

Package ‘rsimsum’

April 25, 2019

Version 0.5.2

Title Analysis of Simulation Studies Including Monte Carlo Error

Description Summarise results from simulation studies and compute Monte Carlo standard errors of commonly used summary statistics. This package is modelled on the 'simsum' user-written command in 'Stata' (See White I.R., 2010 <<http://www.stata-journal.com/article.html?article=st0200>>).

License GPL (>= 3)

Depends R (>= 2.10)

Imports checkmate, ggridges, ggplot2, rlang, stats

Suggests covr, devtools, dplyr, eha, knitr, rmarkdown, rstpm2, survival, testthat, usethis, viridis

URL <https://ellessenne.github.io/rsimsum/>

BugReports <https://github.com/ellessenne/rsimsum/issues>

VignetteBuilder knitr

RoxygenNote 6.1.1

LazyData true

ByteCompile true

Encoding UTF-8

Language en-GB

NeedsCompilation no

Author Alessandro Gasparini [aut, cre]
(<<https://orcid.org/0000-0002-8319-7624>>),
Ian R. White [aut]

Maintainer Alessandro Gasparini <ag475@leicester.ac.uk>

Repository CRAN

Date/Publication 2019-04-25 17:40:03 UTC

R topics documented:

autoplot.multisimsum	2
autoplot.simsum	3
autoplot.summary.multisimsum	4
autoplot.summary.simsum	6
dropbig	7
frailty	8
get_data	9
is.multisimsum	10
is.simsum	10
is.summary.multisimsum	11
is.summary.simsum	11
MIsim	11
multisimsum	12
nsim	14
print.multisimsum	15
print.simsum	16
print.summary.multisimsum	16
print.summary.simsum	17
relhaz	18
rsimsum	19
simsum	20
summary.multisimsum	22
summary.simsum	23
Index	25

autoplot.multisimsum *autoplot method for multisimsum objects*

Description

autoplot can produce a series of plot to summarise results of simulation studies. See vignette("plotting", package = "autoplot") for further details.

Usage

```
## S3 method for class 'multisimsum'
autoplot(object, par, type = "forest",
  stats = "bias", target = NULL, fitted = TRUE, scales = "fixed",
  ...)
```

Arguments

object	An object of class <code>multisimsum</code> .
par	The parameter results to plot.
type	The type of the plot to be produced. Possible choices are: <code>forest</code> , <code>lolly</code> , <code>zip</code> , <code>est</code> , <code>se</code> , <code>est_ba</code> , <code>se_ba</code> , <code>est_ridge</code> , <code>se_ridge</code> , with <code>forest</code> being the default.
stats	Summary statistic to plot, defaults to <code>bias</code> . See <code>summary.simsum()</code> for further details on supported summary statistics.
target	Target of summary statistic, e.g. 0 for <code>bias</code> . Defaults to <code>NULL</code> , in which case target will be inferred.
fitted	Superimpose a fitted regression line, useful when <code>type = (est, se, est_ba, se_ba)</code> . Defaults to <code>TRUE</code> .
scales	Should scales be fixed (<code>fixed</code> , the default), free (<code>free</code>), or free in one dimension (<code>free_x</code> , <code>free_y</code>)?
...	Not used.

Value

A `ggplot` object.

Examples

```
data("frailty", package = "rsimsum")
ms <- multisimsum(
  data = frailty,
  par = "par", true = c(trt = -0.50, fv = 0.75),
  estvarname = "b", se = "se", methodvar = "model",
  by = "fv_dist", x = TRUE
)

library(ggplot2)
autoplot(ms, par = "trt")
autoplot(ms, par = "trt", type = "lolly", stats = "cover")
autoplot(ms, par = "trt", type = "zip")
autoplot(ms, par = "trt", type = "est_ba")
```

autoplot.simsum

autoplot method for simsum objects

Description

`autoplot` can produce a series of plot to summarise results of simulation studies. See `vignette("plotting", package = "rsimsum")` for further details.

Usage

```
## S3 method for class 'simsum'
autoplot(object, type = "forest", stats = "bias",
         target = NULL, fitted = TRUE, scales = "fixed", ...)
```

Arguments

object	An object of class simsum.
type	The type of the plot to be produced. Possible choices are: forest, lolly, zip, est, se, est_ba, se_ba, est_ridge, se_ridge, heat, with forest being the default.
stats	Summary statistic to plot, defaults to bias. See summary.simsum() for further details on supported summary statistics.
target	Target of summary statistic, e.g. 0 for bias. Defaults to NULL, in which case target will be inferred.
fitted	Superimpose a fitted regression line, useful when type = (est, se, est_ba, se_ba). Defaults to TRUE.
scales	Should scales be fixed (fixed, the default), free (free), or free in one dimension (free_x, free_y)?
...	Not used.

Value

A ggplot object.

Examples

```
data("MIsim", package = "rsimsum")
s <- rsimsum::simsum(
  data = MIsim, estvarname = "b", true = 0.5, se = "se",
  methodvar = "method", x = TRUE
)

library(ggplot2)
autoplot(s)
autoplot(s, type = "lolly")
```

autoplot.summary.multisumsum

autoplot method for summary.multisumsum objects

Description

autoplot method for summary.multisumsum objects

Usage

```
## S3 method for class 'summary.multisimsum'
autoplot(object, par, type = "forest",
  stats = "bias", target = NULL, fitted = TRUE, scales = "fixed",
  ...)
```

Arguments

object	An object of class <code>summary.multisimsum</code> .
par	The parameter results to plot.
type	The type of the plot to be produced. Possible choices are: <code>forest</code> , <code>lolly</code> , <code>zip</code> , <code>est</code> , <code>se</code> , <code>est_ba</code> , <code>se_ba</code> , <code>est_ridge</code> , <code>se_ridge</code> , with <code>forest</code> being the default.
stats	Summary statistic to plot, defaults to <code>bias</code> . See <code>summary.simsum()</code> for further details on supported summary statistics.
target	Target of summary statistic, e.g. 0 for <code>bias</code> . Defaults to <code>NULL</code> , in which case target will be inferred.
fitted	Superimpose a fitted regression line, useful when <code>type = (est, se, est_ba, se_ba)</code> . Defaults to <code>TRUE</code> .
scales	Should scales be fixed (<code>fixed</code> , the default), free (<code>free</code>), or free in one dimension (<code>free_x</code> , <code>free_y</code>)?
...	Not used.

Value

A `ggplot` object.

Examples

```
data("frailty", package = "rsimsum")
ms <- multisimsum(
  data = frailty,
  par = "par", true = c(trt = -0.50, fv = 0.75),
  estvarname = "b", se = "se", methodvar = "model",
  by = "fv_dist", x = TRUE
)
sms <- summary(ms)

library(ggplot2)
autoplot(sms, par = "trt")
autoplot(sms, par = "trt", type = "lolly", stats = "cover")
autoplot(sms, par = "trt", type = "zip")
autoplot(sms, par = "trt", type = "est_ba")
```

 autoplot.summary.simsum

autoplot method for summary.simsum objects

Description

autoplot method for summary.simsum objects

Usage

```
## S3 method for class 'summary.simsum'
autoplot(object, type = "forest",
  stats = "bias", target = NULL, fitted = TRUE, scales = "fixed",
  ...)
```

Arguments

object	An object of class <code>summary.simsum</code> .
type	The type of the plot to be produced. Possible choices are: <code>forest</code> , <code>lolly</code> , <code>zip</code> , <code>est</code> , <code>se</code> , <code>est_ba</code> , <code>se_ba</code> , <code>est_ridge</code> , <code>se_ridge</code> , <code>heat</code> , with <code>forest</code> being the default.
stats	Summary statistic to plot, defaults to <code>bias</code> . See <code>summary.simsum()</code> for further details on supported summary statistics.
target	Target of summary statistic, e.g. 0 for <code>bias</code> . Defaults to <code>NULL</code> , in which case target will be inferred.
fitted	Superimpose a fitted regression line, useful when <code>type = (est, se, est_ba, se_ba)</code> . Defaults to <code>TRUE</code> .
scales	Should scales be fixed (<code>fixed</code> , the default), free (<code>free</code>), or free in one dimension (<code>free_x</code> , <code>free_y</code>)?
...	Not used.

Value

A ggplot object.

Examples

```
data("MIsim", package = "rsimsum")
s <- rsimsum::simsum(
  data = MIsim, estvarname = "b", true = 0.5, se = "se",
  methodvar = "method", x = TRUE
)
ss <- summary(s)

library(ggplot2)
autoplot(ss)
autoplot(ss, type = "lolly")
```

`dropbig`*Identify replications with large point estimates, standard errors*

Description

`dropbig` is useful to identify replications with large point estimates or standard errors. Large values are defined as standardised values above a given threshold, as defined when calling `dropbig`. Regular standardisation using mean and standard deviation is implemented, as well as robust standardisation using median and inter-quartile range. Further to that, the standardisation process is stratified by data-generating mechanism if by factors are defined.

Usage

```
dropbig(data, estvarname, se, methodvar = NULL, by = NULL, max = 10,
        semax = 100, robust = TRUE)
```

Arguments

<code>data</code>	A <code>data.frame</code> in which variable names are interpreted. It has to be in tidy format, e.g. each variable forms a column and each observation forms a row.
<code>estvarname</code>	The name of the variable containing the point estimates.
<code>se</code>	The name of the variable containing the standard errors of the point estimates.
<code>methodvar</code>	The name of the variable containing the methods to compare. For instance, methods could be the models compared within a simulation study. Can be <code>NULL</code> .
<code>by</code>	A vector of variable names to compute performance measures by a list of factors. Factors listed here are the (potentially several) data-generating mechanisms used to simulate data under different scenarios (e.g. sample size, true distribution of a variable, etc.). Can be <code>NULL</code> .
<code>max</code>	Specifies the maximum acceptable absolute value of the point estimates, after standardisation. Defaults to 10.
<code>semax</code>	Specifies the maximum acceptable absolute value of the standard error, after standardisation. Defaults to 100.
<code>robust</code>	Specifies whether to use robust standardisation (using median and inter-quartile range) rather than normal standardisation (using mean and standard deviation). Defaults to <code>TRUE</code> .

Value

The same `data.frame` given as input with an additional column named `.dropbig` identifying rows that are classified as large (`.dropbig = TRUE`) according to the specified criterion.

Examples

```

data("frailty", package = "rsimsum")
frailty2 <- subset(frailty, par == "fv")

# Using low values of max, semax for illustration purposes:
dropbig(
  data = frailty2, estvarname = "b", se = "se",
  methodvar = "model", by = "fv_dist", max = 2, semax = 2
)

# Using regular standardisation:
dropbig(
  data = frailty2, estvarname = "b", se = "se",
  methodvar = "model", by = "fv_dist", max = 2, semax = 2, robust = FALSE
)

```

frailty

*Example of a simulation study on frailty survival models***Description**

A dataset from a simulation study comparing frailty flexible parametric models fitted using penalised likelihood to semiparametric frailty models. Both models are fitted assuming a Gamma and a log-Normal frailty. One thousand datasets were simulated, each containing a binary treatment variable with a log-hazard ratio of -0.50. Clustered survival data was simulated assuming 50 clusters of 50 individuals each, with a mixture Weibull baseline hazard function and a frailty following either a Gamma or a Log-Normal distribution. The comparison involves estimates of the log-treatment effect, and estimates of heterogeneity (i.e. the estimated frailty variance).

Usage

frailty

Format

A data frame with 16,000 rows and 6 variables:

- i Simulated dataset number.
- b Point estimate.
- se Standard error of the point estimate.
- par The estimand. trt is the log-treatment effect, fv is the variance of the frailty.
- fv_dist The true frailty distribution.
- model Method used (Cox, Gamma, Cox, Log-Normal, RP(P), Gamma, or RP(P), Log-Normal).

Examples

```

data("frailty", package = "rsimsum")

```

get_data	<i>get_data</i>
----------	-----------------

Description

Extract data slots from an object of class `simsum`, `summary.simsum`, `multisimsum`, or `summary.multisimsum`.

Usage

```
get_data(x, stats = NULL, ...)
```

Arguments

<code>x</code>	An object of class <code>simsum</code> .
<code>stats</code>	Summary statistics to include; can be a scalar value or a vector. Possible choices are: <ul style="list-style-type: none"> • <code>nsim</code>, the number of replications with non-missing point estimates and standard error. • <code>thetamean</code>, average point estimate. • <code>thetamedian</code>, median point estimate. • <code>se2mean</code>, average standard error. • <code>se2median</code>, median standard error. • <code>bias</code>, bias in point estimate. • <code>empse</code>, empirical standard error. • <code>mse</code>, mean squared error. • <code>relprec</code>, percentage gain in precision relative to the reference method. • <code>modelse</code>, model-based standard error. • <code>relerror</code>, relative percentage error in standard error. • <code>cover</code>, coverage of a nominal <code>level%</code> confidence interval. • <code>becover</code>, bias-eliminated coverage of a nominal <code>level%</code> confidence interval. • <code>power</code>, power of a $(1 - \text{level}\%)$ level test. Defaults to <code>NULL</code>, in which case all summary statistics are returned.
<code>...</code>	Ignored.

Value

A `data.frame` containing summary statistics from a simulation study.

Examples

```

data(MIsim)
x <- simsum(
  data = MIsim, estvarname = "b", true = 0.5, se = "se",
  methodvar = "method"
)
get_data(x)

# Extracting only bias and coverage:
get_data(x, stats = c("bias", "cover"))

xs <- summary(x)
get_data(xs)

```

is.multisimsum	<i>is.multisimsum</i>
----------------	-----------------------

Description

Reports whether x is a multisimsum object

Usage

```
is.multisimsum(x)
```

Arguments

x	An object to test.
---	--------------------

is.simsum	<i>is.simsum</i>
-----------	------------------

Description

Reports whether x is a simsum object

Usage

```
is.simsum(x)
```

Arguments

x	An object to test.
---	--------------------

```
is.summary.multisimsum
      is.summary.multisimsum
```

Description

Reports whether `x` is a `summary.multisimsum` object

Usage

```
is.summary.multisimsum(x)
```

Arguments

`x` An object to test.

```
is.summary.simsum      is.summary.simsum
```

Description

Reports whether `x` is a `summary.simsum` object

Usage

```
is.summary.simsum(x)
```

Arguments

`x` An object to test.

```
MIsim      Example of a simulation study on missing data
```

Description

A dataset from a simulation study comparing different ways to handle missing covariates when fitting a Cox model (White and Royston, 2009). One thousand datasets were simulated, each containing normally distributed covariates x and z and time-to-event outcome. Both covariates has 20% of their values deleted independently of all other variables so the data became missing completely at random (Little and Rubin, 2002). Each simulated dataset was analysed in three ways. A Cox model was fit to the complete cases (CC). Then two methods of multiple imputation using chained equations (van Buuren, Boshuizen, and Knook, 1999) were used. The MI_LOGT method multiply imputes the missing values of x and z with the outcome included as $\log(t)$ and d , where t is the survival time and d is the event indicator. The MI_T method is the same except that $\log(t)$ is replaced by t in the imputation model. The results are stored in long format.

Usage

```
MIsim
```

Format

A data frame with 3,000 rows and 4 variables:

- dataset Simulated dataset number.
- method Method used (CC, MI_LOGT or MI_T).
- b Point estimate.
- se Standard error of the point estimate.

References

White, I.R., and P. Royston. 2009. Imputing missing covariate values for the Cox model. *Statistics in Medicine* 28(15):1982-1998 doi: [10.1002/sim.3618](https://doi.org/10.1002/sim.3618)

Little, R.J.A., and D.B. Rubin. 2002. *Statistical analysis with missing data*. 2nd ed. Hoboken, NJ: Wiley doi: [10.1002/9781119013563](https://doi.org/10.1002/9781119013563)

Examples

```
data("MIsim", package = "rsimsum")
```

multisimsum

Analyses of simulation studies with multiple estimands at once, including Monte Carlo error

Description

multisimsum is an extension of `simsum()` that can handle multiple estimated parameters at once. multisimsum calls `simsum()` internally, each estimands at once. There is only one new argument that must be set when calling multisimsum: `par`, a string representing the column of data that identifies the different estimands. Additionally, with multisimsum the argument `true` must be a named vector, where names correspond to each estimand (see examples).

Usage

```
multisimsum(data, par, estvarname, true, se, methodvar = NULL,
  ref = NULL, by = NULL, ci.limits = NULL, dropbig = FALSE,
  x = FALSE, control = list())
```

Arguments

<code>data</code>	A <code>data.frame</code> in which variable names are interpreted. It has to be in tidy format, e.g. each variable forms a column and each observation forms a row.
<code>par</code>	The name of the variable containing the methods to compare. Can be <code>NULL</code> .
<code>estvarname</code>	The name of the variable containing the point estimates.
<code>true</code>	The true value of the parameter. This is used in calculations of bias and coverage.
<code>se</code>	The name of the variable containing the standard errors of the point estimates.
<code>methodvar</code>	The name of the variable containing the methods to compare. For instance, methods could be the models compared within a simulation study. Can be <code>NULL</code> .
<code>ref</code>	Specifies the reference method against which relative precision will be calculated. Only useful if <code>methodvar</code> is specified.
<code>by</code>	A vector of variable names to compute performance measures by a list of factors. Factors listed here are the (potentially several) data-generating mechanisms used to simulate data under different scenarios (e.g. sample size, true distribution of a variable, etc.). Can be <code>NULL</code> .
<code>ci.limits</code>	A numeric vector of length 2 specifying the limits (lower and upper) of confidence intervals used to calculate coverage. Useful for non-Wald type estimators (e.g. bootstrap). Defaults to <code>NULL</code> , where Wald-type confidence intervals based on the provided SEs are calculated for coverage. This feature is experimental, use with caution.
<code>dropbig</code>	Specifies that point estimates or standard errors beyond the maximum acceptable values should be dropped. Defaults to <code>FALSE</code> .
<code>x</code>	Set to <code>TRUE</code> to include the data argument used to calculate summary statistics (i.e. after pre-processing the input dataset e.g. removing values deemed too large via the <code>dropbig</code> argument) as a slot. Calling <code>simsum</code> with <code>x = TRUE</code> is required to produce zipper plots. The downside is that the size of the returned object increases considerably, therefore it is set to <code>FALSE</code> by default.
<code>control</code>	A list of parameters that control the behaviour of <code>simsum</code> . Possible values are: <ul style="list-style-type: none"> • <code>mcse</code>, whether to calculate Monte Carlo standard errors. Defaults to <code>TRUE</code>; • <code>level</code>, the significance level used for coverage, bias-eliminated coverage, and power. Defaults to 0.95; • <code>df</code>, whether to use robust critical values from a t distribution with <code>df</code> degrees of freedom when calculating coverage, bias-eliminated coverage, and power. Defaults to <code>NULL</code>, in which case a Gaussian distribution is used; • <code>na.rm</code>, whether to remove point estimates or standard errors where either (or both) is missing. Defaults to <code>TRUE</code>; • <code>char.sep</code>, a character utilised when splitting the input dataset data. Generally, this should not be changed; • <code>dropbig.max</code>, specifies the maximum acceptable absolute value of the point estimates, after standardisation. Defaults to 10; • <code>dropbig.semax</code>, specifies the maximum acceptable absolute value of the standard error, after standardisation. Defaults to 100

- `dropbig.robust`, specifies whether to use robust standardisation (using median and inter-quartile range) rather than normal standardisation (using mean and standard deviation). Defaults to TRUE, in which case robust standardisation will be used for `dropbig`.

Details

The following names are not allowed for `estvarname`, `se`, `methodvar`, `by`, `par`: `stat`, `est`, `mcse`, `lower`, `upper`.

Value

An object of class `multisimsum`.

Examples

```
data("frailty", package = "rsimsum")
ms <- multisimsum(
  data = frailty,
  par = "par", true = c(trt = -0.50, fv = 0.75),
  estvarname = "b", se = "se", methodvar = "model",
  by = "fv_dist"
)
ms
```

nsim

Compute number of simulations required

Description

The function `nsim` computes the number of simulations B to perform based on the accuracy of an estimate of interest, using the following equation:

$$B = \left(\frac{(Z_{1-\alpha/2} + Z_{1-\theta})\sigma}{\delta} \right)^2,$$

where δ is the specified level of accuracy of the estimate of interest you are willing to accept (i.e. the permissible difference from the true value β), $Z_{1-\alpha/2}$ is the $(1 - \alpha/2)$ quantile of the standard normal distribution, $Z_{1-\theta}$ is the $(1 - \theta)$ quantile of the standard normal distribution with $(1 - \theta)$ being the power to detect a specific difference from the true value as significant, and σ^2 is the variance of the parameter of interest.

Usage

```
nsim(alpha, sigma, delta, power = 0.5)
```

Arguments

alpha	Significance level. Must be a value between 0 and 1.
sigma	Variance for the parameter of interest. Must be greater than 0.
delta	Specified level of accuracy of the estimate of interest you are willing to accept. Must be greater than 0.
power	Power to detect a specific difference from the true value as significant. Must be a value between 0 and 1. Defaults to 0.5, e.g. a power of 50%.

Value

A scalar value B representing the number of simulations to perform based on the accuracy required.

References

Burton, A., Douglas G. Altman, P. Royston. et al. 2006. The design of simulation studies in medical statistics. *Statistics in Medicine* 25: 4279-4292 doi: [10.1002/sim.2673](https://doi.org/10.1002/sim.2673)

Examples

```
# Number of simulations required to produce an estimate to within 5%
# accuracy of the true coefficient of 0.349 with a 5% significance level,
# assuming the variance of the estimate is 0.0166 and 50% power:
nsim(alpha = 0.05, sigma = sqrt(0.0166), delta = 0.349 * 5 / 100, power = 0.5)

# Number of simulations required to produce an estimate to within 1%
# accuracy of the true coefficient of 0.349 with a 5% significance level,
# assuming the variance of the estimate is 0.0166 and 50% power:
nsim(alpha = 0.05, sigma = sqrt(0.0166), delta = 0.349 * 1 / 100, power = 0.5)
```

```
print.multisimsum      print.multisimsum
```

Description

Print method for multisimsum objects

Usage

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

Arguments

x	An object of class multisimsum.
...	Ignored.

Examples

```

data(frailty)
ms <- multisimsum(
  data = frailty, par = "par", true = c(
    trt = -0.50,
    fv = 0.75
  ), estvarname = "b", se = "se", methodvar = "model",
  by = "fv_dist"
)
ms

```

```
print.simsum
```

```
print.simsum
```

Description

Print method for simsum objects

Usage

```

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

```

Arguments

<code>x</code>	An object of class simsum.
<code>...</code>	Ignored.

Examples

```

data("MIsim")
x <- simsum(
  data = MIsim, estvarname = "b", true = 0.5, se = "se",
  methodvar = "method"
)
x

```

```
print.summary.multisimsum
```

```
print.summary.multisimsum
```

Description

Print method for summary.multisimsum objects

Usage

```
## S3 method for class 'summary.multisimsum'
print(x, digits = 4, mcse = TRUE, ...)
```

Arguments

<code>x</code>	An object of class <code>summary.multisimsum</code> .
<code>digits</code>	Number of significant digits used for printing. Defaults to 4.
<code>mcse</code>	Should Monte Carlo standard errors be reported? If <code>mcse = FALSE</code> , confidence intervals based on Monte Carlo standard errors will be reported instead, see summary.multisimsum() . If a NULL value is passed, only point estimates are printed regardless of whether Monte Carlo standard errors were computed or not. Defaults to TRUE.
<code>...</code>	Ignored.

Examples

```
data(frailty)
ms <- multisimsum(
  data = frailty, par = "par", true = c(
    trt = -0.50,
    fv = 0.75
  ), estvarname = "b", se = "se", methodvar = "model",
  by = "fv_dist"
)
sms <- summary(ms, stats = c("bias", "cover", "mse"))
sms

# Printing less significant digits:
print(sms, digits = 3)

# Printing confidence intervals:
print(sms, digits = 3, mcse = FALSE)

# Printing values only:
print(sms, mcse = NULL)
```

```
print.summary.simsum  print.summary.simsum
```

Description

Print method for `summary.simsum` objects

Usage

```
## S3 method for class 'summary.simsum'
print(x, digits = 4, mcse = TRUE, ...)
```

Arguments

<code>x</code>	An object of class <code>summary.simsum</code> .
<code>digits</code>	Number of significant digits used for printing. Defaults to 4.
<code>mcse</code>	Should Monte Carlo standard errors be reported? If <code>mcse = FALSE</code> , confidence intervals based on Monte Carlo standard errors will be reported instead, see summary.simsum() . If a <code>NULL</code> value is passed, only point estimates are printed regardless of whether Monte Carlo standard errors were computed or not. Defaults to <code>TRUE</code> .
<code>...</code>	Ignored.

Examples

```

data("MIsim")
x <- simsum(
  data = MIsim, estvarname = "b", true = 0.5, se = "se",
  methodvar = "method"
)
xs <- summary(x)
xs

# Printing less significant digits:
print(xs, digits = 2)

# Printing confidence intervals:
print(xs, mcse = FALSE)

# Printing values only:
print(xs, mcse = NULL)

```

relhaz

Example of a simulation study on survival modelling

Description

A dataset from a simulation study assessing the impact of misspecifying the baseline hazard in survival models on regression coefficients. One thousand datasets were simulated, each containing a binary treatment variable with a log-hazard ratio of -0.50. Survival data was simulated for two different sample sizes, 50 and 250 individuals, and under two different baseline hazard functions, exponential and Weibull. Consequently, a Cox model (Cox, 1972), a fully parametric exponential model, and a Royston-Parmar (Royston and Parmar, 2002) model with two degrees of freedom were fit to each simulated dataset. See `vignette("relhaz", package = "rsimsum")` for more information.

Usage

pre>relhaz

Format

A data frame with 1,200 rows and 6 variables:

- `dataset` Simulated dataset number.
- `n` Sample size of the simulate dataset.
- `baseline` Baseline hazard function of the simulated dataset.
- `model` Method used (Cox, Exp, or RP(2)).
- `theta` Point estimate for the log-hazard ratio.
- `se` Standard error of the point estimate.

References

Cox D.R. 1972. Regression models and life-tables. *Journal of the Royal Statistical Society, Series B (Methodological)* 34(2):187-220. doi: [10.1007/9781461243809_37](https://doi.org/10.1007/9781461243809_37)

Royston, P. and Parmar, M.K. 2002. Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. *Statistics in Medicine* 21(15):2175-2197 doi: [10.1002/sim.1203](https://doi.org/10.1002/sim.1203)

Examples

```
data("relhaz", package = "rsimsum")
```

rsimsum

Analysis of Simulation Studies Including Monte Carlo Error

Description

Summarise results from simulation studies and compute Monte Carlo standard errors of commonly used summary statistics. This package is modelled on the 'simsum' user-written command in 'Stata' (See White I.R., 2010 <http://www.stata-journal.com/article.html?article=st0200>).

Author(s)

Alessandro Gasparini (ag475@leicester.ac.uk)

Description

simsum computes performance measures for simulation studies in which each simulated data set yields point estimates by one or more analysis methods. Bias, empirical standard error and precision relative to a reference method can be computed for each method. If, in addition, model-based standard errors are available then simsum can compute the average model-based standard error, the relative error in the model-based standard error, the coverage of nominal confidence intervals, the coverage under the assumption that there is no bias (bias-eliminated coverage), and the power to reject a null hypothesis. Monte Carlo errors are available for all estimated quantities.

Usage

```
simsum(data, estvarname, true, se, methodvar = NULL, ref = NULL,
       by = NULL, ci.limits = NULL, dropbig = FALSE, x = FALSE,
       control = list())
```

Arguments

data	A data.frame in which variable names are interpreted. It has to be in tidy format, e.g. each variable forms a column and each observation forms a row.
estvarname	The name of the variable containing the point estimates.
true	The true value of the parameter. This is used in calculations of bias and coverage.
se	The name of the variable containing the standard errors of the point estimates.
methodvar	The name of the variable containing the methods to compare. For instance, methods could be the models compared within a simulation study. Can be NULL.
ref	Specifies the reference method against which relative precision will be calculated. Only useful if methodvar is specified.
by	A vector of variable names to compute performance measures by a list of factors. Factors listed here are the (potentially several) data-generating mechanisms used to simulate data under different scenarios (e.g. sample size, true distribution of a variable, etc.). Can be NULL.
ci.limits	A numeric vector of length 2 specifying the limits (lower and upper) of confidence intervals used to calculate coverage. Useful for non-Wald type estimators (e.g. bootstrap). Defaults to NULL, where Wald-type confidence intervals based on the provided SEs are calculated for coverage. This feature is experimental, use with caution.
dropbig	Specifies that point estimates or standard errors beyond the maximum acceptable values should be dropped. Defaults to FALSE.

- `x` Set to TRUE to include the data argument used to calculate summary statistics (i.e. after pre-processing the input dataset e.g. removing values deemed too large via the `dropbig` argument) as a slot. Calling `simsum` with `x = TRUE` is required to produce zipper plots. The downside is that the size of the returned object increases considerably, therefore it is set to FALSE by default.
- `control` A list of parameters that control the behaviour of `simsum`. Possible values are:
- `mcse`, whether to calculate Monte Carlo standard errors. Defaults to TRUE;
 - `level`, the significance level used for coverage, bias-eliminated coverage, and power. Defaults to 0.95;
 - `df`, whether to use robust critical values from a t distribution with `df` degrees of freedom when calculating coverage, bias-eliminated coverage, and power. Defaults to NULL, in which case a Gaussian distribution is used;
 - `na.rm`, whether to remove point estimates or standard errors where either (or both) is missing. Defaults to TRUE;
 - `char.sep`, a character utilised when splitting the input dataset data. Generally, this should not be changed;
 - `dropbig.max`, specifies the maximum acceptable absolute value of the point estimates, after standardisation. Defaults to 10;
 - `dropbig.semax`, specifies the maximum acceptable absolute value of the standard error, after standardisation. Defaults to 100
 - `dropbig.robust`, specifies whether to use robust standardisation (using median and inter-quartile range) rather than normal standardisation (using mean and standard deviation). Defaults to TRUE, in which case robust standardisation will be used for `dropbig`.

Details

The following names are not allowed for `estvarname`, `se`, `methodvar`, `by`: `stat`, `est`, `mcse`, `lower`, `upper`.

Value

An object of class `simsum`.

References

White, I.R. 2010. `simsum`: Analyses of simulation studies including Monte Carlo error. *The Stata Journal* 10(3): 369-385. <http://www.stata-journal.com/article.html?article=st0200>

Morris, T.P., White, I.R. and Crowther, M.J. 2019. *Using simulation studies to evaluate statistical methods*. *Statistics in Medicine*, doi: [10.1002/sim.8086](https://doi.org/10.1002/sim.8086)

Gasparini, A. 2018. `rsimsum`: Summarise results from Monte Carlo simulation studies. *Journal of Open Source Software* 3(26):739, doi: [10.21105/joss.00739](https://doi.org/10.21105/joss.00739)

Examples

```
data("MIsim", package = "rsimsum")
s <- simsum(data = MIsim, estvarname = "b", true = 0.5, se = "se", methodvar = "method", ref = "CC")
```

```
# If 'ref' is not specified, the reference method is inferred
s <- simsum(data = MIsim, estvarname = "b", true = 0.5, se = "se", methodvar = "method")
```

summary.multisimsum *Summarising multisimsum objects*

Description

The `summary()` method for objects of class `multisimsum` returns confidence intervals for performance measures based on Monte Carlo standard errors.

Usage

```
## S3 method for class 'multisimsum'
summary(object, ci_level = 0.95, df = NULL,
        stats = NULL, ...)
```

Arguments

<code>object</code>	An object of class <code>multisimsum</code> .
<code>ci_level</code>	Significance level for confidence intervals based on Monte Carlo standard errors. Ignored if a <code>multisimsum</code> object with control parameter <code>mcse = FALSE</code> is passed.
<code>df</code>	Degrees of freedom of a <code>t</code> distribution that will be used to calculate confidence intervals based on Monte Carlo standard errors. If <code>NULL</code> (the default), quantiles of a Normal distribution will be used instead.
<code>stats</code>	Summary statistics to include; can be a scalar value or a vector (for multiple summary statistics at once). Possible choices are: <ul style="list-style-type: none"> • <code>nsim</code>, the number of replications with non-missing point estimates and standard error. • <code>thetamean</code>, average point estimate. • <code>thetamedian</code>, median point estimate. • <code>se2mean</code>, average standard error. • <code>se2median</code>, median standard error. • <code>bias</code>, bias in point estimate. • <code>empse</code>, empirical standard error. • <code>mse</code>, mean squared error. • <code>relprec</code>, percentage gain in precision relative to the reference method. • <code>modelse</code>, model-based standard error. • <code>relerror</code>, relative percentage error in standard error. • <code>cover</code>, coverage of a nominal <code>level%</code> confidence interval. • <code>becover</code>, bias corrected coverage of a nominal <code>level%</code> confidence interval. • <code>power</code>, power of a $(1 - \text{level})\%$ level test. Defaults to <code>NULL</code>, in which case all possible summary statistics are included.
<code>...</code>	Ignored.

Value

An object of class `summary.multisimsum`.

See Also

`multisimsum()`, `print.summary.multisimsum()`

Examples

```
data(frailty)
ms <- multisimsum(
  data = frailty, par = "par", true = c(
    trt = -0.50,
    fv = 0.75
  ), estvarname = "b", se = "se", methodvar = "model",
  by = "fv_dist"
)
sms <- summary(ms)
sms
```

summary.simsum

Summarising simsum objects

Description

The `summary()` method for objects of class `simsum` returns confidence intervals for performance measures based on Monte Carlo standard errors.

Usage

```
## S3 method for class 'simsum'
summary(object, ci_level = 0.95, df = NULL,
  stats = NULL, ...)
```

Arguments

<code>object</code>	An object of class <code>simsum</code> .
<code>ci_level</code>	Significance level for confidence intervals based on Monte Carlo standard errors. Ignored if a <code>simsum</code> object with control parameter <code>mcse = FALSE</code> is passed.
<code>df</code>	Degrees of freedom of a t distribution that will be used to calculate confidence intervals based on Monte Carlo standard errors. If <code>NULL</code> (the default), quantiles of a Normal distribution will be used instead. However, using Z-based or t-based confidence intervals is valid only for summary statistics such as bias and coverage. Confidence intervals for other quantities may not be appropriate, therefore their usage is not recommended.
<code>stats</code>	Summary statistics to include; can be a scalar value or a vector (for multiple summary statistics at once). Possible choices are:

- nsim, the number of replications with non-missing point estimates and standard error.
- thetamean, average point estimate.
- thetamedian, median point estimate.
- se2mean, average standard error.
- se2median, median standard error.
- bias, bias in point estimate.
- empse, empirical standard error.
- mse, mean squared error.
- relprec, percentage gain in precision relative to the reference method.
- modelse, model-based standard error.
- relerror, relative percentage error in standard error.
- cover, coverage of a nominal level% confidence interval.
- becover, bias corrected coverage of a nominal level% confidence interval.
- power, power of a (1 - level)% level test. Defaults to NULL, in which case all possible summary statistics are included.

... Ignored.

Value

An object of class `summary.simsum`.

See Also

[simsum\(\)](#), [print.summary.simsum\(\)](#)

Examples

```
data("MIsim")
object <- simsum(
  data = MIsim, estvarname = "b", true = 0.5, se = "se",
  methodvar = "method"
)
xs <- summary(object)
xs
```


Index

*Topic **datasets**

frailty, [8](#)

MIsim, [11](#)

relhaz, [18](#)

autoplot.multisimsum, [2](#)

autoplot.simsum, [3](#)

autoplot.summary.multisimsum, [4](#)

autoplot.summary.simsum, [6](#)

dropbig, [7](#)

frailty, [8](#)

get_data, [9](#)

is.multisimsum, [10](#)

is.simsum, [10](#)

is.summary.multisimsum, [11](#)

is.summary.simsum, [11](#)

MIsim, [11](#)

multisimsum, [12](#)

multisimsum(), [23](#)

nsim, [14](#)

print.multisimsum, [15](#)

print.simsum, [16](#)

print.summary.multisimsum, [16](#)

print.summary.multisimsum(), [23](#)

print.summary.simsum, [17](#)

print.summary.simsum(), [24](#)

relhaz, [18](#)

rsimsum, [19](#)

rsimsum-package (rsimsum), [19](#)

simsum, [20](#)

simsum(), [12](#), [24](#)

summary.multisimsum, [22](#)

summary.multisimsum(), [17](#)

summary.simsum, [23](#)

summary.simsum(), [3-6](#), [18](#)