Enrollment rates across simulated enrollment times. See enrollment for more details.
lambda_time	vector. Enrollment time(s) at which the enrollment rates change. Must be same length as lambda. See enrollment for more details.
interim_look	vector. Sample size for each interim look. Note: the maximum sample size should not be included.
end_of_study	scalar. Length of the study; i.e. time at which endpoint will be evaluated.
prior	vector. The prior distributions for the piecewise hazard rate parameters are each $Gamma(a_0, b_0)$, with specified (known) hyper-parameters a_0 and b_0. The default non-informative prior distribution used is Gamma(0.1, 0.1), which is specified by setting prior = c(0.1, 0.1).
block	scalar. Block size for generating the randomization schedule.
rand_ratio	vector. Randomization allocation for the ratio of control to treatment. Integer values mapping the size of the block. See randomization for more details.
prop_loss	scalar. Overall proportion of subjects lost to follow-up. Defaults to zero.
alternative	character. The string specifying the alternative hypothesis, must be one of "greater" (default), "less" or "two.sided".
h0	scalar. Null hypothesis value of $p_{\text{treatment}} - p_{\text{control}}$ when method = "bayes". Default is h0 = 0. The argument is ignored when method = "logrank" or = "cox"; in those cases the usual test of non-equal hazards is assumed.
Fn	vector of [0, 1] values. Each element is the probability threshold to stop at the i-th look early for futility. If there are no interim looks (i.e. interim_look = NULL), then Fn is not used in the simulations or analysis. The length of Fn should be the same as interim_look, else the values are recycled.
Sn	vector of [0, 1] values. Each element is the probability threshold to stop at the i-th look early for expected success. If there are no interim looks (i.e. interim_look = NULL), then Sn is not used in the simulations or analysis. The length of Sn should be the same as interim_look, else the values are recycled.
prob_ha	scalar [0, 1]. Probability threshold of alternative hypothesis.
N_impute	integer. Number of imputations for Monte Carlo simulation of missing data.
N_mcmc	integer. Number of samples to draw from the posterior distribution when using a Bayesian test (method = "bayes").
N_trials	integer. Number of trials to simulate.
method	character. For an imputed data set (or the final data set after follow-up is complete), whether the analysis should be a log-rank (method = "logrank") test, Cox proportional hazards regression model Wald test (method = "cox"), or a fully-Bayesian analysis (method = "bayes"). See Details section.
imputed_final	logical. Should the final analysis (after all subjects have been followed-up to the study end) be based on imputed outcomes for subjects who were LTFU (i.e. right-censored with time <end_of_study)? Default is TRUE. Setting to FALSE means that the final analysis would incorporate right-censoring.
ncores	integer. Number of cores to use for parallel processing.

Details

This is basically a wrapper function for survival_adapt, whereby we repeatedly run the function for a independent number of trials (all with the same input design parameters and treatment effect).

To use will multiple cores (where available), the argument ncores can be increased from the default of 1. Note: on Windows machines, it is not possible to use the mclapply function with ncores > 1.

Value

Data frame with 1 row per simulated trial and columns for key summary statistics. See survival_adapt for details of what is returned in each row.

Examples

```
hc <- prop_to_haz(c(0.20, 0.30), c(0, 12), 36)
ht <- prop_to_haz(c(0.05, 0.15), c(0, 12), 36)

out <- sim_trials(
  hazard_treatment = ht,
  hazard_control = hc,
  cutpoints = c(0, 12),
  N_total = 600,
  lambda = 20,
  lambda_time = 0,
  interim_look = c(400, 500),
  end_of_study = 36,
  prior = c(0.1, 0.1),
  block = 2,
  rand_ratio = c(1, 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,
  N_trials = 2,
  method = "logrank",
  ncores = 1)
```

---

summarise_sims *Summarize simulations to get operating characteristics*

Description

Summarize simulations to get operating characteristics

Usage

```
summarise_sims(data)
```

Arguments

data	list (of data frames) or a single data frame. If summarizing a single run of simulations, data will be a data.frame object returned from survival_adapt. If summarizing multiple simulation scenarios, data will be a list object, with each element being a data.frame object.

Value

Data frame reporting the operating characteristics, including the type 2 error (which will be equal to the type I error in the null case); the proportion of trials that stopped for early expected success, futility, or went to the maximum sample size. The average stopping sample size (and standard deviation) are also recorded. The proportion of trials that stopped early for expected success, yet went to ultimately fail are also reported.

---

survival_adapt *Simulate a single adaptive clinical trial with a time-to-event endpoint*

Description

Simulate a single adaptive clinical trial with a time-to-event endpoint

Usage

```
survival_adapt(
  hazard_treatment,
  hazard_control = NULL,
  cutpoints = 0,
  N_total,
  lambda = 0.3,
  lambda_time = 0,
  interim_look = NULL,
  end_of_study,
  prior = c(0.1, 0.1),
  block = 2,
  rand_ratio = c(1, 1),
  prop_loss = 0,
  alternative = "greater",
  h0 = 0,
  Fn = 0.05,
  Sn = 0.9,
  prob_ha = 0.95,
  N_impute = 10,
  N_mcmc = 10,
  method = "logrank",
  imputed_final = FALSE,
  debug = FALSE
)
```

Arguments

hazard_treatment	vector. Constant hazard rates under the treatment arm.
hazard_control	vector. Constant hazard rates under the control arm.
cutpoints	vector. Times at which the baseline hazard changes. Default is `cutpoints = 0`, which corresponds to a simple (non-piecewise) exponential model.
N_total	integer. Maximum sample size allowable
lambda	vector. Enrollment rates across simulated enrollment times. See `enrollment` for more details.
lambda_time	vector. Enrollment time(s) at which the enrollment rates change. Must be same length as lambda. See `enrollment` for more details.
interim_look	vector. Sample size for each interim look. Note: the maximum sample size should not be included.
end_of_study	scalar. Length of the study; i.e. time at which endpoint will be evaluated.
prior	vector. The prior distributions for the piecewise hazard rate parameters are each $Gamma(a_0, b_0)$, with specified (known) hyper-parameters a_0 and b_0. The default non-informative prior distribution used is Gamma(0.1, 0.1), which is specified by setting `prior = c(0.1, 0.1)`.
block	scalar. Block size for generating the randomization schedule.
rand_ratio	vector. Randomization allocation for the ratio of control to treatment. Integer values mapping the size of the block. See `randomization` for more details.
prop_loss	scalar. Overall proportion of subjects lost to follow-up. Defaults to zero.
alternative	character. The string specifying the alternative hypothesis, must be one of `"greater"` (default), `"less"` or `"two.sided"`.
h0	scalar. Null hypothesis value of $p_{\text{treatment}} - p_{\text{control}}$ when `method = "bayes"`. Default is `h0 = 0`. The argument is ignored when `method = "logrank"` or `= "cox"`; in those cases the usual test of non-equal hazards is assumed.
Fn	vector of `[0, 1]` values. Each element is the probability threshold to stop at the i-th look early for futility. If there are no interim looks (i.e. `interim_look = NULL`), then `Fn` is not used in the simulations or analysis. The length of `Fn` should be the same as `interim_look`, else the values are recycled.
Sn	vector of `[0, 1]` values. Each element is the probability threshold to stop at the i-th look early for expected success. If there are no interim looks (i.e. `interim_look = NULL`), then `Sn` is not used in the simulations or analysis. The length of `Sn` should be the same as `interim_look`, else the values are recycled.
prob_ha	scalar `[0, 1]`. Probability threshold of alternative hypothesis.
N_impute	integer. Number of imputations for Monte Carlo simulation of missing data.
N_mcmc	integer. Number of samples to draw from the posterior distribution when using a Bayesian test (`method = "bayes"`).
method	character. For an imputed data set (or the final data set after follow-up is complete), whether the analysis should be a log-rank (`method = "logrank"`) test, Cox proportional hazards regression model Wald test (`method = "cox"`), or a fully-Bayesian analysis (`method = "bayes"`). See Details section.

`imputed_final` logical. Should the final analysis (after all subjects have been followed-up to the study end) be based on imputed outcomes for subjects who were LTFU (i.e. right-censored with time <`end_of_study`)? Default is `TRUE`. Setting to `FALSE` means that the final analysis would incorporate right-censoring.

`debug` logical. If `TRUE` can be used to debug aspects of the code, including producing Kaplan-Meier graphs at each step of the algorithm. Default is `debug = FALSE`.

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 would go on to be success at the specified threshold. At each interim analysis it is compared to the corresponding element of `Sn`, and if it exceeds the threshold, accrual/enrollment is suspended and the outstanding follow-up allowed to complete before conducting the pre-specified final analysis.
2. **The posterior predictive probability of final success**. This is calculated as the proportion of imputed datasets at the *maximum* threshold that would go on to be successful. Similar to above, it is compared to the corresponding element of `Fn`, and if it is less than 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 `Fn = 0`.

Hence, at each interim analysis look, 3 decisions are allowed:

1. **Stop for expected success**
2. **Stop for futility**
3. **Continue to enroll** new subjects, or if at maximum sample size, proceed to final analysis.

At each interim (and final) analysis methods as:

- 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 `post_prob_ha` > 0.95. The reason for this is to enable simple switching between Bayesian and frequentist paradigms for analysis.
- Cox proportional hazards regression Wald test (`method = "cox"`). Similar to the log-rank test, a *P*-value is calculated based on a two-sided test. However, for consistency, $1 - P$, 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 `post_prob_ha` > 0.95.
- Bayesian absolute difference (`method = "bayes"`). Each imputed dataset is used to update the conjugate Gamma prior (defined by `prior`), 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. If a single arm study, then this summarizes the treatment effect, else, if a two-armed study, the independent posteriors are used to estimate the posterior distribution of the

difference. A posterior probability is calculated according to the specification of the test type (`alternative`) and the value of the null hypothesis (`h0`).

- 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 (default) or on the non-imputed dataset. Since the imputations/predictions used during the interim analyses assume all subjects are imputed (since loss to follow-up is not yet known), it would seem most appropriate to conduct the trial in the same manner, especially if loss to follow-up rates are appreciable. Note, this only applies to subjects who are right-censored due to loss to follow-up, which we assume is a non-informative process. This can be used with any `method`.

Value

A data frame containing some input parameters (arguments) as well as statistics from the analysis, including:

`N_treatment` integer. The number of patients enrolled in the treatment arm for each simulation.

`N_control` integer. The number of patients enrolled in the control arm for each simulation.

`est_interim` scalar. The treatment effect that was estimated at the time of the interim analysis. Note this is not actually used in the final analysis.

`est_final` scalar. The treatment effect that was estimated at the final analysis. Final analysis occurs when either the maximum sample size is reached and follow-up complete, or the interim analysis triggered an early stopping of enrollment/accrual and follow-up for those subjects is complete.

`post_prob_ha` scalar. The corresponding posterior probability from the final analysis. If `imputed_final` is true, this is calculated as the posterior probability of efficacy (or equivalent, depending on how `alternative` and `h0` were specified) for each imputed final analysis dataset, and then averaged over the `N_impute` imputations. If `method = "logrank"`, `post_prob_ha` is calculated in the same fashion, but value represents $1 - P$, where P denotes the frequentist P-value.

`stop_futility` integer. A logical indicator of whether the trial was stopped early for futility.

`stop_expected_success` integer. A logical indicator of whether the trial was stopped early for expected success.

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.

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 = 0,
 N_total = 600,
 lambda = 20,
```

```
  lambda_time = 0,
  interim_look = 400,
  end_of_study = 36,
  prior = c(0.1, 0.1),
  block = 2,
  rand_ratio = c(1, 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")
```

Index