Package: SimTOST (via r-universe)

October 9, 2026

Title Sample Size Estimation for Bio-Equivalence Trials Through Simulation

Version 1.1.0

Description Sample size estimation for bio-equivalence trials is supported through a simulation-based approach that extends the Two One-Sided Tests (TOST) procedure. The methodology provides flexibility in hypothesis testing, accommodates multiple treatment comparisons, and accounts for correlated endpoints. Users can model complex trial scenarios, including parallel and crossover designs, intra-subject variability, and different equivalence margins. Monte Carlo simulations enable accurate estimation of power and type I error rates, ensuring well-calibrated study designs. The statistical framework builds on established methods for equivalence testing and multiple hypothesis testing in bio-equivalence studies, as described in Schuirmann (1987) <doi:10.1007/BF01068419>, Mielke et al. (2018) <doi:10.1080/19466315.2017.1371071>, Shieh (2022) <doi:10.1371/journal.pone.0269128>, and Sozu et al. (2015) <doi:10.1007/978-3-319-22005-5>. Comprehensive documentation and vignettes guide users through implementation and interpretation of results.

License Apache License (>= 2)

Encoding UTF-8

Imports MASS, Rcpp (>= 1.0.13), data.table, matrixcalc, parallel

Suggests ggplot2, kableExtra, knitr, rmarkdown, scales, testthat (>= 3.0.0), tibble

VignetteBuilder knitr

Config/testthat/edition 3

LinkingTo Rcpp, RcppArmadillo

URL https://smartdata-analysis-and-statistics.github.io/SimTOST/, https://github.com/smartdata-analysis-and-statistics/SimTOST

Config/roxygen2/version 8.1.0

BugReports https://github.com/smartdata-analysis-and-statistics/SimTOST/issues

NeedsCompilation yes

Author Thomas Debray [aut, cre], Tim Friede [ctb], Johanna Munoz [ctb], Dewi Amaliah [ctb], Wei Wei [ctb], Marian Mitroiu [ctb], Scott McDonald [ctb], Biogen Inc [cph, fnd] (Copyright holder and funder of the original implementation v1.0.0), Smart Data Analysis and Statistics B.V. [cph, fnd] (Copyright holder and funder of subsequent developments)

Maintainer Thomas Debray <tdebray@fromdatatowisdom.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-09 15:20:07 UTC

RemoteUrl https://github.com/cran/SimTOST

RemoteRef HEAD

RemoteSha 368ae699e7153e288f5e3917e6dbedbe3ad8ef9c

Contents

.plot_correlation . . . 3
.power_count_joint_serial . . . 4
.power_count_serial . . . 6
[.simss_mielke . . . 8
as.numeric.simss_mielke . . . 9
confint.countpower . . . 9
confint.simpower . . . 10
confint.simss . . . 10
get_par . . . 11
plot.countpower . . . 12
plot.countss . . . 13
plot.simpower . . . 14
plot.simpower_curve . . . 14
plot.simss . . . 15
plot.type1error . . . 16
plot.type1error_joint . . . 17
plot_decision_heatmap . . . 17
plot_distribution . . . 18
plot_mc_error . . . 19
plot_stability . . . 20
power_dom . . . 21
print.countpower . . . 22
print.countss . . . 22
print.simpower . . . 23
print.simss . . . 23
print.simss_mielke . . . 24
run_simulations . . . 24
run_simulations_2x2_dom . . . 25

run_simulations_2x2_rom 26
run_simulations_par_dom 28
run_simulations_par_rom 30
sampleSize 31
sampleSize_count 37
sampleSize_count_joint 39
sampleSize_Mielke 42
simParallelEndpoints 44
simPower 45
SimTOST 48
summary.countpower 50
summary.countss 50
summary.simpower 51
summary.simss 51
test_2x2_dom 52
test_2x2_rom 53
test_par_dom 54
test_par_rom 56
type1Error 58
update.countpower 59
update.countss 60
update.simpower 60
update.simss 61

Index **62**

`.plot_correlation` *Plot simulated endpoint correlations*

Description

Shows the distribution of within-trial, within-arm endpoint correlations. The dashed line is the user-specified target correlation.

Usage

```
.plot_correlation(
  x,
  type = c("density", "histogram", "ecdf"),
  display = "all",
  endpoint = NULL,
  arm = NULL,
  show_reference = TRUE,
  max_points = 100000L,
  ...
)
```

Arguments

x	A result returned by sampleSize() or simPower() with keep_sim_data = TRUE.
type	Plot type: "density", "histogram", or "ecdf".
display	Comparator names to display, or "all".
endpoint	Optional endpoint names to display.
arm	Optional arm names to display.
show_reference	Logical; overlay the target correlation.
max_points	Maximum number of retained observations used.
...	Unused additional arguments.

Value

A ggplot object.

.power_count_joint_serial

Estimate joint power for correlated count endpoints and multiple comparisons

---

Description

Each simulated trial contains all arms and endpoints. Endpoint counts are generated with their requested marginal Poisson or negative-binomial distributions and a Gaussian-copula dependence structure. Every comparison must pass at least k endpoints for the trial to count as a success.

Usage

```
.power_count_joint_serial(
  n_per_arm,
  rates,
  comparisons,
  exposure = 1,
  margin_lower = 0.8,
  margin_upper = 1.25,
  model = c("poisson", "negative-binomial"),
  dispersion = 0.1,
  alpha = 0.05,
  endpoint_corr = NULL,
  k = NULL,
  type_y = NULL,
  adjust = c("none", "bonferroni", "sidak", "t", "pc", "partial-conjunction",
    "partial_conjunction", "sequential"),
  nsim = 5000,
  seed = NULL,
```

```
  design = c("parallel"),
  list_margin_lower = NULL,
  list_margin_upper = NULL,
  type_y_active = FALSE
)
```

Arguments

n_per_arm	Subjects in each arm. This joint implementation supports parallel-group designs.
rates	Named list of equal-length endpoint-rate vectors, one per arm.
comparisons	Named list of length-two character vectors. The first arm is the test arm and the second is the reference arm.
exposure	Exposure per subject, scalar or one value per endpoint, or a named list with one scalar/vector per arm.
margin_lower	Lower rate-ratio equivalence margin.
margin_upper	Upper rate-ratio equivalence margin.
model	Count model: "poisson" or "negative-binomial".
dispersion	Positive negative-binomial dispersion parameter, scalar or a named list with one scalar/vector per arm.
alpha	One-sided significance level, scalar or one value per endpoint.
endpoint_corr	Positive-definite latent Gaussian correlation matrix across endpoints. The default is independence.
k	Number of endpoints that must pass within every comparison.
type_y	Numeric endpoint hierarchy used with adjust = "seq": 1 for primary/co-primary endpoints and 2 for secondary endpoints.
adjust	Multiplicity adjustment within each comparison's selected endpoint family: "none", "bonferroni", "sidak", "t", or "seq"/"sequential". The "t" option uses Mielke's strong k-out-of-m calibration alpha / (m - k + 1); legacy partial-conjunction labels are accepted. Sequential testing uses the primary gate and secondary-family rule used by the continuous kernels. When all supplied endpoints are required (k equals the endpoint count), endpoint-wise adjustment is not necessary for the intersection-union decision; a requested adjustment remains available with a warning.
nsim	Number of simulated trials.
seed	Optional random seed.
design	Joint multi-arm design; currently only "parallel" is supported because multiple reference arms do not define a single standard 2x2 crossover design.
list_margin_lower	Optional named list of lower margins, one vector per comparison. Each vector is scalar or has one value per endpoint.
list_margin_upper	Optional named list of upper margins, one vector per comparison. Each vector is scalar or has one value per endpoint.
type_y_active	Internal flag indicating whether type_y is active.

Value

An object of class countpower containing joint power and a binomial confidence interval.

Examples

```
rates <- list(TEST = c(.20, .20), REF = c(.20, .20), ALT = c(.20, .20))
SimTOST:::power_count_joint(100, rates, list(REF = c("TEST", "REF"),
                     ALT = c("TEST", "ALT")), nsim = 100, seed = 1)
```

.power_count_serial *Estimate power for count-rate equivalence*

Description

Estimate power for count-rate equivalence

Usage

```
.power_count_serial(
  n_per_arm,
  rate_test,
  rate_reference,
  exposure = 1,
  margin_lower = 0.8,
  margin_upper = 1.25,
  model = c("poisson", "negative-binomial"),
  dispersion = 0.1,
  alpha = 0.05,
  nsim = 5000,
  seed = NULL,
  design = c("parallel", "2x2"),
  k = NULL,
  endpoint_corr = NULL,
  type_y = NULL,
  adjust = c("none", "bonferroni", "sidak", "t", "pc", "partial-conjunction",
    "partial_conjunction", "sequential"),
  sigmaB = 0,
  Eper = c(0, 0),
  Eco = c(0, 0),
  dropout = c(0, 0),
  type_y_active = FALSE
)
```

Arguments

n_per_arm	Subjects per arm.
rate_test	Event rate in the test arm.
rate_reference	Event rate in the reference arm.
exposure	Exposure per subject; a scalar or one value per endpoint.
margin_lower	Lower rate-ratio margin; a scalar or one value per endpoint.
margin_upper	Upper rate-ratio margin; a scalar or one value per endpoint.
model	Count model: "poisson" or "negative-binomial".
dispersion	Positive negative-binomial dispersion parameter. The per-subject negative-binomial size is 1 / dispersion; parallel-arm totals use size n / dispersion.
alpha	One-sided significance level.
nsim	Number of simulations.
seed	Optional random seed.
design	Trial design: "parallel" or "2x2". For "2x2", n_per_arm is interpreted as subjects per sequence.
k	Number of endpoints that must demonstrate equivalence. Defaults to all supplied endpoints.
endpoint_corr	Endpoint correlation matrix used by the Gaussian copula for multi-endpoint count simulations. The default is independence.
type_y	Numeric endpoint hierarchy used with adjust = "seq": 1 for primary/co-primary endpoints and 2 for secondary endpoints. Named vectors are recommended when endpoint names are available.
adjust	Multiplicity adjustment for endpoint-wise one-sided alpha: "none", "bonferroni", "sidak", "t", or "seq"/"sequential". The "t" option uses Mielke's strong k-out-of-m calibration alpha / (m - k + 1); legacy partial-conjunction labels are accepted. Sequential testing applies the same primary-gate/secondary-family rule as the continuous kernels. When k equals the number of supplied endpoints, endpoint-wise adjustment is not necessary for the all-endpoints-required intersection-union decision; the requested method is retained but a warning is issued.
sigmaB	Between-subject standard deviation on the log-rate scale for the count 2x2 design.
Eper	Numeric vector of length 2 containing period effects on the log-rate scale.
Eco	Numeric vector of length 2 containing carry-over effects on the log-rate scale, ordered as reference carry-over and treatment carry-over.
dropout	Numeric vector of length 2 containing dropout proportions for the two crossover sequences.
type_y_active	Internal flag indicating whether type_y is active.

Details

For `design = "2x2"`, complete participants contribute one count under each treatment. The kernel analyzes within-participant log-rate contrasts, averages the two sequence-specific estimates to remove period effects, and applies the carry-over correction implied by `Eco = c(reference_carryover, treatment_carryover)`. `exposure` is used as the log-rate offset. `sigmaB` is the standard deviation of a subject random intercept used in the count-generating model; it cancels from the within-participant treatment contrast. The standard error is estimated from the empirical variance of the subject-level contrasts. Participants who drop out before completing both periods do not contribute to this paired analysis.

Value

An object of class `countpower` containing estimated power and its binomial confidence interval.

Examples

```
SimTOST:::power_count(40, 0.20, 0.20, nsim = 100, seed = 1)
```

`[.simss_mielke` *Preserve named-vector indexing for legacy Mielke examples*

Description

Preserve named-vector indexing for legacy Mielke examples

Usage

```
## S3 method for class 'simss_mielke'
x[i, ...]
```

Arguments

`x`	A `simss_mielke` object.
`i`	Index supplied to `[`.
`...`	Unused additional arguments.

as.numeric.simss_mielke *Coerce a Mielke result to a numeric sample size*

Description

Coerce a Mielke result to a numeric sample size

Usage

```
## S3 method for class 'simss_mielke'
as.numeric(x, ...)
```

Arguments

x	A simss_mielke object.
...	Unused additional arguments.

confint.countpower *Extract the Monte Carlo confidence interval from count power results*

Description

Extract the Monte Carlo confidence interval from count power results

Usage

```
## S3 method for class 'countpower'
confint(object, parm = NULL, level = 0.95, ...)

## S3 method for class 'countss'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

object	An object returned by a count power calculation.
parm	Unused; included for compatibility with stats::confint().
level	Confidence level for the Monte Carlo interval.
...	Unused additional arguments.

Value

A named vector containing the lower and upper interval limits.

confint.simpower — *Extract the Monte Carlo confidence interval from fixed-sample-size power results*

Description

Extract the Monte Carlo confidence interval from fixed-sample-size power results

Usage

```
## S3 method for class 'simpower'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	An object returned by simPower().
parm	Unused; included for compatibility with stats::confint().
level	Confidence level for the Monte Carlo interval.
...	Unused additional arguments.

Value

A two-column matrix containing the lower and upper interval limits.

confint.simss — *Confidence Interval for Achieved Power from simss object*

Description

Confidence Interval for Achieved Power from simss object

Usage

```
## S3 method for class 'simss'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

object	An object of class "simss" returned by a sampleSize function
parm	Unused; included for compatibility with stats::confint().
level	Confidence level for the Monte Carlo interval.
...	Additional arguments (currently unused).

Value

A named numeric vector with two elements:

Achieved Power Achieved power.

Lower Lower bound of the confidence interval.

Upper Upper bound of the confidence interval.

Examples

```
## Not run:
# confint(res), where res is returned by sampleSize()

## End(Not run)
```

`get_par` *Parameter Configuration for Endpoints and Comparators*

Description

Constructs and returns a list of key parameters (mean vectors, variance-covariance matrices, and allocation rates) required for input into the `sampleSize` function. This function ensures that the parameters for each endpoint and comparator are consistent, properly named, and formatted.

Usage

```
get_par(
  mu_list,
  varcov_list,
  TAR_list,
  type_y = NA,
  arm_names = NA,
  y_names = NA
)
```

Arguments

`mu_list`	A list of mean (μ) vectors. Each element in the list represents a comparator, with the corresponding μ vector having a length equal to the number of endpoints.
`varcov_list`	A list of variance-covariance matrices. Each element corresponds to a comparator, with a matrix of size $(n \times n)$, where n is the number of endpoints.
`TAR_list`	A list of treatment allocation rates (TARs) for each comparator. Each element contains a numeric value (can be fractional or integer) representing the allocation rate for the respective comparator.
`type_y`	A numeric vector specifying the type of each endpoint. Use `1` for primary endpoints and `2` for secondary or other endpoints.

arm_names	(Optional) A character vector containing names of the arms. If not provided, default names (e.g., T1, T2, ...) will be generated.
y_names	(Optional) A character vector containing names of the endpoints. If not provided, default names (e.g., y1, y2, ...) will be generated.

Value

A named list with the following components:

mu A list of mean vectors, named according to arm_names.

varcov A list of variance-covariance matrices, named according to arm_names.

tar A list of treatment allocation rates (TARs), named according to arm_names.

type_y A vector specifying the type of each endpoint.

weight_seq A weight sequence calculated from type_y, used for endpoint weighting.

y_names A vector of names for the endpoints, named as per y_names.

#' @details This function ensures that all input parameters (mu_list, varcov_list, and TAR_list) are consistent across comparators and endpoints. It performs checks for positive semi-definiteness of variance-covariance matrices and automatically assigns default names for arms and endpoints if not provided.

Examples

```
mu_list <- list(c(0.1, 0.2), c(0.15, 0.25))
varcov_list <- list(matrix(c(1, 0.5, 0.5, 1), ncol = 2), matrix(c(1, 0.3, 0.3, 1), ncol = 2))
TAR_list <- list(0.5, 0.5)
get_par(mu_list, varcov_list, TAR_list, type_y = c(1, 2), arm_names = c("Arm1", "Arm2"))
```

---

plot.countpower *Plot count-outcome power results*

Description

Plot count-outcome power results

Usage

```
## S3 method for class 'countpower'
plot(x, target_power = 0.8, ...)
```

Arguments

x	An object returned by simPower().
target_power	Target power shown as a horizontal reference line. The default is 0.80, unless the object stores a planning target, in which case that target is used when this argument is omitted. An explicit value always overrides the stored target.
...	Unused additional arguments.

Value

A ggplot object showing estimated power and its Monte Carlo confidence interval. The interval is shown as a clearly visible vertical line through each point.

`plot.countss` *Plot count-outcome sample-size results*

Description

Plot count-outcome sample-size results

Usage

```
## S3 method for class 'countss'
plot(x, target_power = 0.8, display = "all", all = TRUE, endpoint = NULL, ...)
```

Arguments

x	An object returned by sampleSize(). Count-specific dispatch is retained through the secondary compatibility class.
target_power	Target power shown as a horizontal reference line. The default is 0.80, unless the object stores a planning target, in which case that target is used when this argument is omitted. An explicit value always overrides the stored target.
display	Character vector of comparator panels to display. Use "all" (the default) to display every comparator and, for joint count analyses, the all-comparisons panel.
all	Logical. If TRUE (the default), display only the aggregate All comparators result without a comparator facet. If FALSE, display the comparator panels selected by display.
endpoint	Endpoint names to display. By default, "Total" is used when all = TRUE, while all retained endpoints are used when all = FALSE. Use "all" to show all retained endpoints explicitly.
...	Unused additional arguments.

Value

A ggplot object showing the simulated power curve over the evaluated candidate sample sizes, with the selected sample size highlighted. Confidence intervals are shown as clearly visible vertical lines through each point. If an older countss object has no search history, the plot falls back to the selected-sample-size point.

plot.simpower *Plot fixed-sample-size power results*

Description

Plot fixed-sample-size power results

Usage

```
## S3 method for class 'simpower'
plot(x, target_power = 0.8, display = "all", all = TRUE, endpoint = NULL, ...)
```

Arguments

x	An object returned by simPower().
target_power	Target power shown as a horizontal reference line.
display	Character vector of comparator panels to display. This argument is retained for compatibility with curve results.
all	Logical retained for compatibility with curve results. If TRUE, the aggregate result is displayed when available.
endpoint	Endpoint names to display for curve results. By default, "Total" is used when all = TRUE, while all retained endpoints are used when all = FALSE. Use "all" to show all retained endpoints.
...	Unused additional arguments.

Value

A ggplot object showing estimated power and its Monte Carlo confidence interval.

plot.simpower_curve *Plot fixed-sample-size power results*

Description

Plot fixed-sample-size power results

Usage

```
## S3 method for class 'simpower_curve'
plot(x, target_power = 0.8, display = "all", all = TRUE, endpoint = NULL, ...)
```

Arguments

x	An object returned by simPower().
target_power	Target power shown as a horizontal reference line. The default is 0.80, unless the object stores a planning target, in which case that target is used when this argument is omitted. An explicit value always overrides the stored target.
display	Character vector of comparator panels to display. Use "all" (the default) to display every comparator, or provide comparator names such as "R_vs_T" to display selected panels.
all	Logical. If TRUE (the default), display only the aggregate All comparators result without a comparator facet. If FALSE, display the current comparator panels selected by display.
endpoint	Endpoint names to display. By default, "Total" is used when all = TRUE, while all retained endpoints are used when all = FALSE. Use "all" to show all retained endpoints explicitly.
...	Unused additional arguments.

Value

A ggplot object showing estimated power and its Monte Carlo confidence interval as a clearly visible vertical line.

---

plot.simss *Plot Power vs Sample Size for Simulation Results*

Description

Generates a detailed plot showing the relationship between power and total sample size for each comparator and the overall combined comparators. The combined-comparator panel is omitted when only one comparator is present. The plot also includes confidence intervals for power estimates and highlights the target power with a dashed line for easy visual comparison.

Usage

```
## S3 method for class 'simss'
plot(x, target_power = 0.8, display = "all", all = TRUE, endpoint = NULL, ...)
```

Arguments

x	An object of class simss containing simulation results.
target_power	Target power shown as a horizontal reference line. The default is 0.80, unless the object stores a planning target, in which case that target is used when this argument is omitted. An explicit value always overrides the stored target.
display	Character vector of comparator panels to display. Use "all" (the default) to display every comparator, or provide comparator names such as "R_vs_T" to display selected panels.

all	Logical. If TRUE (the default), display only the aggregate All comparators result without a comparator facet. If FALSE, display the comparator panels selected by display.
endpoint	Endpoint names to display. By default, "Total" is used when all = TRUE, while all retained endpoints are used when all = FALSE. Use "all" to show all retained endpoints explicitly.
...	Additional arguments to be passed to the plot.simss function for customization.

Details

The plot dynamically adjusts to exclude unnecessary components, such as redundant endpoints or comparators with insufficient data, ensuring clarity and simplicity. The ggplot2 framework is used for visualizations, allowing further customization if needed.

Value

A ggplot object illustrating:

- Power (y-axis) vs. Total Sample Size (x-axis) for individual endpoints and comparators.
- Clearly visible vertical lines representing the 95% confidence interval of the power estimates.
- A dashed horizontal line indicating the target power for comparison.
- A larger outlined point at the selected sample size.
- Faceted panels for each comparator, making it easy to compare results across different groups.

Author(s)

Johanna Muñoz <johanna.munoz@fromdatatowisdom.com>

---

plot.type1error *Plot empirical Type I error*

Description

The plot displays the complete-trial Type I error at all evaluated boundaries. The worst-case scenario is highlighted with a thicker interval and a larger point, and its estimated value is shown next to the point. Error bars are 95% Monte Carlo confidence intervals.

Usage

```
## S3 method for class 'type1error'
plot(x, ...)
```

Arguments

x	An object returned by type1Error().
...	Unused additional arguments.

Value

A ggplot object.

---

plot.type1error_joint *Plot joint Type I error scenarios*

Description

Displays the complete-trial Type I error for the minimal required boundary scenarios by default. The worst-case displayed scenario is highlighted with a thicker interval and a larger point, and its estimated value is shown directly above the point. The full scenario table, including larger null counts, remains available in the returned object and can be displayed with `null_count = "all"`. The dashed line is the nominal alpha level. The dotted line is the simultaneous one-sided Monte Carlo upper bound for the maximum.

Usage

```
## S3 method for class 'type1error_joint'
plot(x, null_count = c("minimal", "all"), ...)
```

Arguments

x	An object returned by `type1Error(..., joint = TRUE)`.
null_count	Character string. The default, `"minimal"`, plots only the minimal required number of boundary endpoints (`m-k+1`) for each comparator. Use `"all"` to plot every boundary count evaluated by `type1Error()`.
...	Unused additional arguments.

Value

A ggplot object.

---

plot_decision_heatmap *Plot simulated decision heatmaps*

Description

Shows the equivalence decision (pass or fail) for every endpoint and comparator in every simulated trial. Each tile is one binary decision; green means that the criterion passed and orange means that it failed. The `Total` row is the combined decision for the comparator, and an `All comparators` panel, when present, shows the overall decision. This makes isolated failures, unstable endpoints, and systematic comparator differences easy to identify.

Usage

```
plot_decision_heatmap(x, display = "all", ...)
```

Arguments

x	A simss, simpower, or simpower_curve object.
display	Comparator names to display, or "all".
...	Unused additional arguments.

Details

The x-axis is the simulation-trial number and has no scientific meaning beyond showing the sequence of simulated datasets. Endpoint rows show individual decisions; Total shows the combined endpoint decision according to k and the specified decision rule. The heatmap is a diagnostic of trial-level decisions, not a replacement for the numerical power estimate.

Value

A ggplot object.

---

plot_distribution *Plot distributions of retained simulated observations*

Description

Creates density, histogram, ECDF, or Q–Q plots for a selected trial-level quantity. The estimand argument selects an arm parameter, a reconstructed TOST statistic, a continuous comparison, a count rate ratio, or endpoint correlations.

Usage

```
plot_distribution(x, estimand = "mu", arms = NULL, endpoints = NULL, ...)
```

Arguments

x	A result returned by sampleSize() or simPower() with keep_sim_data = TRUE.
estimand	Quantity to display: "mu" (or "mean"), "sigma", "dispersion", "rate", "t_value", "DOM", "ROM", "RR", or "correlation". "mean" and "sigma" display arm-specific simulation parameters; "DOM" and "ROM" display continuous comparison estimates; "RR" displays count rate ratios; and "correlation" displays endpoint correlations.
arms	Optional character vector of arm names. For mean and sigma, these are the arms to facet. For comparison estimands, these select comparator panels containing at least one supplied arm. If NULL, all are shown.
endpoints	Optional character vector of endpoint names. If NULL, all endpoints are displayed.

...	Optional plotting controls, including type, show_reference, n_trials, seed, and max_points. Legacy selector names are accepted for backward compatibility.

Details

Every comparator follows the package convention c(test, reference). For t_value, the lower and upper TOST statistics are reconstructed from the retained outcomes using the same continuous parallel formulas as the test. The dashed lines are the corresponding one-sided critical values. The plotted DOM is test - reference and the plotted ROM is test / reference; reversing the comparator reverses the displayed estimand. The plotted RR is rate_test / rate_reference.

Value

A ggplot object.

---

plot_mc_error *Plot Monte Carlo error*

Description

Displays the 95\ of simulated trials increases. Decreasing curves indicate improving precision; a curve that has not flattened may require a larger nsim.

Usage

```
plot_mc_error(x, display = "all", overall = FALSE, endpoint = "Total", ...)
```

Arguments

x	A simss, simpower, or simpower_curve object, or a list of such objects. A list can compare simulations with different n and nsim values.
display	Comparator names to display, or "all".
overall	Logical. If TRUE, display only the All comparators aggregate result. This is useful for assessing the overall decision when several comparisons are required; display is ignored in this mode.
endpoint	Endpoint names to display. The default, "Total", shows only the aggregate endpoint. Use "all" to show all retained endpoints.
...	Unused additional arguments.

Value

A ggplot object.

plot_stability *Plot simulation stability*

Description

Displays cumulative achieved power over simulated trials. A stable curve approaches a horizontal plateau, while large late movements indicate that more simulations may be needed.

Usage

```
plot_stability(
  x,
  target_power = 0.8,
  display = "all",
  overall = FALSE,
  endpoint = "Total",
  ...
)
```

Arguments

x	A simss, simpower, or simpower_curve object, or a list of such objects. A list can compare simulations with different n and nsim values.
target_power	Target power shown as a horizontal reference line.
display	Comparator names to display, or "all".
overall	Logical. If TRUE, display only the All comparators aggregate result. This is useful for assessing the overall decision when several comparisons are required; display is ignored in this mode.
endpoint	Endpoint names to display. The default, "Total", shows only the aggregate endpoint. Use "all" to show all retained endpoints.
...	Unused additional arguments.

Value

A ggplot object.

power_dom *Power Calculation for Difference of Means (DOM) Hypothesis Test*

Description

Computes the statistical power for testing the difference of means (DOM) between two groups using Monte Carlo simulations. The power is estimated based on specified sample sizes, means, standard deviations, and significance level.

Usage

```
power_dom(
  seed,
  mu_test,
  mu_control,
  sigma_test,
  sigma_control,
  N_test,
  N_control,
  lb,
  ub,
  alpha = 0.05,
  nsim = 10000
)
```

Arguments

seed	Integer. Seed for reproducibility.
mu_test	Numeric. Mean of the test group.
mu_control	Numeric. Mean of the control group.
sigma_test	Numeric. Standard deviation of the test group.
sigma_control	Numeric. Standard deviation of the control group.
N_test	Integer. Sample size of the test group.
N_control	Integer. Sample size of the control group.
lb	Numeric. Lower bound for the equivalence margin.
ub	Numeric. Upper bound for the equivalence margin.
alpha	Numeric. Significance level (default = 0.05).
nsim	Integer. Number of simulations (default = 10,000).

Value

Numeric. Estimated power (probability between 0 and 1).

print.countpower *Print count-outcome power results*

Description

Print count-outcome power results

Usage

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

Arguments

x	A countpower object.
...	Unused additional arguments.

print.countss *Print count-outcome sample-size results*

Description

Print count-outcome sample-size results

Usage

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

Arguments

x	A countss object.
...	Unused additional arguments.

print.simpower *Print fixed-sample-size power results*

Description

Print fixed-sample-size power results

Usage

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

Arguments

x	An object returned by simPower().
...	Unused additional arguments.

Value

The input object, invisibly.

print.simss *Print Summary of Sample Size Estimation*

Description

Prints the summary results of the sample size estimation for bioequivalence trials, including achieved power, total sample size, and power confidence intervals. The function also details the study design, primary endpoint comparisons, and applied multiplicity corrections.

Usage

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

Arguments

x	An object of class "simss", typically generated by a sample size estimation function.
...	Optional arguments to be passed from or to other methods.

Details

This function displays key metrics from a sample size estimation analysis. It provides an overview of the study design, treatment comparisons, tested endpoints, significance level adjustments, and estimated sample size. For studies with multiple primary endpoints, it describes the multiplicity correction applied.

Value

No return value, called for side effects. The function prints the summary results of the sample size estimation to the console in a structured format.

Author(s)

Thomas Debray <tdebray@fromdatatowisdom.com>

`print.simss_mielke` *Print a Mielke sample-size result*

Description

Print a Mielke sample-size result

Usage

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

Arguments

`x`	A `simss_mielke` object.
`...`	Unused additional arguments.

`run_simulations` *Run a design-specific simulation*

Description

Dispatches to the compiled simulation routine for a selected trial design and test type.

Usage

```
run_simulations(design = c("parallel", "2x2"), ctype = c("DOM", "ROM"), ...)
```

Arguments

`design`	Trial design, either `"parallel"` or `"2x2"`.
`ctype`	Test type, either `"DOM"` or `"ROM"`.
`...`	Arguments passed to the selected simulation routine.

Value

The result returned by the selected simulation routine.

`run_simulations_2x2_dom` *Run Simulations for a 2x2 Crossover Design with Difference of Means (DOM) test*

Description

This function simulates a 2x2 crossover trial across multiple iterations. It evaluates equivalence across multiple endpoints using the Difference of Means (DOM) test.

Usage

```
run_simulations_2x2_dom(
  nsim,
  n,
  muT,
  muR,
  SigmaW,
  lequi_tol,
  uequi_tol,
  alpha,
  sigmaB,
  dropout,
  Eper,
  Eco,
  typey,
  adseq,
  k,
  arm_seed
)
```

Arguments

`nsim`	Integer. The number of simulations to run.
`n`	Integer. The sample size per period.
`muT`	Numeric vector. Mean outcomes for the active treatment.
`muR`	Numeric vector. Mean outcomes for the reference treatment.
`SigmaW`	Numeric matrix. Within-subject covariance matrix for endpoints.
`lequi_tol`	Numeric vector. Lower equivalence thresholds for each endpoint.
`uequi_tol`	Numeric vector. Upper equivalence thresholds for each endpoint.
`alpha`	Numeric vector. Significance levels for hypothesis testing across endpoints.
`sigmaB`	Numeric. Between-subject variance for the crossover model.
`dropout`	Numeric vector of size 2. Dropout rates for each sequence.
`Eper`	Numeric vector. Expected period effects for each sequence.

Eco	Numeric vector. Expected carryover effects for each sequence.
typey	Integer vector indicating the classification of each endpoint, where 1 corresponds to a primary endpoint and 2 corresponds to a secondary endpoint.
adseq	Logical. If TRUE, applies sequential (hierarchical) testing.
k	Integer. Minimum number of endpoints required for equivalence.
arm_seed	Integer vector. Random seed for each simulation.

Details

This function evaluates equivalence using the Difference of Means (DOM) test. Equivalence is determined based on predefined lower (lequi_tol) and upper (uequi_tol) equivalence thresholds, and hypothesis testing is conducted at the specified significance level (alpha). If adseq is TRUE, primary endpoints must establish equivalence before secondary endpoints are evaluated. The sample size per period is adjusted based on dropout rates, ensuring valid study conclusions. The simulation incorporates within-subject correlation using SigmaW and accounts for between-subject variance with sigmaB. Expected period effects (Eper) and carryover effects (Eco) are included in the model. A fixed random seed (arm_seed) is used to ensure reproducibility across simulations.

Value

A numeric matrix where each column stores simulation results: The first row (totaly) represents the overall equivalence decision (1 = success, 0 = failure). Subsequent rows contain equivalence decisions per endpoint, mean estimates for the treatment group, mean estimates for the reference group, standard deviations for treatment, and standard deviations for reference.

Author(s)

Thomas Debray <tdebray@fromdatatowisdom.com>

---

run_simulations_2x2_rom

Run Simulations for a 2x2 Crossover Design with Ratio of Means (ROM) test

---

Description

This function simulates a 2x2 crossover trial across multiple iterations. It evaluates equivalence across multiple endpoints using the Ratio of Means (ROM) test.

Usage

```
run_simulations_2x2_rom(
  nsim,
  n,
  muT,
  muR,
```

```
  SigmaW,
  lequi_tol,
  uequi_tol,
  alpha,
  sigmaB,
  dropout,
  Eper,
  Eco,
  typey,
  adseq,
  k,
  arm_seed
)
```

Arguments

nsim	Integer. The number of simulations to run.
n	Integer. The sample size per period.
muT	Numeric vector. Mean outcomes for the active treatment.
muR	Numeric vector. Mean outcomes for the reference treatment.
SigmaW	Numeric matrix. Within-subject covariance matrix for endpoints.
lequi_tol	Numeric vector. Lower equivalence thresholds for each endpoint.
uequi_tol	Numeric vector. Upper equivalence thresholds for each endpoint.
alpha	Numeric vector. Significance levels for hypothesis testing across endpoints.
sigmaB	Numeric. Between-subject variance for the crossover model.
dropout	Numeric vector of size 2. Dropout rates for each sequence.
Eper	Numeric vector. Expected period effects for each sequence.
Eco	Numeric vector. Expected carryover effects for each sequence.
typey	Integer vector indicating the classification of each endpoint, where 1 corresponds to a primary endpoint and 2 corresponds to a secondary endpoint.
adseq	Logical. If TRUE, applies sequential (hierarchical) testing.
k	Integer. Minimum number of endpoints required for equivalence.
arm_seed	Integer vector. Random seed for each simulation.

Details

This function evaluates equivalence using the Ratio of Means (ROM) test. Equivalence is determined based on predefined lower lequi_tol and upper uequi_tol equivalence thresholds, and hypothesis testing is conducted at the specified significance level alpha. If adseq is TRUE, primary endpoints must establish equivalence before secondary endpoints are evaluated. The sample size per period is adjusted based on dropout rates, ensuring valid study conclusions. The simulation incorporates within-subject correlation using SigmaW and accounts for between-subject variance with sigmaB. Expected period effects Eper and carryover effects Eco are included in the model. A fixed random seed arm_seed is used to ensure reproducibility across simulations.//'

Value

A numeric matrix where each column stores simulation results: The first row (`totaly`) represents the overall equivalence decision (1 = success, 0 = failure). Subsequent rows contain equivalence decisions per endpoint, mean estimates for the treatment group, mean estimates for the reference group, standard deviations for treatment, and standard deviations for reference.

@author Thomas Debray <tdebray@fromdatatowisdom.com>

`run_simulations_par_dom` *Run Simulations for a Parallel Design with Difference of Means (DOM) test*

Description

This function simulates a parallel-group trial across multiple iterations. It evaluates equivalence across multiple endpoints using the Difference of Means (DOM) test.

Usage

```
run_simulations_par_dom(
  nsim,
  n,
  muT,
  muR,
  SigmaT,
  SigmaR,
  lequi_tol,
  uequi_tol,
  alpha,
  dropout,
  typey,
  adseq,
  k,
  arm_seed_T,
  arm_seed_R,
  TART,
  TARR,
  vareq
)
```

Arguments

`nsim`	Integer. The number of simulations to run.
`n`	Integer. The sample size per arm (before dropout).
`muT`	arma::vec. Mean vector for the treatment arm.

muR	arma::vec. Mean vector for the reference arm.
SigmaT	arma::mat. Covariance matrix for the treatment arm.
SigmaR	arma::mat. Covariance matrix for the reference arm.
lequi_tol	arma::rowvec. Lower equivalence thresholds for each endpoint.
uequi_tol	arma::rowvec. Upper equivalence thresholds for each endpoint.
alpha	arma::rowvec. Significance level for each endpoint.
dropout	arma::vec. Dropout rates for each arm (T, R).
typey	Integer vector indicating the classification of each endpoint, where 1 corresponds to a primary endpoint and 2 corresponds to a secondary endpoint.
adseq	Boolean. If TRUE, applies sequential (hierarchical) testing.
k	Integer. Minimum number of endpoints required for equivalence.
arm_seed_T	arma::ivec. Random seed vector for the treatment group (one per simulation).
arm_seed_R	arma::ivec. Random seed vector for the reference group (one per simulation).
TART	Double. Treatment allocation ratio (proportion of subjects in treatment arm).
TARR	Double. Reference allocation ratio (proportion of subjects in reference arm).
vareq	Boolean. If TRUE, assumes equal variances across treatment and reference groups.

Details

Equivalence testing uses either the Difference of Means (DOM) test, applying predefined equivalence thresholds and significance levels. When hierarchical testing (adseq) is enabled, all primary endpoints must demonstrate equivalence before secondary endpoints are evaluated. Dropout rates are incorporated into the sample size calculation to ensure proper adjustment. Randomization is controlled through separate random seeds for the treatment and reference groups, enhancing reproducibility.

Value

The function returns an arma::mat storing simulation results row-wise for consistency with R's output format. The first row (totaly) contains the overall equivalence decision (1 for success, 0 for failure). The subsequent rows include equivalence deicisons for each endpoint, mean estimates for both treatment and reference groups, and corresponding standard deviations.

Author(s)

Thomas Debray <tdebray@fromdatatowisdom.com>

`run_simulations_par_rom` *Run Simulations for a Parallel Design with Ratio of Means (ROM) test*

Description

This function simulates a parallel-group trial across multiple iterations. It evaluates equivalence across multiple endpoints using the Ratio of Means (ROM) test.

Usage

```
run_simulations_par_rom(
  nsim,
  n,
  muT,
  muR,
  SigmaT,
  SigmaR,
  lequi_tol,
  uequi_tol,
  alpha,
  dropout,
  typey,
  adseq,
  k,
  arm_seed_T,
  arm_seed_R,
  TART,
  TARR,
  vareq
)
```

Arguments

`nsim`	Integer. The number of simulations to run.
`n`	Integer. The sample size per arm (before dropout).
`muT`	arma::vec. Mean vector for the treatment arm.
`muR`	arma::vec. Mean vector for the reference arm.
`SigmaT`	arma::mat. Covariance matrix for the treatment arm.
`SigmaR`	arma::mat. Covariance matrix for the reference arm.
`lequi_tol`	arma::rowvec. Lower equivalence thresholds for each endpoint.
`uequi_tol`	arma::rowvec. Upper equivalence thresholds for each endpoint.
`alpha`	arma::rowvec. Significance level for each endpoint.
`dropout`	arma::vec. Dropout rates for each arm (T, R).

typey	Integer vector indicating the classification of each endpoint, where 1 corresponds to a primary endpoint and 2 corresponds to a secondary endpoint.
adseq	Boolean. If TRUE, applies sequential (hierarchical) testing.
k	Integer. Minimum number of endpoints required for equivalence.
arm_seed_T	arma::ivec. Random seed vector for the treatment group (one per simulation).
arm_seed_R	arma::ivec. Random seed vector for the reference group (one per simulation).
TART	Double. Treatment allocation ratio (proportion of subjects in treatment arm).
TARR	Double. Reference allocation ratio (proportion of subjects in reference arm).
vareq	Boolean. If TRUE, assumes equal variances across treatment and reference groups.

Details

Equivalence testing uses either the Ratio of Means (ROM) test, applying predefined equivalence thresholds and significance levels. When hierarchical testing (adseq) is enabled, all primary endpoints must demonstrate equivalence before secondary endpoints are evaluated. Dropout rates are incorporated into the sample size calculation to ensure proper adjustment. Randomization is controlled through separate random seeds for the treatment and reference groups, enhancing reproducibility.

Value

The function returns an arma::mat storing simulation results row-wise for consistency with R's output format. The first row (totaly) contains the overall equivalence decision (1 for success, 0 for failure). The subsequent rows include equivalence decisions for each endpoint, mean estimates for both treatment and reference groups, and corresponding standard deviations.

Author(s)

Thomas Debray <tdebray@fromdatatowisdom.com>

---

sampleSize *Sample Size Calculation for Bioequivalence and Multi-Endpoint Studies*

Description

Computes the required sample size to achieve a target power in studies with multiple endpoints and treatment arms. The function employs modified root-finding algorithms to estimate sample size while accounting for correlation structures, variance assumptions, and equivalence bounds across endpoints. It is particularly useful for bioequivalence trials and multi-arm studies with complex endpoint structures.

Usage

```
sampleSize(
  distribution = c("norm", "lnorm", "pois", "nbinom"),
  mu_list = NULL,
  varcov_list = NA,
  sigma_list = NA,
  cor_mat = NA,
  sigmaB = NA,
  rate_list = NULL,
  exposure = 1,
  dispersion = 0.1,
  Eper = c(0, 0),
  Eco = c(0, 0),
  rho = 0,
  TAR = rep(1, length(mu_list)),
  arm_names = NA,
  ynames_list = NA,
  type_y = NA,
  list_comparator = NA,
  list_y_comparator = NA,
  power = 0.8,
  alpha = 0.05,
  lequi.tol = NA,
  uequi.tol = NA,
  list_lequi.tol = NA,
  list_uequi.tol = NA,
  dtype = "parallel",
  ctype = "ROM",
  vareq = TRUE,
  k = NA,
  adjust = "no",
  dropout = NA,
  nsim = 5000,
  seed = 1234,
  ncores = 1,
  optimization_method = "fast",
  lower = 2,
  upper = 500,
  step.power = 6,
  step.up = TRUE,
  pos.side = FALSE,
  maxiter = 1000,
  verbose = FALSE,
  keep_sim_data = FALSE,
  .warn_redundant_bon = TRUE
)
```

Arguments

distribution	Outcome distribution. Choose the R distribution names "norm", "lnorm", "pois", or "nbinom". Matching is case-insensitive. The longer labels such as "normal", "lognormal", and "poisson" are accepted for compatibility; results store the R-style labels above.
mu_list	Named list of arithmetic means per treatment arm. Each element is a vector representing expected outcomes for all endpoints in that arm.
varcov_list	List of variance-covariance matrices, where each element corresponds to a comparator. Each matrix has dimensions: number of endpoints × number of endpoints.
sigma_list	List of standard deviation vectors, where each element corresponds to a comparator and contains one standard deviation per endpoint.
cor_mat	Matrix specifying the correlation structure between endpoints. For continuous outcomes it is used with sigma_list to calculate varcov_list; for count outcomes it is used by the joint count engine.
sigmaB	Numeric. Between-subject standard deviation parameter for the continuous 2×2 design; for count outcomes it is the log-rate standard deviation in the 2×2 kernel.
rate_list	Named list of equal-length endpoint-rate vectors, one per count-outcome arm.
exposure	Exposure per subject for count outcomes. Supply a scalar or endpoint vector shared by arms, or a named list of arm-specific values.
dispersion	Positive negative-binomial dispersion parameter. The per-subject negative-binomial size is 1 / dispersion; parallel-arm totals use size n / dispersion. Supply a scalar or endpoint vector shared by arms, or a named list of arm-specific values.
Eper	Optional numeric vector of length 2 specifying period effects. For count outcomes these are log-rate effects applied to periods 1 and 2.
Eco	Optional numeric vector of length 2 specifying carry-over effects in the order reference carry-over and treatment carry-over. For count outcomes these are log-rate effects in period 2.
rho	Numeric. Correlation parameter applied uniformly across all endpoint pairs. Used with sigma_list to compute varcov_list when cor_mat or varcov_list are not provided.
TAR	Numeric vector specifying treatment allocation rates per arm. The order must match arm_names. Defaults to equal allocation across arms if not provided.
arm_names	Optional character vector of treatment names. If not supplied, names are derived from mu_list.
ynames_list	Optional list of vectors specifying endpoint names per arm. If names are missing, arbitrary names are assigned based on order.
type_y	Integer vector indicating endpoint types: 1 for co-primary endpoints, 2 for secondary endpoints.
list_comparator	List of comparators. Each element must be a vector of length 2 in the form c(test, reference). The first arm is treated as the treatment arm and the second arm as the reference arm throughout the simulation and estimand calculations.

`list_y_comparator`
: List of endpoint sets per comparator. Each element is a vector containing endpoint names to compare. If not provided, all endpoints common to both comparator arms are used. For count outcomes, the selected endpoints define the count multiplicity and effective k; joint count analyses require the same endpoint set for every comparison.

`power`
: Numeric. Target power (default = 0.8).

`alpha`
: Numeric. Significance level (default = 0.05).

`lequi.tol`
: Numeric. Lower equivalence bounds (e.g., -0.5) applied uniformly across all endpoints and comparators.

`uequi.tol`
: Numeric. Upper equivalence bounds (e.g., 0.5) applied uniformly across all endpoints and comparators.

`list_lequi.tol`
: List of numeric vectors specifying lower equivalence bounds per comparator.

`list_uequi.tol`
: List of numeric vectors specifying upper equivalence bounds per comparator.

`dtype`
: Character. Trial design: `"parallel"` (default) for parallel-group or `"2x2"` for crossover (only for 2-arm studies).

`ctype`
: Character. Continuous-outcome test type: `"DOM"` (Difference of Means) or `"ROM"` (Ratio of Means). For Poisson and negative-binomial outcomes, `"RR"` (event-rate ratio) is used; an unavailable value triggers a warning and the applicable default is used.

`vareq`
: Logical. Assumes equal variances across arms if `TRUE` (default = `FALSE`).

`k`
: Integer vector. Minimum number of successful endpoints required for global bioequivalence per comparator. Defaults to all endpoints per comparator.

`adjust`
: Character. Alpha adjustment method: `"k"` (K-fold), `"bon"` (Bonferroni across all selected endpoints), `"sid"` (Sidak), `"t"` (Mielke's strong \(k\)-out-of-\(m\) adjustment using `alpha / (m - k + 1)`; legacy `"pc"` aliases are accepted), `"no"` (none), or `"seq"` (sequential).

`dropout`
: Numeric vector specifying dropout proportion per arm.

`nsim`
: Integer. Number of simulated studies (default = 5000).

`seed`
: Integer. Seed for reproducibility.

`ncores`
: Integer. Number of processing cores for parallel computation. Defaults to `1`. Set to `NA` for automatic detection (`ncores - 1`). For count outcomes, Monte Carlo trials are split into independent seeded chunks and evaluated by the count C++ kernel on each worker.

`optimization_method`
: Character. Sample size optimization method: `"fast"` (default, root-finding algorithm) or `"step-by-step"`.

`lower`
: Integer. Minimum sample size for search range (default = 2).

`upper`
: Integer. Maximum sample size for the search range (default = 500). For count outcomes, this is the maximum number of subjects per arm; the plotted and returned total sample size is this value multiplied by the number of trial arms.

`step.power`
: Numeric. Initial step size for sample size search, defined as `2^step.power`. Used when `optimization_method = "fast"`.

`step.up` Logical. If `TRUE` (default), search increments upward from `lower`; if `FALSE`, decrements downward from `upper`. Used when `optimization_method = "fast"`.

`pos.side` Logical. If `TRUE`, finds the smallest integer `i` closest to the root such that `f(i) > 0`. Used when `optimization_method = "fast"`.

`maxiter` Integer. Maximum iterations allowed for sample size estimation (default = 1000). Used when `optimization_method = "fast"`.

`verbose` Logical. If `TRUE`, prints progress and messages during execution (default = `FALSE`).

`keep_sim_data` Logical. If `TRUE`, retain model-scale observations for every simulated trial in `sim_data` for distribution diagnostics. Defaults to `FALSE` because retained data can be large.

`.warn_redundant_bon`
Logical. If `TRUE`, warn when a requested multiplicity adjustment is redundant or uncalibrated for the selected endpoint decision.

Details

The common planning arguments are `power`, `alpha`, `list_comparator`, `list_lequi.tol`, `list_uequi.tol`, `k`, `adjust`, `dtype`, `dropout`, `nsim`, `seed`, `lower`, and `upper`. Use the following distribution-specific arguments in addition to those common arguments:

Normal and Log Normal Supply `mu_list`, `sigma_list` or `varcov_list`; use `cor_mat` or `rho` for endpoint dependence. The `ctype` argument selects DOM or ROM testing.

Poisson and Negative Binomial Supply `rate_list`, `list_comparator`, and comparator-specific `list_lequi.tol` and `list_uequi.tol`. Use `exposure` and, for negative-binomial outcomes, `dispersion`; both may be scalar, endpoint-specific, or named arm-specific lists. Continuous-outcome arguments are ignored.

For count outcomes, `optimization_method = "fast"` brackets the first sample size whose simulated power reaches the target and refines the bracket by integer bisection. The `"step-by-step"` option remains available when a complete candidate-by-candidate power table is preferred. The fast method assumes the usual approximately monotone power curve and uses the same seed at each candidate to reduce simulation noise. The effective endpoint count is comparator-specific: when `list_y_comparator` is omitted, only endpoints present in both arms are tested; when it is supplied, only the listed endpoints are tested. `k` is validated against that comparator-specific count and oversized values are capped with a warning. Formal endpoint-wise adjustment is unnecessary when all selected endpoints are required (`k = m`), although requested Bonferroni or Sidak adjustment remains available with a warning. For `k < m`, `adjust = "no"` is explicitly reported as an uncalibrated choice. For a k-of-m decision, `adjust = "t"` applies Mielke's strong \(k\)-out-of-\(m\) calibration `alpha / (m - k + 1)`. The legacy `adjust = "pc"` label is accepted as an alias. For continuous and count outcomes, `type_y` is used with `adjust = "seq"`; named endpoint vectors are aligned to the selected comparator endpoints. Count analyses use the same primary-gate and secondary-family decision rule as the continuous kernels. The unified function returns primary class `simss` for all outcome distributions. Count results retain `countss` as a secondary compatibility class. Use `summary()` and `plot()` to inspect the result.

Value

A list containing:

`response` Array summarizing simulation results, including estimated sample sizes, achieved power, and confidence intervals.

`table.iter` Data frame showing estimated sample sizes and calculated power at each iteration. For count outcomes, one row is retained for every evaluated candidate.

`table.test` Data frame containing test results for all simulated trials. For count outcomes, this contains complete-trial, comparator, and endpoint decision indicators for each simulated trial and candidate; the count kernel returns aggregate decision counts rather than raw endpoint-level test statistics.

`param.u` Original input parameters.

`param` Final adjusted parameters used in sample size calculation.

`param.d` Trial design parameters used in the simulation.

`sim_data` Optional long-format simulated observations, returned when `keep_sim_data = TRUE`.

References

Schuirmann, D. J. (1987). A comparison of the Two One-Sided Tests procedure and the Power approach for assessing the equivalence of average bioavailability. *Journal of Pharmacokinetics and Biopharmaceutics, 15*(6), 657-680. doi:10.1007/BF01068419

Mielke, J., Jones, B., Jilma, B., & König, F. (2018). Sample size for multiple hypothesis testing in biosimilar development. *Statistics in Biopharmaceutical Research, 10*(1), 39-49. doi:10.1080/19466315.2017.1371071

Berger, R. L., & Hsu, J. C. (1996). Bioequivalence trials, intersection-union tests, and equivalence confidence sets. *Statistical Science*, 283-302.

Sozu, T., Sugimoto, T., Hamasaki, T., & Evans, S. R. (2015). "Sample Size Determination in Clinical Trials with Multiple Endpoints." *SpringerBriefs in Statistics*. doi:10.1007/978-3-319-22005-5

Examples

```
mu_list <- list(SB2 = c(AUCinf = 38703, AUClast = 36862, Cmax = 127.0),
                EUREF = c(AUCinf = 39360, AUClast = 37022, Cmax = 126.2),
                USREF = c(AUCinf = 39270, AUClast = 37368, Cmax = 129.2))

sigma_list <- list(SB2 = c(AUCinf = 11114, AUClast = 9133, Cmax = 16.9),
                   EUREF = c(AUCinf = 12332, AUClast = 9398, Cmax = 17.9),
                   USREF = c(AUCinf = 10064, AUClast = 8332, Cmax = 18.8))

# Equivalent boundaries
lequi.tol <- c(AUCinf = 0.8, AUClast = 0.8, Cmax = 0.8)
uequi.tol <- c(AUCinf = 1.25, AUClast = 1.25, Cmax = 1.25)

# Arms to be compared
list_comparator <- list(EMA = c("SB2", "EUREF"),
                        FDA = c("SB2", "USREF"))
```

```
# Endpoints to be compared
list_y_comparator <- list(EMA = c("AUCinf", "Cmax"),
                          FDA = c("AUClast", "Cmax"))

# Equivalence boundaries for each comparison
lequi_lower <- c(AUCinf = 0.80, AUClast = 0.80, Cmax = 0.80)
lequi_upper <- c(AUCinf = 1.25, AUClast = 1.25, Cmax = 1.25)

# Run the simulation
sampleSize(power = 0.9, alpha = 0.05, mu_list = mu_list,
           sigma_list = sigma_list, list_comparator = list_comparator,
           list_y_comparator = list_y_comparator,
           list_lequi.tol = list("EMA" = lequi_lower, "FDA" = lequi_lower),
           list_uequi.tol = list("EMA" = lequi_upper, "FDA" = lequi_upper),
           adjust = "no", dtype = "parallel", ctype = "ROM", vareq = FALSE,
           distribution = "lnorm", ncores = 1, nsim = 50, seed = 1234)

# The same entry point for a two-arm Poisson count-rate calculation:
sampleSize(power = 0.80, distribution = "Poisson",
           rate_list = list(TEST = 0.21, REF = 0.20),
           list_comparator = list(TEST_vs_REF = c("TEST", "REF")),
           list_lequi.tol = list(TEST_vs_REF = 0.80),
           list_uequi.tol = list(TEST_vs_REF = 1.25),
           exposure = 10, lower = 20, upper = 500,
           nsim = 100, seed = 1234)
```

`sampleSize_count` *Estimate sample size for count-rate equivalence*

Description

Searches for the smallest number of subjects per arm whose simulated power reaches the target for a rate-ratio equivalence test.

Usage

```
sampleSize_count(
  power = 0.8,
  rate_test,
  rate_reference,
  exposure = 1,
  margin_lower = 0.8,
  margin_upper = 1.25,
  model = c("poisson", "negative-binomial"),
  dispersion = 0.1,
  alpha = 0.05,
  nsim = 5000,
  seed = NULL,
  lower = 2,
```

```
  upper = 500,
  design = c("parallel", "2x2"),
  k = NULL,
  endpoint_corr = NULL,
  type_y = NULL,
  adjust = c("none", "bonferroni", "sidak", "t", "pc", "partial-conjunction",
    "partial_conjunction", "sequential"),
  sigmaB = 0,
  Eper = c(0, 0),
  Eco = c(0, 0),
  dropout = c(0, 0),
  optimization_method = c("fast", "step-by-step"),
  step.power = 6,
  step.up = TRUE,
  pos.side = FALSE,
  maxiter = 1000,
  ncores = 1,
  .warn_redundant_bon = TRUE
)
```

Arguments

power	Target power.
rate_test	Event rate in the test arm.
rate_reference	Event rate in the reference arm.
exposure	Exposure per subject; a scalar or one value per endpoint.
margin_lower	Lower rate-ratio equivalence margin.
margin_upper	Upper rate-ratio equivalence margin.
model	Count model: "poisson" or "negative-binomial".
dispersion	Positive negative-binomial dispersion parameter. The per-subject negative-binomial size is 1 / dispersion; parallel-arm totals use size n / dispersion.
alpha	One-sided significance level.
nsim	Number of simulated trials.
seed	Optional random seed.
lower	Minimum subjects per arm.
upper	Maximum subjects per arm.
design	Trial design: "parallel" or "2x2". In a crossover design, the returned sample size is per sequence.
k	Number of endpoints that must demonstrate equivalence. Defaults to all supplied endpoints.
endpoint_corr	Endpoint correlation matrix used by the Gaussian copula for multi-endpoint count simulations. The default is independence.
type_y	Numeric endpoint hierarchy used with adjust = "seq": 1 for primary/co-primary endpoints and 2 for secondary endpoints.

adjust	Multiplicity adjustment for endpoint-wise one-sided alpha: "none", "bonferroni", "sidak", "t", or "seq"/"sequential". The "t" option uses Mielke's strong k-out-of-m calibration alpha / (m - k + 1); legacy partial-conjunction labels are accepted.
sigmaB	Between-subject standard deviation for the count 2x2 design.
Eper	Period effects for the count 2x2 design.
Eco	Carry-over effects for the count 2x2 design.
dropout	Dropout proportions for the count 2x2 design.
optimization_method	Search method. "fast" brackets the power crossing and uses integer bisection; "step-by-step" evaluates every candidate sample size.
step.power	Initial power-of-two jump used by the fast search.
step.up	Direction of the initial bracketing search.
pos.side	Retained for compatibility with sampleSize(); count searches return the smallest candidate reaching the target.
maxiter	Maximum number of power evaluations.
ncores	Number of worker processes used for count simulations. Set to 1 for serial execution. Parallel execution splits nsim into reproducible independent chunks and combines the resulting successes.
.warn_redundant_bon	Logical. If TRUE, warn about redundant or uncalibrated multiplicity configurations.

Value

An object of class countss containing the selected sample size, achieved power, confidence interval, input parameters, and the search history in table.iter and table.test. For count outcomes, table.iter has one row per evaluated candidate sample size. table.test contains complete-trial, comparator, and endpoint decision indicators for each simulated trial and candidate. The count kernel returns aggregate decision counts rather than raw endpoint-level test statistics, so component columns preserve the simulated marginal success counts.

Examples

```
SimTOST:::sampleSize_count(0.80, 0.20, 0.20, lower = 100, upper = 2000,
                nsim = 100, seed = 1)
```

---

sampleSize_count_joint

Estimate sample size for joint correlated count equivalence

Description

Estimate sample size for joint correlated count equivalence

Usage

```
sampleSize_count_joint(
  power = 0.8,
  rates,
  comparisons,
  exposure = 1,
  margin_lower = 0.8,
  margin_upper = 1.25,
  model = c("poisson", "negative-binomial"),
  dispersion = 0.1,
  alpha = 0.05,
  endpoint_corr = NULL,
  k = NULL,
  type_y = NULL,
  adjust = c("none", "bonferroni", "sidak", "t", "pc", "partial-conjunction",
    "partial_conjunction", "sequential"),
  nsim = 5000,
  seed = NULL,
  lower = 2,
  upper = 500,
  design = c("parallel"),
  list_margin_lower = NULL,
  list_margin_upper = NULL,
  optimization_method = c("fast", "step-by-step"),
  step.power = 6,
  step.up = TRUE,
  pos.side = FALSE,
  maxiter = 1000,
  ncores = 1,
  .warn_redundant_bon = TRUE
)
```

Arguments

power	Target joint power.
rates	Named list of equal-length endpoint-rate vectors, one per arm.
comparisons	Named list of treatment-reference arm pairs.
exposure	Exposure per subject, scalar or one value per endpoint, or a named list with one scalar/vector per arm.
margin_lower	Lower rate-ratio equivalence margin.
margin_upper	Upper rate-ratio equivalence margin.
model	Count model: "poisson" or "negative-binomial".
dispersion	Positive negative-binomial dispersion parameter.
alpha	One-sided significance level.
endpoint_corr	Endpoint correlation matrix; the default is independence.

`k` Number of endpoints that must pass within every comparison.

`type_y` Numeric endpoint hierarchy used with `adjust = "seq"`: `1` for primary/co-primary endpoints and `2` for secondary endpoints.

`adjust` Multiplicity adjustment within each comparison's endpoint family.

`nsim` Number of simulated trials.

`seed` Optional random seed.

`lower` Minimum subjects per arm.

`upper` Maximum subjects per arm.

`design` Joint multi-arm design; currently only `"parallel"` is supported.

`list_margin_lower` Optional named list of lower margins, one vector per comparison.

`list_margin_upper` Optional named list of upper margins, one vector per comparison.

`optimization_method` Search method: `"fast"` uses bracketing and integer bisection; `"step-by-step"` evaluates every candidate.

`step.power` Initial power-of-two jump for the fast search.

`step.up` Direction of the initial fast-search bracketing.

`pos.side` Retained for compatibility with `sampleSize()`; count searches always return the smallest candidate reaching the target.

`maxiter` Maximum number of power evaluations.

`ncores` Number of worker processes used for count simulations. Set to 1 for serial execution. Parallel execution splits `nsim` into reproducible independent chunks and combines the resulting successes.

`.warn_redundant_bon` Logical. If `TRUE`, warn about redundant or uncalibrated multiplicity configurations.

Value

An object of class `countss` containing the selected sample size, achieved joint power, confidence interval, input parameters, and the search history in `table.iter` and `table.test`. For count outcomes, `table.iter` has one row per evaluated candidate sample size and `table.test` contains complete-trial, comparator, and endpoint decision indicators for each simulated trial and candidate. The count kernel returns aggregate decision counts rather than raw endpoint-level test statistics, so component columns preserve the simulated marginal success counts.

`sampleSize_Mielke` *Sample Size Estimation for Multiple Hypothesis Testing Using Mielke's Method*

Description

Estimates the required sample size to achieve a specified power level for multiple hypothesis testing, using the approach described by Mielke et al. (2018). This function is particularly useful for bioequivalence or biosimilar studies with multiple correlated endpoints, where a minimum number of endpoints must meet equivalence criteria.

Usage

```
sampleSize_Mielke(
  power,
  Nmax,
  m,
  k,
  rho,
  sigma,
  true.diff,
  equi.tol,
  design,
  alpha,
  adjust = "no",
  seed = NULL,
  nsim = 10000
)
```

Arguments

`power`	Numeric. Desired statistical power.
`Nmax`	Integer. Maximum allowable sample size.
`m`	Integer. Total number of endpoints.
`k`	Integer. Number of endpoints that must meet the success criteria for overall study success.
`rho`	Numeric. Constant correlation coefficient among endpoints.
`sigma`	Numeric or vector. Standard deviation of each endpoint. If a single value is provided, it is assumed to be constant across all endpoints. In a 2x2 crossover design, this is the within-subject standard deviation; in a parallel design, it represents the treatment group's standard deviation, assumed to be the same for both test and reference.
`true.diff`	Numeric or vector. Assumed true difference between test and reference for each endpoint. If a single value is provided, it is applied uniformly across all endpoints.

equi.tol	Numeric. Equivalence margin; the equivalence interval is defined as (-equi.tol, +equi.tol).
design	Character. Study design, either "22co" for a 2x2 crossover design or "parallel" for a parallel groups design.
alpha	Numeric. Significance level for the hypothesis test.
adjust	Character. Method for multiplicity adjustment: "no" (none), "bon" (Bonferroni), "k" (Mielke's weak k-adjustment), or "t" (Mielke's strong k-out-of-m adjustment; legacy "pc" aliases are accepted).
seed	Integer. Random seed for reproducibility.
nsim	Integer. Number of simulations to run for power estimation (default: 10,000).

Details

This function uses the method proposed by Mielke et al. (2018) to estimate the sample size required to achieve the desired power level in studies with multiple correlated endpoints. The function iteratively increases sample size until the target power is reached or the maximum allowable sample size (Nmax) is exceeded. The approach accounts for endpoint correlation and supports adjustments for multiple testing using various correction methods.

Value

An object of class `simss_mielke`, containing:

"power.a" Achieved power with the estimated sample size.

"SS" Required sample size per sequence to achieve the target power.

References

Mielke, J., Jones, B., Jilma, B. & König, F. Sample Size for Multiple Hypothesis Testing in Biosimilar Development. Statistics in Biopharmaceutical Research 10, 39–49 (2018).

Examples

```
# Example 1 from Mielke
sampleSize_Mielke(power = 0.8, Nmax = 1000, m = 5, k = 5, rho = 0,
                  sigma = 0.3, true.diff =  log(1.05), equi.tol = log(1.25),
                  design = "parallel", alpha = 0.05, adjust = "no",
                  seed = 1234, nsim = 100)
```

`simParallelEndpoints` *Generate Simulated Endpoint Data for Parallel Group Design*

Description

Generate simulated endpoint data for a parallel design, with options for normal and lognormal distributions.

Usage

```
simParallelEndpoints(
  n,
  mu.arithmetic,
  mu.geometric = NULL,
  Sigma,
  CV = NULL,
  seed,
  dist = "normal"
)
```

Arguments

`n`	Integer. The sample size for the generated data.
`mu.arithmetic`	Numeric vector. The arithmetic mean of the endpoints on the original scale.
`mu.geometric`	Numeric vector. The geometric mean of the endpoints on the original scale. Only used if `dist = "lognormal"`.
`Sigma`	Matrix. Variance-covariance matrix of the raw data on the original scale. If `dist = "lognormal"`, this matrix is transformed to the log scale.
`CV`	Numeric vector. Coefficient of variation (CV) of the raw data. Only used when `dist = "lognormal"`, where it is transformed to the log scale.
`seed`	Integer. Seed for random number generation, ensuring reproducibility.
`dist`	Character. Assumed distribution of the endpoints: either `"normal"` or `"lognormal"`.

Value

A matrix of simulated endpoint values for a parallel design, with dimensions `n` by the number of variables in `mu.arithmetic` or `mu.geometric`.

Author(s)

Thomas Debray <tdebray@fromdatatowisdom.com>

simPower *Estimate power at a fixed sample size*

Description

Calculates simulated power for a prespecified sample size. The outcome family is selected through `distribution`, matching the unified `sampleSize()` interface.

Usage

```
simPower(
  n,
  distribution = c("norm", "lnorm", "pois", "nbinom"),
  mu_list = NULL,
  varcov_list = NA,
  sigma_list = NA,
  cor_mat = NA,
  sigmaB = 0,
  rate_list = NULL,
  exposure = 1,
  dispersion = 0.1,
  Eper = c(0, 0),
  Eco = c(0, 0),
  rho = 0,
  TAR = NULL,
  arm_names = NA,
  ynames_list = NA,
  type_y = NA,
  list_comparator = NA,
  list_y_comparator = NA,
  alpha = 0.05,
  lequi.tol = NA,
  uequi.tol = NA,
  list_lequi.tol = NA,
  list_uequi.tol = NA,
  dtype = "parallel",
  ctype = "ROM",
  vareq = TRUE,
  k = NA,
  adjust = "no",
  dropout = NA,
  nsim = 5000,
  seed = 1234,
  ncores = 1,
  keep_sim_data = FALSE,
  .warn_redundant_bon = TRUE
)
```

Arguments

n	Integer sample size or vector of sample sizes used for the simulation. For parallel studies this is the base sample size used to derive arm sizes; for 2x2 studies it is the number per sequence. A vector returns a simpower_curve object and enables plotting power across base sample sizes.
distribution	Outcome distribution using R's names: "norm", "lnorm", "pois", or "nbinom" (case-insensitive). Longer labels such as "normal", "lognormal", and "poisson" are also accepted.
mu_list	Named list of continuous-outcome means per arm.
varcov_list	Optional list of covariance matrices for continuous outcomes.
sigma_list	Optional list of standard-deviation vectors for continuous outcomes.
cor_mat	Optional endpoint correlation matrix. For count outcomes, this is also the endpoint correlation matrix used by the joint count engine.
sigmaB	Between-subject parameter for a 2x2 design.
rate_list	Named arm-rate list for count outcomes.
exposure	Count exposure per subject. This can be a scalar or endpoint vector shared by arms, or a named list of arm-specific values.
dispersion	Negative-binomial dispersion. The per-subject negative-binomial size is 1 / dispersion; parallel-arm totals use size n / dispersion. This can be a scalar or endpoint vector shared by arms, or a named list of arm-specific values.
Eper	Period effects for a 2x2 design.
Eco	Carry-over effects for a 2x2 design.
rho	Common endpoint correlation when cor_mat is not supplied.
TAR	Treatment allocation rates for continuous parallel designs.
arm_names	Optional arm names.
ynames_list	Optional endpoint names by arm.
type_y	Endpoint hierarchy for sequential testing for continuous and count outcomes. Use 1 for primary/co-primary and 2 for secondary endpoints.
list_comparator	Named list of treatment-reference comparisons. Each element must be c(test, reference); the first arm is the test arm and the second arm is the reference arm.
list_y_comparator	Endpoint selections by comparison. For count outcomes, the selected endpoints are used to define the count multiplicity and the effective k; in a joint count analysis all comparisons must currently use the same selected endpoint set.
alpha	One-sided significance level.
lequi.tol	Common lower equivalence bound.
uequi.tol	Common upper equivalence bound.
list_lequi.tol	Comparator-specific lower bounds.
list_uequi.tol	Comparator-specific upper bounds.

dtype	Trial design: "parallel" or "2x2".
ctype	Test type. Use "DOM" (difference of means) or "ROM" (ratio of means) for Normal and Lognormal outcomes, and "RR" (event-rate ratio) for Poisson and Negative Binomial outcomes.
vareq	Whether variances are assumed equal for continuous outcomes.
k	Number of endpoints required per comparison.
adjust	Multiplicity adjustment: "no" (none), "bon" (Bonferroni across all selected endpoints), "sid" (Sidak), "k" (the existing weak K-fold rule), "t" (Mielke's strong k-out-of-m adjustment, using alpha / (m - k + 1); legacy "pc" aliases are accepted), or "seq" (sequential hierarchy).
dropout	Dropout proportions. For count 2x2 studies, supply two sequence-specific values.
nsim	Number of simulated trials.
seed	Random seed.
ncores	Number of computation cores. For continuous outcomes this is passed to the compiled simulation backend; for count outcomes it splits Monte Carlo trials into independent seeded chunks whose C++ results are combined.
keep_sim_data	Logical. If TRUE, retain model-scale observations for every simulated trial in sim_data for distribution diagnostics. Defaults to FALSE because retained data can be large.
.warn_redundant_bon	Logical. If TRUE, warn when a requested multiplicity adjustment is redundant or uncalibrated.

Details

For Normal and Log Normal outcomes, supply the continuous-outcome inputs (mu_list, sigma_list or varcov_list, and optionally cor_mat or rho). For Poisson and Negative Binomial outcomes, supply rate_list, list_comparator, list_lequi.tol, and list_uequi.tol. Count exposure and dispersion may be scalar, endpoint-specific, or named arm-specific lists. Arguments for the other outcome family are not used. The returned object supports summary(), confint(), and plot(). The effective endpoint count is determined separately for each comparator from list_y_comparator (or from the endpoints common to both arms when the argument is omitted). k is checked against that count. The function warns when an endpoint-wise adjustment is unnecessary because all selected endpoints are required, and when adjust = "no" is used for a k < m decision. type_y is used with adjust = "seq" for both continuous and count outcomes; for other adjustments it is ignored with a warning. For a k-of-m decision, adjust = "t" allocates alpha over the m - k + 1 boundary endpoints relevant to the strong k-out-of-m null. The legacy adjust = "pc" label is accepted as an alias. Comparisons always use the order supplied in list_comparator: the first arm is the test and the second arm is the reference. Thus, c("T", "R") gives T - R for DOM, T / R for ROM, and the event-rate ratio rate_T / rate_R for RR. Reversing the two names reverses the estimand. For count outcomes, the estimand is the event-rate ratio lambda_T / lambda_R. The equivalence hypotheses are H0: lambda_T / lambda_R <= L or lambda_T / lambda_R >= U versus H1: L < lambda_T / lambda_R < U, assessed by TOST. The same interval hypotheses apply to continuous mean differences (DOM) or mean ratios (ROM), with the log-normal ROM analysis performed on the log scale.

Value

An object containing estimated power and its 95% Monte Carlo confidence interval. The unified function returns primary class `simpower` for every distribution. If `keep_sim_data = TRUE`, the object also contains long-format model-scale observations in `sim_data`. Count results additionally inherit from the compatibility class `countpower`.

Examples

```
simPower(n = 100, distribution = "Poisson",
      rate_list = list(TEST = .21, REF = .20),
      list_comparator = list(TEST_vs_REF = c("TEST", "REF")),
      list_lequi.tol = list(TEST_vs_REF = .80),
      list_uequi.tol = list(TEST_vs_REF = 1.25),
      exposure = 10, nsim = 100, seed = 1)
```

`SimTOST` *Sample Size Estimation via Simulation*

Description

SimTOST: A Package for Sample Size Simulations

Details

The SimTOST package provides tools for simulating sample sizes, calculating power, and assessing type-I error for various statistical scenarios.

Planning

Each function name links to its full help page.

- `sampleSize`: simulation-based sample-size planning for continuous and count outcomes, including multiple endpoints and comparator families.
- `simPower`: simulated power for a fixed sample size, with support for continuous and count-outcome analyses.
- `sampleSize_Mielke`: Mielke et al.'s sample-size calculation for multiple, correlated hypotheses and k-out-of-m rules.

Continuous outcomes

- `simParallelEndpoints`: generates correlated normal or log-normal endpoint data for a parallel-group design.
- `get_par`: prepares and validates endpoint, covariance, allocation, and hierarchy parameters for planning.
- `run_simulations`: dispatches a continuous simulation to the selected design and test combination.

Simulation-result methods

- print.simss (print): prints a concise design, power, confidence-interval, and sample-size report.
- summary.simss (summary): returns and prints a structured summary of the simulation result.
- confint.simss (confint): extracts the stored Monte Carlo confidence interval for achieved power.
- plot.simss (plot): plots simulated power against sample size with confidence intervals and the target-power line.
- update.simss and update.simpower (update): reruns a result while replacing only explicitly supplied planning or simulation parameters.

Author(s)

Thomas Debray <tdebray@fromdatatowisdom.com> (author and maintainer)

Other contributors:

- Tim Friede <tim.friede@med.uni-goettingen.de> [contributor]
- Johanna Munoz <johanna.munoz@fromdatatowisdom.com> [contributor]
- Dewi Amaliah <dewi.amaliah@fromdatatowisdom.com> [contributor]
- Wei Wei <wei.wei@biogen.com> [contributor]
- Marian Mitroiu <marian.mitroiu@biogen.com> [contributor]
- Scott McDonald <scott.mcdonald@fromdatatowisdom.com> [contributor]
- Biogen Inc [copyright holder, funder] (original implementation v1.0.0)
- Smart Data Analysis and Statistics B.V. [copyright holder, funder] (subsequent developments)

References

Mielke, J., Jones, B., Jilma, B. & König, F. Sample Size for Multiple Hypothesis Testing in Biosimilar Development. *Statistics in Biopharmaceutical Research* 10, 39–49 (2018).

See Also

Useful links:

- https://smartdata-analysis-and-statistics.github.io/SimTOST/

summary.countpower *Summarize count-outcome power results*

Description

Summarize count-outcome power results

Usage

```
## S3 method for class 'countpower'
summary(object, ...)
```

Arguments

object	An object of class countpower.
...	Unused additional arguments.

Value

A data frame containing the design, model, sample size, estimated power, and Monte Carlo confidence interval.

summary.countss *Summary for Count Sample-Size Results*

Description

Prints the same design-oriented sample-size report used for continuous outcomes. Count-specific result fields and search histories are retained in the object, while the invisible return value is a data frame of the selected per-arm (or per-sequence) and total sample sizes.

Usage

```
## S3 method for class 'countss'
summary(object, ...)
```

Arguments

object	A countss object returned by a count sample-size calculation or by the unified sampleSize() function.
...	Unused additional arguments.

Value

Invisibly, a data frame containing the selected sample size for each arm (or sequence) and the total sample size.

summary.simpower *Summarize fixed-sample-size power results*

Description

Summarize fixed-sample-size power results

Usage

```
## S3 method for class 'simpower'
summary(object, ...)
```

Arguments

object	An object returned by simPower().
...	Unused additional arguments.

Value

A data frame containing the distribution, design, comparison, estimand, hypotheses, sample size, estimated power, and Monte Carlo confidence interval.

summary.simss *Summary for Simulation Results*

Description

Generates a summary of the simulation results, including per-arm and total sample sizes. The printed summary also states the equivalence null and alternative hypotheses for the selected outcome and design.

Usage

```
## S3 method for class 'simss'
summary(object, ...)
```

Arguments

object	An object of class "simss" returned by a sampleSize function.
...	Additional arguments (currently unused).

Value

Invisibly, a data frame containing the estimated sample size for each arm (or sequence), plus the total sample size. The printed report also includes the design, estimand, equivalence margins, target power, achieved power, and Monte Carlo interval. Count-outcome results use this same report and retain their count-specific fields in the returned object.

Author(s)

Johanna Muñoz <johanna.munoz@fromdatatowisdom.com>

Examples

```
## Not run:
res <- sampleSize(mu_list = list(T = c(y1 = 1), R = c(y1 = 1)),
                  sigma_list = list(T = c(y1 = .2), R = c(y1 = .2)),
                  list_comparator = list(c("T", "R")),
                  list_lequi.tol = list(c(y1 = .8)),
                  list_uequi.tol = list(c(y1 = 1.25)),
                  ctype = "ROM", distribution = "lnorm", nsim = 10,
                  lower = 2, upper = 4)
summary(res)

## End(Not run)
```

---

test_2x2_dom *Simulate a 2x2 Crossover Design and Compute Difference of Means (DOM)*

Description

Simulates a two-sequence, two-period (2x2) crossover design and evaluate equivalence for the difference of means (DOM).

Usage

```
test_2x2_dom(
  n,
  muT,
  muR,
  SigmaW,
  lequi_tol,
  uequi_tol,
  alpha,
  sigmaB,
  dropout,
  Eper,
  Eco,
  typey,
  adseq,
  k,
  arm_seed
)
```

Arguments

n	integer number of subjects per sequence
muT	vector mean of endpoints on treatment arm
muR	vector mean of endpoints on reference arm
SigmaW	matrix within subject covar-variance matrix across endpoints
lequi_tol	vector lower equivalence tolerance band across endpoints
uequi_tol	vector upper equivalence tolerance band across endpoints
alpha	vector alpha value across endpoints
sigmaB	double between subject variance (assumed same for all endpoints)
dropout	vector of size 2 with dropout proportion per sequence (0,1)
Eper	vector of size 2 with period effect on period (0,1)
Eco	vector of size 2 with carry over effect of arm c(Reference, Treatment).
typey	vector with positions of primary endpoints
adseq	boolean is used a sequential adjustment?
k	integer minimum number of equivalent endpoints
arm_seed	seed for the simulation

Value

A numeric matrix containing the simulated hypothesis test results. The first column represents the overall equivalence decision, where 1 indicates success and 0 indicates failure. The subsequent columns contain the hypothesis test results for each endpoint, followed by mean estimates for the reference and treatment groups, and standard deviations for the reference and treatment groups.

---

test_2x2_rom *Simulate a 2x2 Crossover Design and Compute Ratio of Means (ROM)*

Description

Simulates a two-sequence, two-period (2x2) crossover design and evaluate equivalence for the ratio of means (ROM).

Usage

```
test_2x2_rom(
  n,
  muT,
  muR,
  SigmaW,
  lequi_tol,
  uequi_tol,
  alpha,
```

```
  sigmaB,
  dropout,
  Eper,
  Eco,
  typey,
  adseq,
  k,
  arm_seed
)
```

Arguments

n	integer number of subjects per sequence
muT	vector mean of endpoints on treatment arm
muR	vector mean of endpoints on reference arm
SigmaW	matrix within subject covar-variance matrix across endpoints
lequi_tol	vector lower equivalence tolerance band across endpoints
uequi_tol	vector upper equivalence tolerance band across endpoints
alpha	vector alpha value across endpoints
sigmaB	double between subject variance (assumed same for all endpoints)
dropout	vector of size 2 with dropout proportion per sequence (0,1)
Eper	vector of size 2 with period effect on period (0,1)
Eco	vector of size 2 with carry over effect of arm c(Reference, Treatment).
typey	vector with positions of primary endpoints
adseq	boolean is used a sequential adjustment?
k	integer minimum number of equivalent endpoints
arm_seed	seed for the simulation

Value

A numeric matrix containing the simulated hypothesis test results. The first column represents the overall equivalence decision, where 1 indicates success and 0 indicates failure. The subsequent columns contain the hypothesis test results for each endpoint, followed by mean estimates for the reference and treatment groups, and standard deviations for the reference and treatment groups.

---

test_par_dom *Simulate a Parallel Design and Test Difference of Means (DOM)*

Description

Simulates a parallel-group design and performs equivalence testing using the difference of means (DOM) approach. This function evaluates whether the treatment and reference groups are equivalent based on predefined equivalence margins and hypothesis testing criteria.

Usage

```
test_par_dom(
  n,
  muT,
  muR,
  SigmaT,
  SigmaR,
  lequi_tol,
  uequi_tol,
  alpha,
  dropout,
  typey,
  adseq,
  k,
  arm_seedT,
  arm_seedR,
  TART,
  TARR,
  vareq
)
```

Arguments

n	integer number of subjects per arm
muT	vector mean of endpoints on treatment arm
muR	vector mean of endpoints on reference arm
SigmaT	matrix covar-variance matrix on treatment arm across endpoints
SigmaR	matrix covar-variance matrix on reference arm across endpoints
lequi_tol	vector lower equivalence tolerance band across endpoints
uequi_tol	vector upper equivalence tolerance band across endpoints
alpha	vector alpha value across endpoints
dropout	vector of size 2 with dropout proportion per arm (T,R)
typey	vector with positions of primary endpoints
adseq	boolean is used a sequential adjustment?
k	integer minimum number of equivalent endpoints
arm_seedT	integer seed for the simulation on treatment arm
arm_seedR	integer seed for the simulation on reference arm
TART	double treatment allocation rate for the treatment arm
TARR	double treatment allocation rate for the reference arm
vareq	boolean assumed equivalence variance between arms for the t-test

Details

The function simulates a parallel-group study design and evaluates equivalence using the difference of means (DOM) approach. It accounts for dropout rates and treatment allocation proportions while generating simulated data based on the specified covariance structure. The test statistics are computed, and a final equivalence decision is made based on the predefined number of required significant endpoints (`k`). If sequential testing (`adseq`) is enabled, primary endpoints must establish equivalence before secondary endpoints are evaluated. When `vareq = TRUE`, the test assumes equal variances between groups and applies Schuirmann's two one-sided tests (TOST).

Value

A numeric matrix containing the simulated hypothesis test results. The first column represents the overall equivalence decision, where 1 indicates success and 0 indicates failure. The subsequent columns contain the hypothesis test results for each endpoint, followed by mean estimates for the reference and treatment groups, and standard deviations for the reference and treatment groups.

`test_par_rom` *Simulate a Parallel Design and Test Ratio of Means (ROM)*

Description

Simulates a parallel-group design and performs equivalence testing using the ratio of means (ROM) approach. This function evaluates whether the treatment and reference groups are equivalent based on predefined equivalence margins and hypothesis testing criteria.

Usage

```
test_par_rom(
  n,
  muT,
  muR,
  SigmaT,
  SigmaR,
  lequi_tol,
  uequi_tol,
  alpha,
  dropout,
  typey,
  adseq,
  k,
  arm_seedT,
  arm_seedR,
  TART,
  TARR,
  vareq
)
```

Arguments

n	integer number of subjects per arm
muT	vector mean of endpoints on treatment arm
muR	vector mean of endpoints on reference arm
SigmaT	matrix covar-variance matrix on treatment arm across endpoints
SigmaR	matrix covar-variance matrix on reference arm across endpoints
lequi_tol	vector lower equivalence tolerance band across endpoints
uequi_tol	vector upper equivalence tolerance band across endpoints
alpha	vector alpha value across endpoints
dropout	vector of size 2 with dropout proportion per arm (T,R)
typey	vector with positions of primary endpoints
adseq	boolean is used a sequential adjustment?
k	integer minimum number of equivalent endpoints
arm_seedT	integer seed for the simulation on treatment arm
arm_seedR	integer seed for the simulation on reference arm
TART	double treatment allocation rate for the treatment arm
TARR	double treatment allocation rate for the reference arm
vareq	Boolean. If TRUE, assumes equal variance between arms and applies Schuirmann's two one-sided tests (TOST) for equivalence using a pooled variance.

Details

The function simulates a parallel-group study design and evaluates equivalence using the ratio of means (ROM) approach. It accounts for dropout rates and treatment allocation proportions while generating simulated data based on the specified covariance structure. The test statistics are computed, and a final equivalence decision is made based on the predefined number of required significant endpoints (k). If sequential testing (adseq) is enabled, primary endpoints must establish equivalence before secondary endpoints are evaluated. When vareq = TRUE, the test assumes equal variances between groups and applies Schuirmann's two one-sided tests (TOST).

Value

A numeric matrix containing the simulated hypothesis test results. The first column represents the overall equivalence decision, where 1 indicates success and 0 indicates failure. The subsequent columns contain the hypothesis test results for each endpoint, followed by mean estimates for the reference and treatment groups, and standard deviations for the reference and treatment groups.

type1Error *Empirical Type I Error at the Least-Favorable Null*

Description

Generates a boundary-null configuration and estimates its empirical rejection probability using simPower(). The comparator convention is c(test, reference). For normal ROM and count RR, a lower-bound null sets mean(test) / mean(reference) = L. For log-normal ROM, the boundary is imposed on the log-analysis scale used by the test; this coincides with the arithmetic mean ratio when the two arms have the same coefficient of variation.

Usage

```
type1Error(
  null = c("lower", "upper", "both"),
  x = NULL,
  comparator = NULL,
  endpoint = NULL,
  joint = FALSE,
  conf.level = 0.95,
  ...
)
```

Arguments

null	Boundary to evaluate: "lower", "upper", or "both".
x	Optional existing simss or simpower object. When supplied, its design and outcome parameters are reused; arguments in ... override stored settings such as n or nsim.
comparator	Optional comparator name identifying the boundary scenario when joint = FALSE. It must be one of the names in list_comparator; by default, the first comparator is used. It is ignored when joint = TRUE, because joint mode evaluates all comparators.
endpoint	Optional endpoint identifying the boundary component when joint = FALSE. It is available for the all-endpoints-required case (k equal to the number of endpoints), where one endpoint is placed on the boundary. For k smaller than the number of endpoints, use joint = TRUE so that all composite boundary configurations are evaluated.
joint	Logical. If TRUE, evaluate every comparator and every endpoint partial-null configuration and return the joint Type I-error result for each complete-trial decision. For a comparator with m endpoints and an at-least-k rule, all boundary counts from m - k + 1 through m are evaluated. Defaults to FALSE for backward compatibility.
conf.level	Confidence level for the simultaneous one-sided Monte Carlo upper bound across the evaluated joint scenarios. Defaults to 0.95.
...	Arguments passed to simPower(). The call must include the outcome parameters, comparator definitions, and equivalence margins.

Details

Comparators use the convention `c(test, reference)`. For ROM, the tested estimand is `test / reference`; for DOM it is `test - reference`. With log-normal ROM, the test is performed after converting the supplied original-scale means and variances to the log-analysis scale, so the boundary scenario is calibrated on that scale. For count rate ratios, the analogous midpoint is `sqrt(L * U)` on the rate-ratio scale because the test is performed on the log-rate-ratio scale. Absolute rates and dispersion remain nuisance parameters, so this midpoint should be supplemented by a grid or optimization when a global supremum is required. If a comparator has `m` endpoints and requires `k` endpoints to pass, a composite null configuration has at least `m - k + 1` non-equivalent endpoints. With `joint = TRUE`, all endpoint subsets with boundary counts from `m - k + 1` through `m`, and all lower/upper direction combinations, are evaluated, while the complete decision still requires the `k`-of-`m` rule for every comparator in the same simulated trial. The returned joint object includes a Bonferroni simultaneous one-sided Monte Carlo upper bound for the maximum scenario probability.

Value

For `null = "lower"` or `"upper"`, an object of class `type1error` containing the empirical Type I error in `type1_error` and its Monte Carlo interval. For `null = "both"`, a named list containing the lower- and upper-bound results. With `joint = TRUE`, a `type1error_joint` object containing one joint result for every valid comparator, endpoint-subset, boundary-direction combination is returned.

`update.countpower` *Update a standalone count power calculation*

Description

Update a standalone count power calculation

Usage

```
## S3 method for class 'countpower'
update(object, ..., evaluate = TRUE)
```

Arguments

`object`	A standalone `countpower` object returned by a count power function.
`...`	Arguments to replace in the original calculation.
`evaluate`	If `FALSE`, return the updated call instead of evaluating it.

Value

A newly calculated count power object, or an unevaluated call.

update.countss *Update a standalone count sample-size calculation*

Description

Update a standalone count sample-size calculation

Usage

```
## S3 method for class 'countss'
update(object, ..., evaluate = TRUE)
```

Arguments

object	A standalone countss object returned by a count sample-size function.
...	Arguments to replace in the original calculation.
evaluate	If FALSE, return the updated call instead of evaluating it.

Value

A newly calculated count sample-size object, or an unevaluated call.

update.simpower *Update a SimTOST fixed-sample-size power calculation*

Description

Update a SimTOST fixed-sample-size power calculation

Usage

```
## S3 method for class 'simpower'
update(object, ..., evaluate = TRUE)
```

Arguments

object	A simpower object, including unified count-outcome results.
...	Arguments to replace in the original calculation.
evaluate	If FALSE, return the updated call instead of evaluating it.

Value

A newly calculated simpower object, or an unevaluated call.

`update.simss` *Update a SimTOST sample-size calculation*

Description

Re-runs the calculation represented by `object`, replacing only arguments supplied in `....` This is useful for changing, for example, the target power or number of simulations without repeating the complete original call.

Usage

```
## S3 method for class 'simss'
update(object, ..., evaluate = TRUE)
```

Arguments

`object`	A `simss` object, including count-outcome sample-size objects.
`...`	Arguments to replace in the original calculation.
`evaluate`	If `FALSE`, return the updated call instead of evaluating it.

Value

A newly calculated object of the same result family as `object`, or an unevaluated call when `evaluate = FALSE`.

Index

.plot_correlation, 3
.power_count_joint_serial, 4
.power_count_serial, 6
[.simss_mielke, 8

as.numeric.simss_mielke, 9

confint.countpower, 9
confint.countss *(confint.countpower)*, 9
confint.simpower, 10
confint.simss, 10, *49*

get_par, 11, *48*

plot.countpower, 12
plot.countss, 13
plot.simpower, 14
plot.simpower_curve, 14
plot.simss, 15, *49*
plot.type1error, 16
plot.type1error_joint, 17
plot_decision_heatmap, 17
plot_distribution, 18
plot_mc_error, 19
plot_stability, 20
power_dom, 21
print.countpower, 22
print.countss, 22
print.simpower, 23
print.simss, 23, *49*
print.simss_mielke, 24

run_simulations, 24, *48*
run_simulations_2x2_dom, 25
run_simulations_2x2_rom, 26
run_simulations_par_dom, 28
run_simulations_par_rom, 30

sampleSize, 31, *48*
sampleSize(), *13*
sampleSize_count, 37
sampleSize_count_joint, 39
sampleSize_Mielke, 42, *48*
simParallelEndpoints, 44, *48*
simPower, 45, *48*
simPower(), *10*, *12*, *14, 15*, *23*, *51*, *58*
SimTOST, 48
SimTOST-package *(SimTOST)*, 48
stats::confint(), *9, 10*
summary.countpower, 50
summary.countss, 50
summary.simpower, 51
summary.simss, *49*, 51

test_2x2_dom, 52
test_2x2_rom, 53
test_par_dom, 54
test_par_rom, 56
type1Error, 58
type1Error(), *16*

update.countpower, 59
update.countss, 60
update.simpower, *49*, 60
update.simss, *49*, 61