Package ‘lrstat’

February 9, 2022

Type Package

Title Power and Sample Size Calculation for Non-Proportional Hazards

Version 0.1.3

Date 2022-01-12

Author Kaifeng Lu

Maintainer Kaifeng Lu <kaifenglu@gmail.com>

Description Performs power and sample size calculation for non-proportional hazards model using the Fleming-Harrington family of weighted log-rank tests. The sequentially calculated log-rank test score statistics are assumed to have independent increments as characterized in Anastasios A. Tsiatis (1982) <doi:10.1080/01621459.1982.10477898>. The mean and variance of log-rank test score statistics are calculated based on Edward Lakatos (1986) <doi:10.1016/0197-2456(86)90047-4> and Kaifeng Lu (2021) <doi:10.1002/pst.2069>. The boundary crossing probabilities are calculated using the recursive integration algorithm described in Christopher Jennison and Bruce W. Turnbull (2000, ISBN:0849303168). The inverse normal combination method due to Lu Cui, H. M. James Hung, and Sue-Jane Wang (1999) <doi:10.1111/j.0006-341x.1999.00853.x> is used in the simulation study to accommodate the information time not proportional to the total number of events for weighted log-rank tests.

License GPL (>= 2)

Imports Rcpp (>= 1.0.7)

LinkingTo Rcpp

RoxygenNote 7.1.2

Suggests knitr, rmarkdown, rpact, mvtnorm, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation yes

Config/testthat/edition 3

Repository CRAN

Date/Publication 2022-02-09 12:20:07 UTC

R topics documented:

`lrstat-package` *Power and Sample Size Calculation for Non-Proportional Hazards*

Description

Performs power and sample size calculation for non-proportional hazards model using the Fleming-Harrington family of weighted log-rank tests.

Details

For proportional hazards, the power is determined by the total number of events and the constant hazard ratio along with the information rates and the spending functions. For non-proportional hazards, the hazard ratio varies over time and the calendar time plays a key role in determining the mean and variance of the log-rank test score statistic. It requires an iterative algorithm to find the calendar time at which the targeted number of events will be reached for each interim analysis. The lrstat package uses the discrete state nonstationary Markov process method from Lakatos (1986) and further developed in Lu (2021) to find the mean and variance of the weighted log-rank test score statistic at each interim analysis. In addition, the package approximates the variance and covariance matrix of the sequentially calculated log-rank test statistics under the alternative hypothesis with that under the null hypothesis to take advantage of the independent increments structure in Tsiatis (1982) applicable for the Fleming-Harrington family of weighted log-rank tests.

The most useful functions in the package are `lrstat`, `lrpower`, `lrsamplesize`, and `lrsim`, which calculate the mean and variance of log-rank test score statistic at a sequence of given calendar times, the power of the log-rank test, the sample size in terms of accrual duration and follow-up duration, and the log-rank test simulation, respectively. The `accrual` function calculates the number of patients accrued at given calendar times. The `caltime` function finds the calendar times to reach the targeted number of events. The `exitprob` function calculates the stagewise exit probabilities for specified boundaries with a varying mean parameter over time based on an adaptation of the recursive integration algorithm described in Chapter 19 of Jennison and Turnbull (2000).

For critical values on the z-statistic scale, we recommend using standard group sequential design packages such as rpact and gsDesign as the mean parameter is zero under the null hypothesis. However, the information is not proportional to the number of events for weighted log-rank tests, in which case, the `caltime` and `lrstat` functions can be used together to find the appropriate information fractions.

With changing hazard ratio over time, the futility bounds derived from a beta-spending function for the standard group sequential design with a fixed parameter value cannot be used directly, in which case, the `exitprob` function can be used to derive the futility bounds.

The development of the lrstat package is heavily influenced by the rpact package. We find their function arguments to be self-explanatory. We have used the same names whenever appropriate so that users familiar with the rpact package can learn the lrstat package quickly. However, there are notable differences:

- lrstat uses the Lakatos method, while rpact uses the Schoenfeld method for log-rank test power and sample size calculation.
- lrstat uses `accrualDuration` to explicitly set the end of accrual period, while rpact incorporates the end of accrual period in `accrualTime`.
- lrstat considers the trial a failure at the last stage if the log-rank test cannot reject the null hypothesis up to this stage and cannot stop for futility at an earlier stage.
- the `lrsim` function directly uses the variance of the log-rank test score statistic as the information and uses the inverse normal combination method of Cui, Hung, and Wang (1999) to construct the test statistic at each stage.

Author(s)

Kaifeng Lu, <kaifenglu@gmail.com>

References

Anastasios A. Tsiatis. Repeated significance testing for a general class of statistics used in censored survival analysis. J Am Stat Assoc. 1982;77:855-861. doi: 10.1080/01621459.1982.10477898

Edward Lakatos. Sample size determination in clinical trials with time-dependent rates of losses and noncompliance. Control Clin Trials. 1986;7:189-199. doi: 10.1016/01972456(86)900474

Lu Cui, H. M. James Hung, Sue-Jane Wang. Modification of sample size in group sequential clinical trials. Biometrics. 1999;55:853-857. doi: 10.1111/j.0006341x.1999.00853.x

Christopher Jennison, Bruce W. Turnbull. Group Sequential Methods with Applications to Clinical Trials. Chapman & Hall/CRC: Boca Raton, 2000, ISBN:0849303168

Kaifeng Lu. Sample size calculation for logrank test and prediction of number of events over time. Pharm Stat. 2021;20:229-244. doi: 10.1002/pst.2069

See Also

rpact, gsDesign

Examples

```
lrpower(kMax = 2, informationRates = c(0.8, 1),
        criticalValues = c(2.250, 2.025), accrualIntensity = 20,
        piecewiseSurvivalTime = c(0, 6),
        lambda1 = c(0.0533, 0.0309), lambda2 = c(0.0533, 0.0533),
        gamma1 = 0.00427, gamma2 = 0.00427,
        accrualDuration = 22, followupTime = 18)
```

`accrual` *Number of enrolled subjects*

Description

Obtains the number of subjects enrolled by given calendar times.

Usage

```
accrual(
  time = NA_real_,
  accrualTime = 0L,
  accrualIntensity = NA_real_,
  accrualDuration = NA_real_
)
```

Arguments

`time`	A vector of calendar times at which to calculate the number of enrolled subjects.
`accrualTime`	Accrual time intervals, must start with 0, e.g., `c(0,3)` breaks the time axis into 2 accrual intervals: [0, 3) and [3, Inf).
`accrualIntensity`	A vector of accrual intensities, one for each accrual time interval.
`accrualDuration`	Duration of the enrollment period.

Value

A vector of total number of subjects enrolled by the specified calendar times.

Examples

```
# Example 1: Uniform enrollment with 20 patients per month for 12 months.
accrual(time = 3, accrualTime = 0, accrualIntensity = 20,
    accrualDuration = 12)

# Example 2: Piecewise accrual, 10 patients per month for the first
# 3 months, and 20 patients per month thereafter. Patient recruitment
# ends at 12 months for the study.
accrual(time = c(2, 9), accrualTime = c(0, 3),
    accrualIntensity = c(10, 20), accrualDuration = 12)
```

caltime *Calendar times for target number of events*

Description

Obtains the calendar times to reach the target number of subjects having an event.

Usage

```
caltime(
  nevents = NA_real_,
  allocationRatioPlanned = 1,
  accrualTime = 0L,
  accrualIntensity = NA_real_,
  piecewiseSurvivalTime = 0L,
  stratumFraction = 1L,
  lambda1 = NA_real_,
  lambda2 = NA_real_,
  gamma1 = 0L,
  gamma2 = 0L,
  accrualDuration = NA_real_,
  followupTime = NA_real_,
  fixedFollowup = 0L,
  numSubintervals = 300L
)
```

Arguments

`nevents` A vector of target number of events.

`allocationRatioPlanned` Allocation ratio for the active treatment versus control. Defaults to 1 for equal randomization.

`accrualTime` Accrual time intervals, must start with 0, e.g., `c(0,3)` breaks the time axis into 2 accrual intervals: [0, 3) and [3, Inf).

`accrualIntensity` A vector of accrual intensities, one for each accrual time interval.

`piecewiseSurvivalTime` A vector that specifies the time intervals for the piecewise exponential survival distribution, must start with 0, e.g., `c(0,6)` breaks the time axis into 2 event intervals: [0, 6) and [6, Inf). Defaults to 0 for exponential distribution.

`stratumFraction` A vector of stratum fractions that sum to 1. Defaults to 1 for no stratification.

`lambda1` A vector of hazard rates for the event for the active treatment group, one for each analysis time interval, by stratum.

`lambda2` A vector of hazard rates for the event for the control group, one for each analysis time interval, by stratum.

`gamma1` The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the active treatment group. Defaults to 0 for no dropout.

`gamma2` The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the control group. Defaults to 0 for no dropout.

`accrualDuration` Duration of the enrollment period.

`followupTime` Follow-up time for the last enrolled subject.

`fixedFollowup` Whether a fixed follow-up design is used. Defaults to 0 for variable follow-up.

`numSubintervals` Number of sub-intervals to approximate the mean and variance of the weighted log-rank test score statistic. Defaults to 300. Specify a larger number for better approximation.

Value

A vector of calendar times expected to yield the target number of events.

Examples

```
# Piecewise accrual, piecewise exponential survivals, and 5% dropout by
# the end of 1 year.
caltime(nevents = c(24, 80), allocationRatioPlanned = 1,
    accrualTime = seq(0, 9),
    accrualIntensity = c(26/9*seq(1, 9), 26),
    piecewiseSurvivalTime = c(0, 6),
    stratumFraction = c(0.2, 0.8),
    lambda1 = c(0.0533, 0.0309, 1.5*0.0533, 1.5*0.0309),
    lambda2 = c(0.0533, 0.0533, 1.5*0.0533, 1.5*0.0533),
    gamma1 = -log(1-0.05)/12,
    gamma2 = -log(1-0.05)/12,
    accrualDuration = 22,
    followupTime = 18, fixedFollowup = FALSE)
```

`exitprob` *Stagewise exit probabilities*

Description

Obtains the stagewise exit probabilities for both efficacy and futility stopping.

Usage

```
exitprob(b = NA_real_, a = NA_real_, theta = NA_real_, I = NA_real_, r = 18L)
```

Arguments

- `b` Upper boundaries on the z-test statistic scale.
- `a` Lower boundaries on the z-test statistic scale. Defaults to `c(rep(-Inf,kMax-1),b[kMax])` if left unspecified, where `kMax = length(b)`.
- `theta` Stagewise parameter of interest, e.g., `-U/V` for weighted log-rank test, where `U` is the mean and `V` is the variance of the weighted log-rank test score statistic at each stage. For proportional hazards and conventional log-rank test, use the scalar input, `theta = -log(HR)`.
- `I` Stagewise cumulative information, e.g., `V`, the variance of the weighted log-rank test score statistic at each stage. For conventional log-rank test, information can be approximated by `phi*(1-phi)*D`, where `phi` is the probability of being allocated to the active arm, and `D` is the total number of events at each stage.
- `r` Integer value controlling grid for numerical integration as in Jennison and Turnbull (2000). Defaults to 18. Specify a larger number for greater accuracy.

Value

A list of stagewise exit probabilities: one vector for efficacy stopping probabilities, and the other vector for futility stopping probabilities.

Examples

```
exitprob(b = c(3.471, 2.454, 2.004), theta = -log(0.6),
    I = c(50, 100, 150)/4)
```

---

lrpower *Log-rank test power*

Description

Estimates the power, stopping probabilities, and expected sample size in a two-group survival design.

Usage

```
lrpower(
  kMax = NA_integer_,
  informationRates = NA_real_,
  efficacyStopping = NA_integer_,
  futilityStopping = NA_integer_,
  criticalValues = NA_real_,
  alpha = 0.025,
  typeAlphaSpending = "sfOF",
  parameterAlphaSpending = NA_real_,
  userAlphaSpending = NA_real_,
```

```
  futilityBounds = NA_real_,
  typeBetaSpending = "none",
  parameterBetaSpending = NA_real_,
  allocationRatioPlanned = 1,
  accrualTime = 0L,
  accrualIntensity = 20L,
  piecewiseSurvivalTime = 0L,
  stratumFraction = 1L,
  lambda1 = 0.0309,
  lambda2 = 0.0533,
  gamma1 = 0L,
  gamma2 = 0L,
  accrualDuration = 11.6,
  followupTime = 18,
  fixedFollowup = 0L,
  rho1 = 0,
  rho2 = 0,
  numSubintervals = 300L
)
```

Arguments

`kMax` The maximum number of stages.

`informationRates` The information rates in terms of number of events. Fixed prior to the trial. Defaults to `(1:kMax) / kMax` if left unspecified.

`efficacyStopping` Indicators of whether efficacy stopping is allowed at each stage. Defaults to true if left unspecified.

`futilityStopping` Indicators of whether futility stopping is allowed at each stage. Defaults to true if left unspecified.

`criticalValues` Upper boundaries on the z-test statistic scale for stopping for efficacy.

`alpha` The significance level. Defaults to 0.025.

`typeAlphaSpending` The type of alpha spending. One of the following: "OF" for O'Brien-Fleming boundaries, "P" for Pocock boundaries, "WT" for Wang & Tsiatis boundaries, "sfOF" for O'Brien-Fleming type spending function, "sfP" for Pocock type spending function, "sfKD" for Kim & DeMets spending function, "sfHSD" for Hwang, Shi & DeCani spending function, "user" for user defined spending, and "none" for no early efficacy stopping. Defaults to "sfOF".

`parameterAlphaSpending` The parameter value for the alpha spending. Corresponds to Delta for "WT", rho for "sfKD", and gamma for "sfHSD".

`userAlphaSpending` The user defined alpha spending. Cumulative alpha spent up to each stage.

`futilityBounds` Lower boundaries on the z-test statistic scale for stopping for futility at stages 1, ..., kMax-1. Defaults to `rep(-Inf,kMax-1)` if left unspecified.

`typeBetaSpending`
The type of beta spending. One of the following: "sfOF" for O'Brien-Fleming type spending function, "sfP" for Pocock type spending function, "sfKD" for Kim & DeMets spending function, "sfHSD" for Hwang, Shi & DeCani spending function, and "none" for no early futility stopping. Defaults to "none".

`parameterBetaSpending`
The parameter value for the beta spending. Corresponds to rho for "sfKD", and gamma for "sfHSD".

`allocationRatioPlanned`
Allocation ratio for the active treatment versus control. Defaults to 1 for equal randomization.

`accrualTime` Accrual time intervals, must start with 0, e.g., `c(0,3)` breaks the time axis into 2 accrual intervals: [0, 3) and [3, Inf).

`accrualIntensity`
A vector of accrual intensities, one for each accrual time interval.

`piecewiseSurvivalTime`
A vector that specifies the time intervals for the piecewise exponential survival distribution, must start with 0, e.g., `c(0,6)` breaks the time axis into 2 event intervals: [0, 6) and [6, Inf). Defaults to 0 for exponential distribution.

`stratumFraction`
A vector of stratum fractions that sum to 1. Defaults to 1 for no stratification.

`lambda1` A vector of hazard rates for the event for the active treatment group, one for each analysis time interval, by stratum.

`lambda2` A vector of hazard rates for the event for the control group, one for each analysis time interval, by stratum.

`gamma1` The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the active treatment group. Defaults to 0 for no dropout.

`gamma2` The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the control group. Defaults to 0 for no dropout.

`accrualDuration`
Duration of the enrollment period.

`followupTime` Follow-up time for the last enrolled subject.

`fixedFollowup` Whether a fixed follow-up design is used. Defaults to 0 for variable follow-up.

`rho1` First parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

`rho2` Second parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

`numSubintervals`
Number of sub-intervals to approximate the mean and variance of the weighted log-rank test score statistic. Defaults to 300. Specify a larger number for better approximation.

Value

A list of the overall and stagewise rejection probabilities, the futility stopping probabilities, the overall and stagewise expected number of events, number of patients, and analysis time, the input accrual and follow-up durations, and whether a fixed follow-up is used.

Examples

```
# Piecewise accrual, piecewise exponential survivals, and 5% dropout by
# the end of 1 year.

lrpower(kMax = 2, informationRates = c(0.8, 1),
    alpha = 0.025, typeAlphaSpending = "sfOF",
    allocationRatioPlanned = 1, accrualTime = seq(0, 9),
    accrualIntensity = c(26/9*seq(1, 9), 26),
    piecewiseSurvivalTime = c(0, 6),
    stratumFraction = c(0.2, 0.8),
    lambda1 = c(0.0533, 0.0309, 1.5*0.0533, 1.5*0.0309),
    lambda2 = c(0.0533, 0.0533, 1.5*0.0533, 1.5*0.0533),
    gamma1 = -log(1-0.05)/12,
    gamma2 = -log(1-0.05)/12, accrualDuration = 22,
    followupTime = 18, fixedFollowup = FALSE)
```

lrsamplesize   *Log-rank test sample size*

Description

Obtains the needed accrual duration given power and follow-up time, or the needed follow-up time given power and accrual duration in a two-group survival design.

Usage

```
lrsamplesize(
  beta = 0.2,
  kMax = NA_integer_,
  informationRates = NA_real_,
  efficacyStopping = NA_integer_,
  futilityStopping = NA_integer_,
  criticalValues = NA_real_,
  alpha = 0.025,
  typeAlphaSpending = "sfOF",
  parameterAlphaSpending = NA_real_,
  userAlphaSpending = NA_real_,
  futilityBounds = NA_real_,
  typeBetaSpending = "none",
  parameterBetaSpending = NA_real_,
  userBetaSpending = NA_real_,
```

```
  allocationRatioPlanned = 1,
  accrualTime = 0L,
  accrualIntensity = 20L,
  piecewiseSurvivalTime = 0L,
  stratumFraction = 1L,
  lambda1 = 0.0309,
  lambda2 = 0.0533,
  gamma1 = 0L,
  gamma2 = 0L,
  accrualDuration = NA_real_,
  followupTime = 18,
  fixedFollowup = 0L,
  rho1 = 0,
  rho2 = 0,
  numSubintervals = 300L,
  interval = as.numeric(c(0.001, 240))
)
```

Arguments

`beta` Type II error. Defaults to 0.2.

`kMax` The maximum number of stages.

`informationRates` The information rates in terms of number of events. Fixed prior to the trial. Defaults to `(1:kMax) / kMax` if left unspecified.

`efficacyStopping` Indicators of whether efficacy stopping is allowed at each stage. Defaults to true if left unspecified.

`futilityStopping` Indicators of whether futility stopping is allowed at each stage. Defaults to true if left unspecified.

`criticalValues` Upper boundaries on the z-test statistic scale for stopping for efficacy.

`alpha` The significance level. Defaults to 0.025.

`typeAlphaSpending` The type of alpha spending. One of the following: "OF" for O'Brien-Fleming boundaries, "P" for Pocock boundaries, "WT" for Wang & Tsiatis boundaries, "sfOF" for O'Brien-Fleming type spending function, "sfP" for Pocock type spending function, "sfKD" for Kim & DeMets spending function, "sfHSD" for Hwang, Shi & DeCani spending function, "user" for user defined spending, and "none" for no early efficacy stopping. Defaults to "sfOF".

`parameterAlphaSpending` The parameter value for the alpha spending. Corresponds to Delta for "WT", rho for "sfKD", and gamma for "sfHSD".

`userAlphaSpending` The user defined alpha spending. Cumulative alpha spent up to each stage.

`futilityBounds` Lower boundaries on the z-test statistic scale for stopping for futility at stages 1, ..., kMax-1. Defaults to `rep(-Inf,kMax-1)` if left unspecified.

`typeBetaSpending`
: The type of beta spending. One of the following: "sfOF" for O'Brien-Fleming type spending function, "sfP" for Pocock type spending function, "sfKD" for Kim & DeMets spending function, "sfHSD" for Hwang, Shi & DeCani spending function, "user" for user defined spending, and "none" for no early futility stopping. Defaults to "none".

`parameterBetaSpending`
: The parameter value for the beta spending. Corresponds to rho for "sfKD", and gamma for "sfHSD".

`userBetaSpending`
: The user defined beta spending. Cumulative beta spent up to each stage.

`allocationRatioPlanned`
: Allocation ratio for the active treatment versus control. Defaults to 1 for equal randomization.

`accrualTime`
: Accrual time intervals, must start with 0, e.g., `c(0,3)` breaks the time axis into 2 accrual intervals: [0, 3) and [3, Inf).

`accrualIntensity`
: A vector of accrual intensities, one for each accrual time interval.

`piecewiseSurvivalTime`
: A vector that specifies the time intervals for the piecewise exponential survival distribution, must start with 0, e.g., `c(0,6)` breaks the time axis into 2 event intervals: [0, 6) and [6, Inf). Defaults to 0 for exponential distribution.

`stratumFraction`
: A vector of stratum fractions that sum to 1. Defaults to 1 for no stratification.

`lambda1`
: A vector of hazard rates for the event for the active treatment group, one for each analysis time interval, by stratum.

`lambda2`
: A vector of hazard rates for the event for the control group, one for each analysis time interval, by stratum.

`gamma1`
: The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the active treatment group. Defaults to 0 for no dropout.

`gamma2`
: The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the control group. Defaults to 0 for no dropout.

`accrualDuration`
: Duration of the enrollment period.

`followupTime`
: Follow-up time for the last enrolled subject.

`fixedFollowup`
: Whether a fixed follow-up design is used. Defaults to 0 for variable follow-up.

`rho1`
: First parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

`rho2`
: Second parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

`numSubintervals`
: Number of sub-intervals to approximate the mean and variance of the weighted log-rank test score statistic. Defaults to 300. Specify a larger number for better approximation.

`interval` The interval to search for the solution of accrualDuration or followupDuration. Defaults to `c(0.001,240)`. Adjustment may be needed for non-monotone relationship with study power.

Value

A list of the overall and stagewise rejection probabilities, the futility stopping probabilities, the overall and stagewise expected number of events, number of patients, and analysis time, the input or calculated accrual and follow-up durations, and whether a fixed follow-up is used.

Examples

```
# Piecewise accrual, piecewise exponential survivals, and 5% dropout by
# the end of 1 year.

# Example 1: Obtains accrual duration given power and follow-up duration
lrsamplesize(beta = 0.2, kMax = 2,
      informationRates = c(0.8, 1),
      alpha = 0.025, typeAlphaSpending = "sfOF",
      accrualTime = seq(0, 9),
      accrualIntensity = c(26/9*seq(1, 9), 26),
      piecewiseSurvivalTime = c(0, 6),
      stratumFraction = c(0.2, 0.8),
      lambda1 = c(0.0533, 0.0309, 1.5*0.0533, 1.5*0.0309),
      lambda2 = c(0.0533, 0.0533, 1.5*0.0533, 1.5*0.0533),
      gamma1 = -log(1-0.05)/12,
      gamma2 = -log(1-0.05)/12,
      accrualDuration = NA,
      followupTime = 18, fixedFollowup = FALSE)

# Example 2: Obtains follow-up duration given power and accrual duration
lrsamplesize(beta = 0.2, kMax = 2,
      informationRates = c(0.8, 1),
      alpha = 0.025, typeAlphaSpending = "sfOF",
      accrualTime = seq(0, 9),
      accrualIntensity = c(26/9*seq(1, 9), 26),
      piecewiseSurvivalTime = c(0, 6),
      stratumFraction = c(0.2, 0.8),
      lambda1 = c(0.0533, 0.0309, 1.5*0.0533, 1.5*0.0309),
      lambda2 = c(0.0533, 0.0533, 1.5*0.0533, 1.5*0.0533),
      gamma1 = -log(1-0.05)/12,
      gamma2 = -log(1-0.05)/12,
      accrualDuration = 22,
      followupTime = NA, fixedFollowup = FALSE)
```

`lrsim` *Log-rank test simulation*

Description

Performs simulation for two-arm group sequential superiority trials based on log-rank test.

Usage

```
lrsim(
  kMax = NA_integer_,
  informationTime = NA_real_,
  criticalValues = NA_real_,
  futilityBounds = NA_real_,
  allocation1 = 1L,
  allocation2 = 1L,
  accrualTime = 0L,
  accrualIntensity = NA_real_,
  piecewiseSurvivalTime = 0L,
  stratumFraction = 1L,
  lambda1 = NA_real_,
  lambda2 = NA_real_,
  gamma1 = 0L,
  gamma2 = 0L,
  accrualDuration = NA_real_,
  followupTime = NA_real_,
  fixedFollowup = 0L,
  rho1 = 0,
  rho2 = 0,
  plannedEvents = NA_integer_,
  maxNumberOfIterations = 1000L,
  maxNumberOfRawDatasetsPerStage = 0L,
  seed = NA_integer_
)
```

Arguments

`kMax` The maximum number of stages.

`informationTime` Information time in terms of variance of weighted log-rank test score statistic. Same as informationRates in terms of number of events for unweighted log-rank test. Fixed prior to the trial. Defaults to `(1:kMax) / kMax` if left unspecified.

`criticalValues` Upper boundaries on the z-test statistic scale for stopping for efficacy.

`futilityBounds` Lower boundaries on the z-test statistic scale for stopping for futility at stages 1, ..., kMax-1. Defaults to `rep(-Inf,kMax-1)` if left unspecified.

`allocation1` Number of subjects in the active treatment group in a randomization block. Defaults to 1 for equal randomization.

`allocation2` Number of subjects in the control group in a randomization block. Defaults to 1 for equal randomization.

`accrualTime` Accrual time intervals, must start with 0, e.g., `c(0,3)` breaks the time axis into 2 accrual intervals: [0, 3) and [3, Inf).

accrualIntensity
: A vector of accrual intensities, one for each accrual time interval.

piecewiseSurvivalTime
: A vector that specifies the time intervals for the piecewise exponential survival distribution, must start with 0, e.g., `c(0,6)` breaks the time axis into 2 event intervals: [0, 6) and [6, Inf). Defaults to 0 for exponential distribution.

stratumFraction
: A vector of stratum fractions that sum to 1. Defaults to 1 for no stratification.

lambda1
: A vector of hazard rates for the event for the active treatment group, one for each analysis time interval, by stratum.

lambda2
: A vector of hazard rates for the event for the control group, one for each analysis time interval, by stratum.

gamma1
: The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the active treatment group. Defaults to 0 for no dropout.

gamma2
: The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the control group. Defaults to 0 for no dropout.

accrualDuration
: Duration of the enrollment period.

followupTime
: Follow-up time for the last enrolled subject.

fixedFollowup
: Whether a fixed follow-up design is used. Defaults to 0 for variable follow-up.

rho1
: First parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

rho2
: Second parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

plannedEvents
: The planned cumulative total number of events at each stage.

maxNumberOfIterations
: The number of simulation iterations. Defaults to 1000.

maxNumberOfRawDatasetsPerStage
: The number of raw datasets per stage to extract. Defaults to 1.

seed
: The seed to reproduce the simulation results. The computer clock will be used if left unspecified,

Value

A list of S3 class lrsim with 3 components: overview is a list of the operating characteristics of the design, sumdata is a data frame for the summary data for each iteration, and rawdata is a data frame for selected raw data if maxNumberOfRawDatasetsPerStage is a positive integer.

Examples

```
sim = lrsim(kMax = 2, informationTime = c(0.5, 1),
            criticalValues = c(2.797, 1.977),
            accrualIntensity = 11,
            lambda1 = 0.018, lambda2 = 0.030,
            accrualDuration = 12,
            plannedEvents = c(60, 120),
```

```
            maxNumberOfIterations = 1000,
            maxNumberOfRawDatasetsPerStage = 1,
            seed = 314159)

# summary statistics
sim

# summary for each simulated data set
head(sim$sumdata)

# raw data for selected replication
head(sim$rawdata)
```

---

lrstat *Number of subjects having an event and log-rank statistics*

Description

Obtains the number of subjects having an event in each treatment group, the mean and variance of weighted log-rank test score statistic at given calendar times.

Usage

```
lrstat(
  time = NA_real_,
  allocationRatioPlanned = 1,
  accrualTime = 0L,
  accrualIntensity = NA_real_,
  piecewiseSurvivalTime = 0L,
  stratumFraction = 1L,
  lambda1 = NA_real_,
  lambda2 = NA_real_,
  gamma1 = 0L,
  gamma2 = 0L,
  accrualDuration = NA_real_,
  followupTime = NA_real_,
  fixedFollowup = 0L,
  rho1 = 0,
  rho2 = 0,
  numSubintervals = 300L,
  predictEventOnly = 0L
)
```

Arguments

time	A vector of calendar times at which to calculate the number of events and the mean and variance of log-rank test score statistic.

`allocationRatioPlanned`
: Allocation ratio for the active treatment versus control. Defaults to 1 for equal randomization.

`accrualTime`
: Accrual time intervals, must start with 0, e.g., `c(0,3)` breaks the time axis into 2 accrual intervals: [0, 3) and [3, Inf).

`accrualIntensity`
: A vector of accrual intensities, one for each accrual time interval.

`piecewiseSurvivalTime`
: A vector that specifies the time intervals for the piecewise exponential survival distribution, must start with 0, e.g., `c(0,6)` breaks the time axis into 2 event intervals: [0, 6) and [6, Inf). Defaults to 0 for exponential distribution.

`stratumFraction`
: A vector of stratum fractions that sum to 1. Defaults to 1 for no stratification.

`lambda1`
: A vector of hazard rates for the event for the active treatment group, one for each analysis time interval, by stratum.

`lambda2`
: A vector of hazard rates for the event for the control group, one for each analysis time interval, by stratum.

`gamma1`
: The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the active treatment group. Defaults to 0 for no dropout.

`gamma2`
: The hazard rate for exponential dropout or a vector of hazard rates for piecewise exponential dropout for the control group. Defaults to 0 for no dropout.

`accrualDuration`
: Duration of the enrollment period.

`followupTime`
: Follow-up time for the last enrolled subject.

`fixedFollowup`
: Whether a fixed follow-up design is used. Defaults to 0 for variable follow-up.

`rho1`
: First parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

`rho2`
: Second parameter of the Fleming-Harrington family of weighted log-rank test. Defaults to 0 for conventional log-rank test.

`numSubintervals`
: Number of sub-intervals to approximate the mean and variance of the weighted log-rank test score statistic. Defaults to 300. Specify a larger number for better approximation.

`predictEventOnly`
: Whether to predict the number of events only. Defaults to 0 for obtaining log-rank test score statistic mean and variance.

Value

A data frame of the number of patients enrolled, the number of patients having an event overall and in each treatment group, the mean and variance of weighted log-rank test score statistic at the specified calendar times by stratum.

Examples

```
# Piecewise accrual, piecewise exponential survivals, and 5% dropout by
# the end of 1 year.
lrstat(time = c(22, 40), allocationRatioPlanned = 1,
   accrualTime = seq(0, 9),
   accrualIntensity = c(26/9*seq(1, 9), 26),
   piecewiseSurvivalTime = c(0, 6),
   stratumFraction = c(0.2, 0.8),
   lambda1 = c(0.0533, 0.0309, 1.5*0.0533, 1.5*0.0309),
   lambda2 = c(0.0533, 0.0533, 1.5*0.0533, 1.5*0.0533),
   gamma1 = -log(1-0.05)/12,
   gamma2 = -log(1-0.05)/12,
   accrualDuration = 22,
   followupTime = 18, fixedFollowup = FALSE)
```

Index