Package 'ssutil'

September 17, 2026

Title Sample Size Calculation Tools

Version 1.2.0

Description Functions for sample size estimation and simulation in clinical trials. Includes methods for selecting the best group using the Indifference-zone approach, as well as designs for non-inferiority, equivalence, and negative binomial models. For the sample size calculation for non-inferiority of vaccines, the approach is based on Fleming, Powers, and Huang (2021) <doi:10.1177/1740774520988244>. The Indifference-zone approach is based on Sobel and Huyett (1957) <doi:10.1002/j.1538-7305.1957.tb02411.x> and Bechhofer, Santner, and Goldsman (1995, ISBN:978-0-471-57427-9).

License AGPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1)

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

Language en-US

Imports stats, stringr, broom, MASS, mvtnorm, tibble

Config/testthat/edition 3

VignetteBuilder knitr

URL https://johnaponte.github.io/ssutil/, https://github.com/johnaponte/ssutil

BugReports https://github.com/johnaponte/ssutil/issues

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author John J. Aponte [aut, cre] (ORCID: <https://orcid.org/0000-0002-3014-3673>), Chris Gast [ctb]

Maintainer John J. Aponte <john.j.aponte@gmail.com>

Repository CRAN

Date/Publication 2026-09-17 17:50:02 UTC

Contents

`empirical_power_result`

Create an Empirical Power Result object

Description

Constructs an S3 object of class `empirical_power_result`, storing the estimated power, its confidence interval, and the number of simulations used to compute it.

Usage

```
empirical_power_result(x, n, conf.level = 0.95)
```

Arguments

`x`	Number of successes
`n`	Number of trials.
`conf.level`	Confidence level for the returned confidence interval power.

Details

It is a wrap to `binom.test`

Value

An object of class `empirical_power_result`, a list with components:

- `power`: Estimated power.
- `conf.low`: Lower bound of confidence interval.
- `conf.high`: Upper bound of confidence interval.
- `conf.level`: Confidence level for the returned confidence interval.
- `nsim`: Number of simulations.

Examples

```
result <- empirical_power_result(
  x = 10,
  n = 100,
  conf.level = 0.95
)
print(result)
```

`format.power_events_rate` *Format method for power_events_rate class*

Description

Format method for power_events_rate class

Usage

```
## S3 method for class 'power_events_rate'
format(x, digits = 1, ...)
```

Arguments

`x`	an R object of class power_events_rate
`digits`	a positive integer indicating how many decimal digits are to be used to display the probability columns as percentages.
`...`	further arguments passed to or from other methods

Value

A character string with a markdown-style table of the probabilities of observing each event threshold, by sample size and risk.

format.power_single_rate *Format method for power_single_rate class*

Description

Format method for power_single_rate class

Usage

```
## S3 method for class 'power_single_rate'
format(x, digits = 3, ...)
```

Arguments

x	an R object of class power_single_rate
digits	a positive integer indicating how many significant digits are to be used for numeric x.
...	further arguments passed to or from other methods

Value

A character string with a human-readable summary of the detection power or a markdown-style table, depending on the number of rows.

is.empirical_power_result *Check if an object is a sim_power_result*

Description

Check if an object is a sim_power_result

Usage

```
is.empirical_power_result(x)
```

Arguments

x	Any R object.

Value

Logical. TRUE if x inherits from "sim_power_result".

`multp` *Multivariate Normal Equicoordinate Cumulative Distribution Function*

Description

Computes the joint CDF for a multivariate standard normal distribution with unit variances and a common correlation coefficient `rho`. It is the exact functional inverse of `multz`: `multp(multz(q, k, rho), k, rho) == q`.

Usage

```
multp(q, k, rho, lower.tail = TRUE, seed = NULL)
```

Arguments

`q`	Numeric. Quantile of the distribution.
`k`	Integer. Number of variables in the multivariate normal distribution. Must be >= 1.
`rho`	Numeric. Common correlation coefficient between variables (typically between 0 and 1).
`lower.tail`	Logical. If `TRUE` (default), probability is $P(X \leq q)$; if `FALSE`, $P(X > q)$. Mirrors `pnorm`.
`seed`	Optional. An object specifying if and how the random number generator should be initialized. Passed to `pmvnorm`.

Value

Numeric. The joint cumulative probability.

Examples

```
q <- 1.3
k <- 3
rho <- 0.5
multp(q, k, rho)
```

multz *Calculate the Upper Equicoordinate Point of a Multivariate Normal Distribution*

Description

Computes the upper equicoordinate quantile for a multivariate standard normal distribution with unit variances and a common correlation coefficient `rho`. That is, it returns the value z such that the joint probability $P(X_1 \leq z, \ldots, X_n \leq z) = p$.

Usage

```
multz(p, k, rho, lower.tail = TRUE, seed = NULL)
```

Arguments

`p`	Numeric. Cumulative probability (e.g., 0.95 for the 95th
`k`	Integer. Number of variables in the multivariate normal distribution. Must be >= 1.
`rho`	Numeric. Common correlation coefficient between variables (typically between 0 and 1).
`lower.tail`	Logical. If `TRUE` (default), probability is $P(X \leq z)$; if `FALSE`, $P(X > z)$
`seed`	Optional. An object specifying if and how the random number generator should be initialized. Passed to `qmvnorm`.

Value

Numeric. The equicoordinate quantile z

Examples

```
p <- 0.9    # Significance level (10%)
k <- 3      # Number of variables
rho <- 0.5  # Common correlation coefficient
multz(p, k, rho)
```

`power_best_binomial` *Power to Correctly Select the Best Group in a Binomial Test*

Description

Computes the exact probability of correctly identifying the best group when the outcome follows a binomial distribution. It assumes that `p1` is the probability of success in the best group, and that the success probability in all other groups is lower by a fixed difference `dif`.

Usage

```
power_best_binomial(p1, dif, ngroups, npergroup)
```

Arguments

`p1`	Numeric. Probability of success in the best group (must be in [0, 1]).
`dif`	Numeric. Difference in success probability between the best group and the next best (must be > 0).
`ngroups`	Integer. Number of groups (must be greater than 1).
`npergroup`	Integer. Number of subjects per group (must be positive).

Details

The formula is based on the exact method described by Sobel and Huyett (1957).

Value

A numeric value representing the probability of correctly identifying the best group.

References

Sobel, M., & Huyett, M. J. (1957). Selecting the Best One of Several Binomial Populations. *Bell System Technical Journal*, 36(2), 537–576. doi:10.1002/j.15387305.1957.tb02411.x

Examples

```
power_best_binomial(p1 = 0.8, dif = 0.2, ngroups = 4, npergroup = 50)
```

power_best_normal *Power calculation for the Indifferent-zone approach for normal outcomes*

Description

Estimate the probability of correctly select the best group among ngroups groups if the difference between the best group and the next best is at least dif(the Indifferent-Zone), and the standard deviation is sd

Usage

```
power_best_normal(dif, sd, ngroups, npergroup, seed = NULL)
```

Arguments

dif	Numeric. Indifferent-zone. Minimum difference that is considered meaningful.
sd	Numeric. Common standard deviation of the response variable.
ngroups	Integer. Number of groups (treatments) being compared.
npergroup	Integer. Number in each group.
seed	Optional. Integer seed to use in the internal call to multp().

Value

Numeric. Probability of correctly selecting the best group.

Note

The function uses multp(), which computes the joint cumulative distribution for the selection procedure. This implementation assumes equal variances and independent samples.

Examples

```
power_best_normal(dif = 0.5, sd = 1, ngroups = 3, npergroup = 11)
```

power_events_rate *Probability of Observing At Least a Given Number of Events*

Description

Computes the exact binomial probability of observing at least `e` events, for every combination of sample size and risk, and for one or more event thresholds.

Usage

```
power_events_rate(n, r, e)
```

Arguments

`n`	Integer or vector of integers. Sample size(s).
`r`	Numeric or vector of numerics. Risk(s) (per-subject event probability), between 0 and 1.
`e`	Integer or vector of integers. Event count threshold(s), e.g. 1, 2, 3.

Value

A matrix of class `power_events_rate` with columns:

N Sample size

Risk Per-subject event probability

>= e One column per threshold in `e`, named using the "greater than or equal to" symbol followed by the threshold value, giving P(X >= e) for X ~ Binomial(N, Risk)

Examples

```
power_events_rate(30, 0.1, 1:3)
power_events_rate(c(30, 60), c(0.05, 0.1), c(1, 2, 3))
```

power_single_rate *Detectable Event Rate with Specified Power and Sample Size*

Description

Estimates the minimum true proportion of events needed to detect at least one event, given a sample size and desired statistical power.

Usage

```
power_single_rate(subjects, power)
```

Arguments

subjects	Integer or vector of integers. Sample size(s).
power	Numeric or vector of numerics. Desired power(s), between 0 and 1.

Value

A matrix of class power_single_rate with columns:

n Sample size

power Requested power

proportion Minimum detectable event rate to observe at least one event

Examples

```
power_single_rate(30, 0.9)
power_single_rate(c(30, 50, 100), 0.9)
```

---

print.empirical_power_result

Print method for empirical_power_result

Description

Nicely formats the output of an object of class empirical_power_result, showing the power estimate, confidence interval, and number of simulations.

Usage

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

Arguments

x	An object of class "empirical_power_result".
...	Further arguments passed to or from other methods (ignored).

Value

Invisibly returns the object passed in.

`print.power_events_rate` *Print method for class power_events_rate*

Description

Print method for class power_events_rate

Usage

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

Arguments

`x`	an object of class power_events_rate
`...`	further arguments passed to or from other methods

Value

Invisibly returns the object passed in.

`print.power_single_rate` *Print method for class power_single_rate*

Description

Print method for class power_single_rate

Usage

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

Arguments

`x`	an object of class power_single_rate
`...`	further arguments passed to or from other methods

Value

Invisibly returns the object passed in.

prophr *Calculate Event Probability in the Experimental Group Given a Hazard Ratio*

Description

Computes the event probability in the experimental group based on the event probability in the control group and a specified hazard ratio, assuming proportional hazards.

Usage

```
prophr(p0, hr)
```

Arguments

p0	Numeric scalar or vector. Probability of an event in the control group (between 0 and 1).
hr	Numeric scalar or vector. Hazard ratio (must be > 0). Must be the same length as p0.

Details

This is useful for sample size calculations, for example in PASS (TM), which does not automatically adjust the event rate for the experimental group.

Value

Numeric. The probability of an event in the experimental group.

Examples

```
prophr(0.05, 0.6)
```

sim_power_best_binomial *Simulate Power to Select the Best Group Using Binomial Outcomes*

Description

Estimates the empirical power to correctly identify the best group as having the highest outcome, under a binomial distribution. Assumes that the most promising group has a higher success probability than the others by at least dif, and that outcomes are independent.

Usage

```
sim_power_best_binomial(
  noutcomes,
  p1,
  dif,
  ngroups,
  npergroup,
  nsim,
  conf.level = 0.95
)
```

Arguments

noutcomes	Integer. Number of outcomes to evaluate.
p1	Numeric. Probability in the most promising group (scalar or vector).
dif	Numeric. Difference between the best group and the rest.
ngroups	Integer. Number of groups.
npergroup	Integer or vector. Number of subjects per group.
nsim	Integer. Number of simulations.
conf.level	Numeric. Confidence level for the empirical power estimate

Details

Multiple outcomes can be evaluated simultaneously. The power is estimated as the proportion of simulations where the most promising group is selected best in all outcomes.

Value

An S3 object of class `empirical_power_result`, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also `empirical_power_result`.

See Also

`empirical_power_result`

Examples

```
sim_power_best_binomial(
  noutcomes = 1,
  p1 = 0.7,
  dif = 0.2,
  ngroups = 3,
  npergroup = 30,
  nsim = 1000
)
```

`sim_power_best_bin_rank` *Simulate Power to Rank the Best Group Using Binomial Outcomes*

Description

Estimates the empirical power to rank the most promising group as the best, based on binomial outcomes, via simulation.

Usage

```
sim_power_best_bin_rank(
  noutcomes,
  p1,
  dif,
  weights,
  ngroups,
  npergroup,
  nsim,
  conf.level = 0.95
)
```

Arguments

`noutcomes`	Integer. Number of outcomes to evaluate.
`p1`	Numeric. Event probability in the first group, i.e. the true best group (scalar or vector of length `noutcomes`).
`dif`	Numeric. Amount by which the first (true best) group's probability exceeds the other `ngroups - 1` groups (scalar or vector of length `noutcomes`).
`weights`	Numeric vector. Weights for each outcome. If scalar, applied equally.
`ngroups`	Integer. Number of groups.
`npergroup`	Integer or vector. Sample size per group.
`nsim`	Integer. Number of simulations.
`conf.level`	Numeric. Confidence level for the empirical power estimate#'

Details

Each outcome is assumed to follow an independent binomial distribution. The first group is always the true best group: it is simulated with event probability `p1`, while the other `ngroups - 1` groups share probability `p1 - dif`. The function sums weighted ranks across multiple outcomes to determine the top group, and estimates the empirical power to correctly identify the first group as the best.

If multiple outcomes are defined, weights can be applied to prioritize some outcomes over others. Weights are automatically scaled to sum 1. For each outcome, groups are ranked from lowest (1)

to highest (ngroups) observed proportion; the group with the highest total weighted rank across outcomes is considered the best. Power is the proportion of simulations in which that group is the first group.

Value

An S3 object of class empirical_power_result, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also empirical_power_result.

See Also

empirical_power_result

Examples

```
sim_power_best_bin_rank(
noutcomes = 2,
p1 = 0.80,
dif = 0.15,
weights = 1,
ngroups = 3,
npergroup = 30,
nsim = 1000,
conf.level = 0.95)
```

---

sim_power_best_normal *Simulate Power to Select Best Group (Normal Outcomes)*

Description

Estimates the empirical power to identify the most promising group as the best, when outcomes are normally distributed and independent.

Usage

```
sim_power_best_normal(
  noutcomes,
  sd,
  dif,
  ngroups,
  npergroup,
  nsim,
  conf.level = 0.95
)
```

Arguments

noutcomes	Integer. Number of outcomes to evaluate.
sd	Numeric vector. Standard deviations for each outcome. Can be a single value.
dif	Numeric vector. Difference in means between the best and the other groups.
ngroups	Number of groups to compare.
npergroup	Number of subjects per group. Can be scalar or vector of length ngroups.
nsim	Integer. Number of simulations to perform.
conf.level	Numeric. Confidence level for the empirical power estimate

Details

The best group (group 1) is assumed to have mean 0, and the rest of the groups have mean -dif.

Multiple outcomes can be evaluated simultaneously. The power is estimated as the proportion of simulations where the most promising group is the best in all outcomes.

The number of subjects per group can be the same or specified per group. In either case, the first group is assumed to be the most promising.

Value

An S3 object of class empirical_power_result, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also empirical_power_result.

See Also

empirical_power_result

Examples

```
sim_power_best_normal(
 noutcomes = 2,
 sd = c(1, 1.2),
 dif = c(0.2, 0.25),
 ngroups = 3,
 npergroup = c(30, 25, 25),
 nsim = 1000
)
```

`sim_power_best_norm_rank` *Simulate Power to Select Best Group by Ranks (Normal Outcomes)*

Description

Estimates the empirical power to identify the most promising group as best, using weighted ranks across outcomes, assuming normally distributed outcomes.

Usage

```
sim_power_best_norm_rank(
  noutcomes,
  sd,
  dif,
  weights,
  ngroups,
  npergroup,
  nsim,
  conf.level = 0.95
)
```

Arguments

`noutcomes`	Integer. Number of outcomes to evaluate.
`sd`	Numeric vector. Standard deviations for each outcome.
`dif`	Numeric vector. Difference in means between the best and other groups.
`weights`	Numeric vector. Weights per outcome.
`ngroups`	Integer. Number of groups.
`npergroup`	Integer or vector. Number of subjects per group.
`nsim`	Integer. Number of simulations.
`conf.level`	Numeric. Confidence level for the empirical power estimate

Details

Each outcome is independent and normally distributed. The most promising group is assumed to have a mean at least `dif` higher than the others. Ranks are weighted and summed per group across outcomes.

If `weights` is specified, it is internally scaled to sum to 1. The most promising group is always considered to be the first group.

Value

An S3 object of class `empirical_power_result`, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also `empirical_power_result`.

See Also

empirical_power_result

Examples

```
sim_power_best_norm_rank(
  noutcomes = 3,
  sd = c(1, 0.8, 1.5),
  dif = c(0.2, 0.15, 0.3),
  weights = c(0.5, 0.3, 0.2),
  ngroups = 3,
  npergroup = c(30, 25, 25),
  nsim = 1000
)
```

---

sim_power_equivalence_normal

Empirical Power for Equivalence (Normal Outcomes)

Description

Estimates the empirical power to detect equivalence among multiple groups assuming no true difference in normally distributed outcomes. Pairwise two-sample t-tests are used, and equivalence is declared if all confidence intervals for differences between group means lie entirely within the interval defined by `llimit` and `ulimit`.

Usage

```
sim_power_equivalence_normal(
  ngroups,
  npergroup,
  sd,
  llimit,
  ulimit,
  nsim,
  t_level = 0.95,
  conf.level = 0.95
)
```

Arguments

`ngroups`	Integer. Number of groups to compare
`npergroup`	Integer. Number of observations per group.
`sd`	Numeric. Standard deviation of the outcome distribution (common across groups).
`llimit`	Numeric. Lower equivalence limit.
`ulimit`	Numeric. Upper equivalence limit.

nsim	Integer. Number of simulations to perform.
t_level	Numeric. Confidence level used for the t-tests (e.g., 0.95 for 95% CI). Equivalence is assessed via a two-sided t_level CI, which is equivalent to two one-sided tests (TOST) each at a one-sided alpha of (1 - t_level) / 2. The default t_level = 0.95 therefore implements TOST at a one-sided alpha of 0.025; for the more common bioequivalence convention of one-sided alpha = 0.05 (e.g. FDA/EMA guidance), use t_level = 0.90 instead.
conf.level	Numeric. Confidence level for the empirical power estimate

Details

This function simulates data under the null hypothesis of no difference between groups and calculates the proportion of simulations in which all pairwise comparisons fall within the specified equivalence limits.

Value

An S3 object of class empirical_power_result, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also empirical_power_result.

See Also

empirical_power_result

Examples

```
#Equivalence testing for three groups with log-scale outcome
sim_power_equivalence_normal(
  ngroups = 3,
  npergroup = 172,
  sd = 0.403,
  llimit = log10(2/3),
  ulimit = log10(3/2),
  nsim = 1000,
  t_level = 0.95
)
```

---

sim_power_nbinom *Empirical Power for Negative Binomial Comparison*

Description

Estimates empirical power to detect a relative risk either above or below a specified boundary, depending on the direction of the alternative hypothesis. Simulates count data with over dispersion, fits a model with glm.nb, and evaluates the power to reject the null hypothesis using a negative binomial model.

Usage

```
sim_power_nbinom(
  n1,
  n2,
  ir1,
  tm,
  rr,
  boundary,
  dispersion,
  alpha,
  direction = c("less", "greater"),
  nsim,
  conf.level = 0.95
)
```

Arguments

n1	Integer. Number of participants in group 1.
n2	Integer. Number of participants in group 2.
ir1	Numeric. Incidence rate in group 1.
tm	Numeric. Average exposure time per subject (assumed equal across subjects).
rr	Numeric. True relative risk between groups (group 2 rate = rr × group 1 rate).
boundary	Numeric. Relative risk boundary under the null hypothesis.
dispersion	Numeric. Dispersion parameter (ϕ) for the negative binomial distribution.
alpha	Numeric. Type I error rate (two-sided).
direction	= Direction of the alternative hypothesis
nsim	Integer. Number of simulation iterations.
conf.level	Numeric. Confidence level for the empirical power estimate

Value

An S3 object of class `empirical_power_result`, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also `empirical_power_result`.

Note

Uses the alternative parameterization of the negative binomial: `mu` is the mean, and `size = 1/dispersion`. In `glm.nb`, dispersion is estimated as `theta`. The 'boundary' parameter defines the relative risk under the null hypothesis. When rr < 1, rejection occurs if the upper limit of the confidence interval is below the boundary. When rr > 1, rejection occurs if the lower limit is above the boundary.

The `alpha` parameter is two-sided as it is used to estimate two-sided confidence intervals

Direction of the alternative hypothesis: "less" rejects when the CI upper bound of the simulation is below `boundary` (e.g. non-inferiority, RR bounded above); "greater" rejects when the CI lower bound of the simulation is above `boundary` (RR bounded below). Must be specified explicitly.

Author(s)

Chris Gast

John J. Aponte

See Also

empirical_power_result

Examples

```
sim_power_nbinom(
 n1 = 150, n2 = 150,
 ir1 = 0.55, tm = 1.7,
 rr = 0.6, boundary = 1,
 dispersion = 2,
 alpha = 0.05,
 direction = "less",
 nsim = 1000
)
```

---

sim_power_ni_normal *Empirical Power for Non-Inferiority (Normal Outcomes)*

Description

Estimates empirical power to declare non-inferiority between two groups across multiple outcomes using t-tests. Simulates normally distributed data under the null (no difference) and applies non-inferiority rules based on user-defined required and optional tests.

Usage

```
sim_power_ni_normal(
  nsim,
  npergroup,
  ntest,
  ni_limit,
  test_req,
  test_opt,
  sd,
  true_diff = 0,
  corr = 0,
  t_level = 0.95,
  conf.level = 0.95
)
```

Arguments

nsim	Integer. Number of simulations to perform.
npergroup	Integer. Number of observations per group.
ntest	Integer. Number of tests (outcomes) to compare.
ni_limit	Numeric. Limit to declare non-inferiority. Can be a scalar or vector of length ntest.
test_req	Integer. Number of required tests that must show non-inferiority (first test_req tests).
test_opt	Integer. Number of optional tests that must also show non-inferiority from the remaining tests.
sd	Numeric. Standard deviation(s) of the outcomes. Scalar or vector of length ntest.
true_diff	Numeric. True difference(s) in means (experimental minus control) used to simulate the data, i.e. $\mu_E - \mu_C$. Scalar or vector of length ntest. Defaults to 0 (no true difference between groups); set to a small negative value to evaluate power when the experimental group is truly slightly worse than control, or a positive value when it is truly better.
corr	Numeric. Correlation between the tests. Scalar (common correlation), or vector of length ntest*(ntest-1)/2.
t_level	Numeric. Confidence level used for the t-tests (e.g., 0.95 for 95% CI).scalar or vector of length ntest.
conf.level	Numeric. Confidence level for the empirical power estimate

Details

A test is considered non-inferior if the lower bound of its confidence interval is greater than the specified non-inferiority limit. Overall non-inferiority is declared if all test_req and at least test_opt of the remaining tests are non-inferior.

Value

An S3 object of class empirical_power_result, which contains the estimated empirical power and its confidence interval. The object can be printed, formatted, or further processed using associated S3 methods. See also empirical_power_result.

Note

If only one test is used, correlation is ignored.

Use correlation 0 for independent outcomes

When using a correlation vector, it must match the number of test pairs: ntest*(ntest-1)/2, in this order: (1,2), (1,3), ..., (1,ntest), (2,3), ..., (ntest-1,ntest).

The covariance matrix is derived from the correlation matrix and the standard deviations.

For example: with ntest = 3 and corr = c(0.2, 0.3, 0.4), the resulting correlation matrix is:

	[,1]	[,2]	[,3]
[1,]	1	0.2	0.3
[2,]	0.2.	1	0.4
[3,]	0.3.	0.4	1

See Also

empirical_power_result

Examples

```
sim_power_ni_normal(
  nsim = 1000,
  npergroup = 250,
  ntest = 7,
  ni_limit = log10(2/3),
  test_req = 2,
  test_opt = 3,
  sd = 0.4,
  corr = 0,
  t_level = 0.95
)
```

---

ss_best_binomial *Sample Size to Select the Best Group in a Binomial Test*

Description

Computes the minimum sample size per group required to achieve a target probability of correctly selecting the best group in a binomial test. The best group is assumed to have success probability `p1`, and the other groups have `p1 - dif`.

Usage

```
ss_best_binomial(power, p1, dif, ngroups, max_n = 1000)
```

Arguments

`power`	Numeric. Desired probability of correctly selecting the best group (in [0, 1]).
`p1`	Numeric. Probability of success in the best group (in [0, 1]).
`dif`	Numeric. Difference in success probability with the next best group (> 0).
`ngroups`	Integer. Number of groups (must be > 1).
`max_n`	Integer. Maximum sample size to evaluate (default is 1000).

Details

The function searches for the smallest npergroup such that the power from power_best_binomial is at least the target power.

Value

An integer representing the minimum sample size per group required to reach the specified power.

Examples

```
ss_best_binomial(power = 0.9, p1 = 0.8, dif = 0.2, ngroups = 4)
```

---

ss_best_normal *Sample Size for Selecting the Best Treatment in a Normal Response (Indifference-Zone)*

Description

Calculates the minimum common sample size per group needed to achieve a specified probability (power) of correctly selecting the best group using the indifference-zone approach. This method assumes normally distributed responses with a known and common standard deviation.

Usage

```
ss_best_normal(power, dif, sd, ngroups, seed = NULL)
```

Arguments

power	Numeric. Desired probability of correctly selecting the best group.
dif	Numeric. Indifferent-zone. Minimum difference that is considered meaningful.
sd	Numeric. Common standard deviation of the response variable.
ngroups	Integer. Number of groups (treatments) being compared.
seed	Optional. Integer seed to use in the internal call to multz().

Details

The indifference-zone approach guarantees that the probability of correct selection is at least power, assuming the best group's mean exceeds the others by at least dif. The calculation is based on Bechhofer's Procedure Nb.

Value

Integer. Sample size required per group to achieve the specified power.

Note

The function uses the quantile function `multz()`, which computes critical values for the selection procedure. This implementation assumes equal variances and independent samples.

References

Bechhofer, R.E., Santner, T.J., & Goldsman, D.M. (1995). *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons.* Wiley Series in Probability and Statistics. ISBN: 0-471-57427-9.

Examples

```
ss_best_normal( power = 0.8, dif = 0.5, sd = 1, ngroups = 3)
```

---

ss_ni_ve *Sample Size and Non-Inferiority Margin for Vaccine Efficacy Trials*

Description

Computes the non-inferiority margin, number of events, and maximum hazard ratio (HR) to declare non-inferiority in vaccine efficacy (VE) trials, based on the approach described by Fleming et al. (2021).

Usage

```
ss_ni_ve(
  ve_lci,
  alpha = 0.025,
  power = 0.9,
  use70 = FALSE,
  preserve = 0.5,
  ve_exp = NULL,
  ve_ac = NULL
)
```

Arguments

`ve_lci`	Numeric. Lower bound of the current vaccine's efficacy (e.g., 0.95 for 95% VE).
`alpha`	Numeric. Type I error rate (default = 0.025).
`power`	Numeric. Desired power for the test (default = 0.90).
`use70`	Logical. If `TRUE`, assumes at least 30% VE for the new vaccine (the 90–70 rule); otherwise, preserves a fixed fraction of the reference VE.
`preserve`	Numeric. Proportion of the current vaccine's efficacy to preserve under `use70 = FALSE` (default = 0.5).

ve_exp — Numeric or NULL (default). Assumed vaccine efficacy of the experimental vaccine versus placebo, used to compute power for the non-inferiority trial. If NULL (the default), the experimental vaccine is assumed to have the same true efficacy as the active comparator (hazard ratio of 1 between them). If specified, ve_ac must also be provided.

ve_ac — Numeric or NULL (default). Point estimate of the active comparator's vaccine efficacy versus placebo (as opposed to ve_lci, which is its confidence interval lower bound). Only used, and required, when ve_exp is specified.

Details

The method applies either the 95–95 rule or 90–70 rule, depending on whether a minimum VE of 30% is assumed (use70 = TRUE) or 50% of the current VE is preserved.

This implementation approximates Tables 1 and 2 of the paper: the total number of events uses Freedman's (1982) log-rank sample size formula, generalized to allow the true experimental-to-comparator hazard ratio assumed for power (ve_exp) to differ from 1; the maximum hazard ratio to declare non-inferiority uses an exact binomial confidence interval via binom.test.

Value

A named list with:

- Upper limit of the HR used to estimate the sample size: Hazard ratio corresponding to ve_lci.
- Non-inferior margin in HR scale: Non-inferiority margin expressed as a hazard ratio.
- Alpha: The type I error used.
- Power: The power used.
- Total number of events: Total number of events required in the trial.
- Max HR to declare NI: Maximum observed hazard ratio that satisfies the non-inferiority criterion.
- Max number of events in the experimental group: Maximum number of events in the experimental group still compatible with non-inferiority.
- Non-inferior criteria: Description of the applied non-inferiority rule ("At least 30% VE" or "or preserved effect").

References

Fleming, T.R., Powers, J.H., & Huang, Y. (2021). The use of active controls and non-inferiority studies in evaluating COVID-19 vaccines. *Clinical Trials*, 18(3), 335–342. doi:10.1177/1740774520988244

Freedman, L.S. (1982). Tables of the number of patients required in clinical trials using the logrank test. *Statistics in Medicine*, 1(2), 121-129. doi:10.1002/sim.4780010204

Examples

```
# Table 1: rule out margin delta, assuming EXP has the same true efficacy
# as AC (175-event active-comparator trial, 95% VE, HR upper 95% CI = 0.0855)
ss_ni_ve(ve_lci = 1 - 0.0855, power = 0.9)
```

```
# The 90-70 rule (use70 = TRUE) rules out the more lenient margin delta0
# instead of delta, for the same active-comparator trial as above
ss_ni_ve(ve_lci = 1 - 0.0855, power = 0.9, use70 = TRUE)

# Table 2: rule out margin delta0, assuming the experimental vaccine has a
# fixed 60% efficacy versus placebo regardless of AC's own efficacy (here,
# AC has 60% VE, HR upper 95% CI = 0.4997)
ss_ni_ve(ve_lci = 1 - 0.4997, power = 0.9, use70 = TRUE, ve_exp = 0.60, ve_ac = 0.60)
```

---

tidy.empirical_power_result *Tidy Method for empirical_power_result*

Description

Creates a one-row tibble with the power estimate and confidence interval.

Usage

```
## S3 method for class 'empirical_power_result'
tidy(x, ...)
```

Arguments

x	A empirical_power_result object.
...	Ignored.

Value

A tibble with columns: power, conf.low, conf.high, conf.level. nsim.

---

wcs_power_best_binomial *Worst-Case Scenario Power for the Best Binomial Group*

Description

Searches for the probability in the best-performing group that yields the lowest statistical power, given an indifference zone specification, a number of groups, and a number of subjects per group.

Usage

```
wcs_power_best_binomial(dif, ngroups, npergroup)
```

Arguments

dif	Numeric. Indifference zone specification (difference threshold).
ngroups	Integer. Number of groups to compare.
npergroup	Integer. Number of subjects per group.

Details

Defines an internal function fx that wraps power_best_binomial with the supplied parameters, then uses optimize over the interval [0,1] to find the probability p1 that minimizes the resulting power.

Value

A named list with components:

p1 Numeric. Probability in the best group that yields the minimum power.

minimum_power Numeric. The minimum power achieved at p1.

See Also

power_best_binomial, optimize

Examples

```
wcs_power_best_binomial(dif = 0.1, ngroups = 3, npergroup = 50)
```

Index