Package ‘extraSuperpower’

October 24, 2025

Title Power Calculation for Two-Way Factorial Designs

Version 1.6.0

Author Louis Macias [aut, cre, cph] (ORCID = <https://orcid.org/0000-0002-3080-2835>), Silke Szymczak [aut] (ORCID = <https://orcid.org/0000-0002-8897-9035>)

Maintainer Louis Macias <louis.macias@uni-luebeck.de>

Description The basic use of this package is with 3 sequential functions. One to generate expected cell means and standard deviations, along with correlation and covariance matrices in the case of repeated measurements. This is followed by experiment simulation i number of times. Finally, power is calculated from the simulated data. Features that may be considered in the model are interaction, measure correlation and non-normal distributions.

Encoding UTF-8

RoxygenNote 7.3.3

Imports MASS (>= 7.3.0), afex (>= 1.3.0), rlang, Matrix, rlist, ggplot2, plyr, reshape2, scales, ggthemes, fGarch, truncnorm, sn, tmvtnorm, ARTool, permuco, methods, stats, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0), linpk, Superpower

VignetteBuilder knitr, rmarkdown

Config/testthat/edition 3

License MIT + file LICENSE

URL https://github.com/luisrmacias/extraSuperpower

BugReports https://github.com/luisrmacias/extraSuperpower/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2025-10-24 16:00:08 UTC

Contents

`calculate_mean_matrix` *Create input for simulation based two-way factorial experiments*

Description

This function will generate a matrix of expected mean values for *ab* factor level combinations of a two-way factorial design by assuming linear effects with possible departure from linearity by interaction. It will also provide a standard deviation matrix for these *ab* combinations of factor levels. If the design has repeated measures, it will additionally provide correlation and covariance matrices calculated depending on which factor has repeated measurements or is the 'within' factor.

Usage

```
calculate_mean_matrix(
  refmean,
  nlfA,
  nlfB,
  fAeffect,
  fBeffect,
  groupswinteraction = NULL,
  interact = 1,
  label_list = NULL,
  sdproportional = TRUE,
  sdratio = 0.2,
  endincrement = TRUE,
  rho = 0,
  withinf = NULL,
  plot = TRUE
)
```

Arguments

`refmean`	Numeric - expected mean for first level of both factors A and B
`nlfA`	Integer - number of levels of factor A
`nlfB`	Integer - number of levels of factor B

`fAeffect` Numeric - multiple that defines how cell means are modified by factor A. With the default `endincrement` (`TRUE`), determines the last level of factor A with respect to its first level. When `endincrement=FALSE` this multiple applies from one level to the next.

`fBeffect` Numeric - multiple that defines how cell means are modified by factor B. With the default `endincrement` (`TRUE`), determines the last level of factor B with respect to its first level. When `endincrement=FALSE` this multiple applies from one level to the next.

`groupswinteraction` Vector length 2 or n*2 matrix - Combination of levels from factors A and B in which interaction is expected

`interact` Numeric - value by which the mean from cell or cells indicated in `groupswinteraction` is multiplied after it has been calculated accordingly to `fAeffect` and `fBeffect`

`label_list` List length 2 - vectors with the names of the factor levels. The objects in this list should be named as the factors. The use of this option is encouraged as these names are used for plotting and inherited to downstream functions.

`sdproportional` Logical - whether the standard deviation for each combination of factor levels is a proportion of the respective factor level combination mean, defaults to `TRUE`

`sdratio` Numeric - value by which the expected mean value of a factor level combination is multiplied to obtain the respective standard deviation, defaults to 0.2.

`endincrement` Logical - determines if the multiples provided in `fAeffect` and `fBeffect` refer to change between first and last levels (default) or level to level changes.

`rho` Vector length 1 or 2, or 2 by 2 matrix - Controls how the correlation and hence de covariance matrix is built. See 'details' and `?gencorrelationmat` examples.

`withinf` Character - Names the factor with repeated measures. Possible values are NULL, "fA", "fB" or "both"

`plot` Logical - Should a line plot with the modeled mean and standard deviations be part of the output. Default is `TRUE`

Details

The user must provide a reference mean (usually mean in control or untreated group), the expected change for each factor from first to last level (or from one level to the next) and the number of levels in each factor.

The user can also specify factor level combinations in which interaction is assumed and its magnitude with respect to the reference mean. The cell mean matrix will be modified accordingly and this can also have an effect of the standard deviation matrix.

We were motivated by sample size calculation for two-way factorial designs with *1,2,...,a* levels of factor *A* and *1,2,...,b* levels of factor *B* in which the mean outcome value for replicates of cell *A=1, B=1* are known. Furthermore, there is an expected change in level mean for each of the factors. Finally, interaction can be explicitly introduced to level combinations in which it is expected to occur.

If a repeated measures experiment is intended `withinf` must be set to "fA", "fB" or "both", depending on which is the 'within' factor. If `rho` is a vector length 1, the within subject correlation will be constant for the factor defined in `withinf`. If `rho` is a vector length 2 and `withinf` is either "fA" or "fB" a correlation gradient will be created from the first to second value of rho. If rho is a vector length 2 and withinf="both", the first element of rho will be the correlation within factor A, while the second element will be the correlation within factor B. If rho is a 2*2 matrix, only possible if withinf="both", a correlation gradient will be created across rows of rho for each of the factors.

Value

If rho and withinf are left at their default values of 0 and NULL, respectively, a cell mean matrix, a cell standard deviation matrix and optionally a graph that represents both.

If rho is between -1 and 1 but different to 0 and withinf is either "fA", "fB" or "both", correlation and covariance matrices are generated along with the aforementioned output.

Examples

```
refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85
factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)
## Independent design
effects_treat_time <- calculate_mean_matrix(refmean = refmean,
                                            fAeffect = treateff, fBeffect = timeeff,
                                            nlfA = treatgroups, nlfB = timepoints,
                                            label_list = factors_levels_names)
## Inspect plot to check if matrices correspond to design
effects_treat_time$meansplot
n <- 20
independent_experiment <- twoway_simulation_independent(group_size = n,
                                      matrices_obj = effects_treat_time)

head(independent_experiment, 10)

## Repeated measures design, suppose subjects from 4 independent treatment groups measured
## at 5 different timepoints.
## We use the same parameters as the independent design example, except we add within factor level
## correlation and we specify that factor B is the within factor.


refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85
rho <- 0.8
withinf <- "fB"

factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

effects_treat_time <- calculate_mean_matrix(refmean = refmean, fAeffect = treateff,
                      fBeffect = timeeff, nlfA = treatgroups,  nlfB = timepoints,
                      rho = rho, withinf = withinf, label_list = factors_levels_names)
```

```
## Plot should look the same, structure within data can be checked once simulated
effects_treat_time$meansplot

n <- 20
repeatedmeasures_experiment <- twoway_simulation_correlated(group_size = n,
                                           matrices_obj = effects_treat_time)
head(repeatedmeasures_experiment, 10)
```

---

effsize *Effect size calculation*

Description

Calculate effect sizes for two-way factorial designs from matrices of expected mean and standard deviation values at each combination of factor levels. The output given is Cohen's f. Calculations are done as exemplified in the G*Power 3.1 manual.

Usage

```
effsize(matrices_obj)
```

Arguments

matrices_obj	List of 2 matrices, named mean.mat and sd.mat. This is the minimal output of the `calculate_mean_matrix` function. The full output from `calculate_mean_matrix` is also valid.

Value

Vector of length 3. The first two elements are the effect sizes for the main effects factor A and factor B, respectively. The third element is the interaction effect size.

Examples

```
# no interaction effect expected
refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85

factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

effects_treat_time <- calculate_mean_matrix(refmean = refmean,
                                            nlfA = treatgroups, nlfB = timepoints,
                                            fAeffect = treateff, fBeffect = timeeff,
                                            label_list = factors_levels_names)
```

```
effsize(effects_treat_time)

# we add cell specific interaction effect keeping design and main effect coefficients
cellswithinteraction <- matrix(c(rep(2,3), 3:5), 3,2)
#second level of factor A interacts with 3rd, 4th and 5th level of factor B

effects_treat_time_interact <- calculate_mean_matrix(refmean = refmean,
                                                   nlfA = treatgroups, nlfB = timepoints,
                                                 fAeffect = treateff, fBeffect = timeeff,
                                                      label_list = factors_levels_names,
                                               groupswinteraction = cellswithinteraction,
                                                      interact=1.3)
effsize(effects_treat_time_interact)
```

`exact_twoway_anova_power` *Two-way factorial ANOVA exact sample size calculation for independent samples*

Description

This functions takes the effect sizes (Cohen's f) for two main effects and their interaction and estimates power a range of sample sizes. The input for this function can be generated by `effsize`.

Usage

```
exact_twoway_anova_power(
  a,
  b,
  effect_sizes,
  n,
  alpha = 0.05,
  factor_names = NULL,
  plot = TRUE,
  title = NULL,
  target_power = NULL,
  target_line = TRUE,
  alpha_line = TRUE
)
```

Arguments

a	Number of levels of the first factor
b	Number of levels of the second factor

effect_sizes
: Numeric vector of length 3. The first two elements are the effect sizes for the main effects of the first and second factors, respectively. The third element is the interaction effect size.

n
: Number of experimental units in each group for which power (1-beta) will be calculated.

alpha
: numeric - Type I error probability. Defaults to 0.05

factor_names
: character - vector of length 2. Names of the 2 factors to be evaluated. Default is to inherit names from effect_sizes. If effect_sizes has no names and no factor_names are provided, factors will be named 'FactorA' and 'FactorB'.

plot
: logical - Should the power curve be plotted. Default is TRUE.

title
: character - Title for the graph. Defaults to 'Power curve from exact ANOVA test'

target_power
: numeric - Desired power to be attained. Accepts values between 0 and 1, defaults to 0.8.

target_line
: logical - Set to TRUE. If FALSE no target line will be drawn. Overrides target_power.

alpha_line
: logical - Should a line at the set type I error be plotted

Details

Probably the best way to calculate power for independent balanced designs

Value

A list that contains the number of levels for each factor, the chosen significance level and a data.frame in which the first column is the group sample size and the remaining three columns are the power for the main effect of the first factor, the main effect of the second factor and their interaction, respectively.

Optionally, a graph that displays the power curves.

Examples

```
refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.25
timeeff <- 0.85
cellswithinteraction <- matrix(c(rep(2,3), 3:5), 3,2)
#second level of factor A interacts with 3rd, 4th and 5th level of factor B

factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

effects_treat_time_interact <- calculate_mean_matrix(refmean = refmean,
                                                      nlfA = treatgroups, nlfB = timepoints,
                                                    fAeffect = treateff, fBeffect = timeeff,
                                                         label_list = factors_levels_names,
                                                  groupswinteraction = cellswithinteraction,
                                                         interact=1.3)
```

```
fxs <- effsize(effects_treat_time_interact)
exact_twoway_anova_power(a= treatgroups, b=timepoints, effect_sizes=fxs, n=5:20)
```

`gencorrelationmat` *Function that generates a correlation matrix taking as input number of factors for each level, factor or factors that present correlation and rho value or values. Additionally, a mean matrix is required to check consistency.*

Description

May be run independently or internally as part of `calculate_mean_matrix`.

Usage

```
gencorrelationmat(mean_matrix, rho, label_list = NULL, withinf, nlfA, nlfB)
```

Arguments

`mean_matrix`	Matrix - cell mean value matrix
`rho`	Vector length 1 or 2, or 2 by 2 matrix - Controls how the correlation and hence de covariance matrix is built. See details.
`label_list`	List length 2 - Names of factor levels
`withinf`	Character- Factor for which measurements are repeated, options are NULL, "fA", "fB" and "both". If NULL (default) independent measurements will be considered.
`nlfA`	Integer - number of levels of factor A
`nlfB`	Integer - number of levels of factor B

Details

For a repeated measures experiment `withinf` must be set to "fA", "fB" or "both", depending on which is the 'within' factor. If `rho` is a vector length 1, the within subject correlation will be constant for the factor defined in `withinf`. If `rho` is a vector length 2 and `withinf` is either "fA" or "fB" a correlation gradient will be created from the first to second value of `rho`. If `rho` is a vector length 2 and `withinf="both"`, the first element of `rho` will be the correlation within factor A, while the second element will be the correlation within factor B. If `rho` is a 2*2 matrix, only possible if `withinf="both"`, a correlation gradient will be created across rows of `rho` for each of the factors.

Value

Correlation matrix

Examples

```
meanvals <- c(seq(3,9,2),seq(2,8,2),seq(1,7,2))
mean_matrix <- matrix(meanvals, 3, 4, byrow = TRUE,
                  dimnames = list(A=LETTERS[1:3], B=letters[1:4]))

mean_matrix

gencorrelationmat(mean_matrix = mean_matrix, rho = 0.7, withinf = "fB", nlfA = 3, nlfB = 4)

##correlation gradient over levels of factor B
gencorrelationmat(mean_matrix = mean_matrix, rho = c(0.7, 0.4), withinf = "fB", nlfA = 3, nlfB = 4)

##gradient both factors

rhovals <- matrix(c(0.7, 0.4), 2, 2, byrow = TRUE)
gencorrelationmat(mean_matrix = mean_matrix, rho = rhovals, withinf = "both", nlfA = 3, nlfB = 4)
```

`gencovariancemat` *Function that generates a covariance matrix taking as input a correlation matrix and a standard deviation matrix or value.*

Description

May be run independently or internally as part of 'calculate_mean_matrix'.

Usage

```
gencovariancemat(
  correlation_matrix,
  sd_matrix,
  withinf,
  label_list = NULL,
  nlfA,
  nlfB
)
```

Arguments

`correlation_matrix`	Matrix - Expected correlation between combinations of factor levels
`sd_matrix`	Numeric or matrix - Standard deviation value or matrix of standard deviation values for combinations of factor levels.
`withinf`	Character- Factor for which measurements are repeated, options are NULL, "fA", "fB" and "both". If NULL (default) independent measurements will be considered.
`label_list`	List length 2 - Names of factor levels
`nlfA`	Integer - number of levels of factor A
`nlfB`	Integer - number of levels of factor B

Value

Covariance matrix

Examples

```
meanvals <- c(seq(3,9,2),seq(2,8,2),seq(1,7,2))
mean_matrix <- matrix(meanvals, 3, 4, byrow = TRUE,
                   dimnames = list(A=LETTERS[1:3], B=letters[1:4]))

mean_matrix
sd_matrix <- mean_matrix*0.2

cor_matrix <- gencorrelationmat(mean_matrix = mean_matrix,
                                rho = 0.7, withinf = "fB", nlfA = 3, nlfB = 4)

gencovariancemat(cor_matrix, sd_matrix, withinf = "fB", nlfA = 3, nlfB = 4)

##correlation gradient over levels of factor B
cor_matrix <- gencorrelationmat(mean_matrix = mean_matrix,
                                rho = c(0.7, 0.4), withinf = "fB", nlfA = 3, nlfB = 4)

gencovariancemat(cor_matrix, sd_matrix, withinf = "fB", nlfA = 3, nlfB = 4)
```

---

graph_twoway_assumptions

Graph modeled means and standard deviations of groups in two-way factorial design

---

Description

Internal function that plots modeled cell means and standard deviations and covariance matrices. Takes input generated by the `calculate_mean_matrix` function and runs inside of it.

Usage

```
graph_twoway_assumptions(group_size = 100, matrices_obj)
```

Arguments

`group_size`	integer - number of subjects in each group
`matrices_obj`	List length 2 - Cell means and standard deviation matrices

Value

Line plot with expected mean and standard deviation for each combination of factor levels

plot_powercurves — *Plots the output of test_twoway_nrange*

Description

Internal function, called by test_twoway_nrange, to plot power against sample size.

Usage

```
plot_powercurves(
  power_over_nrange,
  target_power = NULL,
  title = NULL,
  target_line = TRUE,
  alpha_line = TRUE,
  alpha = 0.05
)
```

Arguments

`power_over_nrange`	data.frame with sample sizes and corresponding powers to be plotted
`target_power`	Numeric. Desired power to be attained. Accepts values between 0 and 1, defaults to 0.8.
`title`	Character. Title for the graph. Defaults to 'Power curve from exact ANOVA test'
`target_line`	Logical. If FALSE no target line will be drawn. Overrides target_power. Default is TRUE.
`alpha_line`	Logical. Should a dashed line at the set alpha level be drawn. Default is TRUE.
`alpha`	Numeric. Type I error rate.

Value

Plot with power curves.

Examples

```
## 'cornorm_model' is created with the calculate_mean_matrix function
refmean <- 10
treateff <- 1.2
timeeff <- 0.75

treatgroups <- 3
treatgroups_names <- c("wt", "DrugA", "DrugB")

timepoints <- 4
timepoints_names <- paste0("t", 1:timepoints)
```

```
nameslist <- list(treatment=treatgroups_names, time=timepoints_names)

rho = 0.7

cornorm_model <- calculate_mean_matrix(refmean = refmean, fAeffect = treateff, fBeffect = timeeff,
nlfA = treatgroups, nlfB = timepoints,
rho = rho, withinf = "fB", label_list = nameslist)

nset <- seq(7, 14, 2)
cornorm_sim <- simulate_twoway_nrange(cornorm_model, nset, repeated_measurements=TRUE, nsims=5)

##used small number of iterations to reduce computation time

power_results <- test_power_overkn(cornorm_sim, test="rank", plot=TRUE)
```

`simulate_twoway_nrange` *Simulated independent and repeated measures two-way experiments over a set of sample sizes*

Description

Wrapper for both independent and repeated measures two-way simulations. A vector of defined sample sizes is simulated under the model provided.

Usage

```
simulate_twoway_nrange(
  matrices_obj,
  nset,
  balanced = TRUE,
  group_size = NULL,
  loss = NULL,
  repeated_measurements = FALSE,
  distribution = "normal",
  skewness = 1,
  shape = 0,
  inferior_limit = -Inf,
  superior_limit = Inf,
  nsims = 200
)
```

Arguments

`matrices_obj` List - Output generated by `calculate_mean_matrix` that include cell mean and standard deviation matrices

`nset`	Vector - If default values are used for both `balanced` and `group_size`, sample sizes to be used in simulations. If `balanced="FALSE"` and a matrix is provided to `group_size`, number to add to all elements of `group_size`.
`balanced`	Logical - Whether the study will be performed with the same number of subjects in all groups. Default is `TRUE`. See 'details'.
`group_size`	Matrix - Sample size for each condition (combination of factor levels). Only to be used when `balanced=FALSE`.
`loss`	Character - Type of selection of subjects in groups that have less observations than `max(group_size)`. Possible values are "random" and "sequential". Ignored if `repeated_measurements=FALSE` or `balanced=TRUE`. See 'details'.
`repeated_measurements`	Logical - Does the design have repeated measurements. Default is false.
`distribution`	Character - Type of distribution to simulate. Possible values are 'normal', 'truncated.normal' or 'skewed'.
`skewness`	Numeric - Momentum of distribution skewness, univariate distribution simulation.
`shape`	Numeric - Degree of skewness in the distribution. May be a single value, have a length equal to the number of levels of any one of the factors or a length equal to the product of the length of each factor. For multivariate distribution simulations.
`inferior_limit`	Numeric - Value of the lower bound for the truncated distribution, defaults to '-Inf'. Ignored if `distribution` is either "normal" or "skewed".
`superior_limit`	Numeric - Value of the upper bound for the truncated distribution, defaults to 'Inf'. Ignored if `distribution` is either "normal" or "skewed".
`nsims`	Integer - Number of iterations

Details

For unbalanced independent measures designs, this function generates a simulation with 'max(group_size)' for all combinations of factors and then eliminates observations at random in those factor combinations that have less participants or study subjects. This is also the behavior for unbalanced repeated measures designs when `loss="random"`.

For unbalanced repeated measures designs in which `loss="sequential"` the participants or subjects from the groups with less observations will be a subset of participants or subjects of groups with more observations. The elimination strategy may not sound like the most efficient way to proceed, is quite fast anyhow.

The 'n' column in the output will reflect how many observations each factor combination has. This should match the input matrix.

Value

List with of `data.frames` of simulated outcome values under different sample sizes. Each data.frame includes factor level labels, iteration number and sample size.

Examples

```
refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85
factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)
## Independent design
effects_treat_time <- calculate_mean_matrix(refmean = refmean,
                                            fAeffect = treateff,fBeffect = timeeff,
                                            nlfA = treatgroups, nlfB = timepoints,
                                            label_list = factors_levels_names)
## Inspect plot to check if matrices correspond to design
effects_treat_time$meansplot
n <- seq(from = 16, to = 24, by = 2)

## In this case, the default 'repeated_measurements', 'distribution' and options are used.
indep_simulation <- simulate_twoway_nrange(effects_treat_time, n)

## Simulate from a truncated distribution
indep_simulation_trunc <- simulate_twoway_nrange(matrices_obj = effects_treat_time, nset = n,
                          distribution="truncated.normal", inferior_limit= 0.8)

##randomly select iteration, select a condition
k <- sample(1:max(indep_simulation_trunc[[1]]$iteration), 1)
toviewdist <- indep_simulation_trunc[[1]]
toviewdist <- subset(toviewdist, iteration==k)
toviewdist <- subset(toviewdist, cond=="V6")
hist(toviewdist$y)
```

`test_power_overkn` *Test simulated two-way factorial design experiments over different sample sizes.*

Description

Wrapper to test data simulated under independent or repeated measurements and under different outcome distributions with different sample sizes. Takes output from `simulate_twoway_nrange` as input, along with test and plotting options.

Usage

```
test_power_overkn(
  data,
  test = "ANOVA",
  plot = TRUE,
  target_power = NULL,
  title = NULL,
```

```
  target_line = TRUE,
  alpha_line = TRUE,
  alpha = 0.05
)
```

Arguments

`data`	data.frame - `data.frame` with modeled outcome values, factor level labels, iteration number and sample size.
`test`	character - Statistical test to be applied, possible values are 'ANOVA', 'rank' and 'permutation'.
`plot`	logical - Should the power curve be plotted. Default is TRUE.
`target_power`	Desired power to be attained. Accepts values between 0 and 1, defaults to 0.8.
`title`	Title for the graph. Defaults to 'Power curve from exact ANOVA test'
`target_line`	Set to TRUE. If FALSE no target line will be drawn. Overrides target_power.
`alpha_line`	• logical Should a line at the set type I error be plotted
`alpha`	• numeric Type I error probability

Value

Data frame with power and confidence intervals for the main effects and interaction for each of the sample sizes. Also presented in graphical form if `plot=TRUE`.

Examples

```
## In this example we simulate an independent sample design with skewed outcome
## Model was specified with the 'calculate_mean_matrix function' (see ?calculate_mean_matrix)
refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.25
timeeff <- 0.85
factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

indep_matrix <- calculate_mean_matrix(refmean = refmean,
                                      fAeffect = treateff, fBeffect = timeeff,
                                      nlfA = treatgroups, nlfB = timepoints,
                                      label_list = factors_levels_names)

indep_skewsim <- simulate_twoway_nrange(indep_matrix, seq(6, 12, 2),
                            distribution = "skewed", skewness = 1.8, nsims=5)
##used low number of iterations to reduce computation time

test_power_overkn(indep_skewsim, test="rank")
```

twoway_simulation_correlated *Simulate measurements repeated over either or both factors of a two-way design*

Description

Both regular and internal function. As regular function takes input generated by the `calculate_mean_matrix` function and iteratively simulates repeated measures two-way factorial experiments. Data are sampled from a normal, skewed normal or truncated normal distribution.

Usage

```
twoway_simulation_correlated(
  group_size,
  matrices_obj,
  distribution = "normal",
  shape = 0,
  inferior_limit = -Inf,
  superior_limit = Inf,
  balanced = TRUE,
  loss = NULL,
  nsims = 200
)
```

Arguments

`group_size`	Integer or matrix - Sample size for each group (combination of factor levels). If `balanced=TRUE` (default) `group_size` must be an integer. If `balanced=FALSE` `group_size` must be a matrix.
`matrices_obj`	List - Output generated by `calculate_mean_matrix` that include cell mean and covariance matrices
`distribution`	Character - Type of distribution from which to sample, possible values are "normal", "skewed" and "truncated"
`shape`	Vector - Degree of skewness in the distribution. May be a single value, have a length equal to the number of levels of any one of the factors or a length equal to the product of the length of each factor.
`inferior_limit`	Numeric - Value for which the distribution is truncated on the left. Only valid if `distribution="truncated.normal"`
`superior_limit`	Numeric - Value for which the distribution is truncated on the right. Only valid if `distribution="truncated.normal"`
`balanced`	Logical - Whether the study will be performed with the same number of subjects in all groups. Default is `TRUE`. See 'details'.
`loss`	Character - Type of selection of subjects in groups that have less observations than `max(group_size)`. Possible values are 'random' and 'sequential'. Ignored if `balanced=TRUE`. See 'details'.

nsims Integer - Number of iterations

Details

As internal function runs with a single iteration inside `graph_twoway_assumptions`, which in itself is inside 'calculate_mean_matrix' to generate data for the cell mean and standard deviation plot.

For unbalanced repeated measures designs, this function generates a simulation with `max(group_size)` for all combinations of factors and then eliminates observations. If `loss="random"` elimination of in those factor combinations that have less participants or study subjects will occur at random. If `loss="sequential"` the participants or subjects from the groups with less observations will be a subset of participants or subjects of groups with more observations. This may not sound like the most efficient way to proceed, is quite fast anyhow.

The 'n' column in the output will reflect how many observations each factor combination has. This should match the input matrix.

Value

Dataframe with simulated outcome values, factor level labels and iteration number.

Examples

```
## Repeated measures design, suppose subjects from 4 independent treatment groups
## measured at 5 different timepoints.

refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85
rho <- 0.8
withinf <- "fB"
factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

effects_treat_time <- calculate_mean_matrix(refmean = refmean,
                                            fAeffect = treateff, fBeffect = timeeff,
                                            nlfA = treatgroups,  nlfB = timepoints,
                                            rho = rho, withinf = withinf,
                                            label_list = factors_levels_names)

## Inspect plot to check if matrices correspond to design
effects_treat_time$meansplot

n <- 20
repeatedmeasures_experiment <- twoway_simulation_correlated(group_size = n,
                                matrices_obj = effects_treat_time)

head(repeatedmeasures_experiment, 10)
```

twoway_simulation_independent

Simulate independent measurements in a two-way factorial design

Description

Both regular and internal function. As regular function takes input generated by the `calculate_mean_matrix` function and iteratively simulates independent measures two-way factorial experiments. Outcome may be normally distributed, have a skewed normal distribution or a truncated normal distribution.

Usage

```
twoway_simulation_independent(
  group_size,
  matrices_obj,
  distribution = "normal",
  skewness = 1,
  inferior_limit = -Inf,
  superior_limit = Inf,
  balanced = TRUE,
  nsims = 200
)
```

Arguments

`group_size`	Integer or matrix - Sample size for each condition (combination of factor levels). If `balanced=TRUE` (default) `group_size` must be an integer. If `balanced=FALSE` `group_size` must be a matrix.
`matrices_obj`	List - Output generated by `calculate_mean_matrix` that include cell mean and standard deviation matrices.
`distribution`	Character - Type of distribution to simulate. Possible values are 'normal', 'skewed' or 'truncated.normal'.
`skewness`	Numeric - Momentum of distribution skewness
`inferior_limit`	Numeric - Value of the lower bound for the truncated distribution, defaults to '-Inf'. Ignored if `distribution` is either 'normal' or 'skewed'.
`superior_limit`	Numeric - Value of the upper bound for the truncated distribution, defaults to 'Inf'. Ignored if `distribution` is either 'normal' or 'skewed'.
`balanced`	Logical - Whether the study will be performed with the same number of subjects in all groups. Default is `TRUE`. See 'details'.
`nsims`	Integer - Number of iterations.

Details

As internal function runs with a single iteration inside `graph_twoway_assumptions`, which in itself is inside `calculate_mean_matrix` to generate data for the cell mean and standard deviation plot.

For unbalanced independent measures designs, this function generates a simulation with `max(group_size)` for all factors combinations and then eliminates observations at random in those factor combinations that have less participants or study subjects. This may not sound like the most efficient way to proceed, is quite fast anyhow. The 'n' column in the output will reflect how many observations each factor combination has. This should match the input matrix.

Value

data.frame with modeled outcome values, factor level labels, iteration number and sample size.

Examples

```
refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85
factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

## Independent design
effects_treat_time <- calculate_mean_matrix(refmean = refmean,
                                            fAeffect = treateff, fBeffect = timeeff,
                                            nlfA = treatgroups, nlfB = timepoints,
                                            label_list = factors_levels_names)
## Inspect plot to check if matrices correspond to design
n <- 20
independent_experiment <- twoway_simulation_independent(group_size = n,
                                                        matrices_obj = effects_treat_time)
head(independent_experiment, 10)
```

`twoway_simulation_testing`

Calculate power for global main effects and interaction from two-way factorial simulated data

---

Description

This functions takes the output of either the `twoway_simulation_independent` or the `twoway_simulation_correlated` functions and calculates the power of the sample size used in the simulation under parametric analysis of variance, rank based analysis of variance or permutation testing.

Usage

```
twoway_simulation_testing(data, test = "ANOVA", alpha = 0.05)
```

Arguments

- data
  - Simulation obtained from the twoway_simulation_independent or twoway_simulation_correla
- test
  - The test to be applied. Possible values are "ANOVA" (default), "rank" and "permutation".
- alpha
  - Type I error rate. Default is 0.05.

Value

A data.frame with the power and 95% confidence interval for each of the main effects and their interaction.

Examples

```
## After creating a 'matrices_obj' with the 'calculate_mean_matrix' function.

refmean <- 1
treatgroups <- 4
timepoints <- 5
treateff <- 1.5
timeeff <- 0.85
rho <- 0.8
withinf <- "fB"
factors_levels_names <- list(treatment=letters[1:treatgroups], time=1:timepoints)

effects_treat_time <- calculate_mean_matrix(refmean = refmean,
                                            fAeffect = treateff, fBeffect = timeeff,
                                            nlfA = treatgroups,  nlfB = timepoints,
                                            rho = rho, withinf = withinf,
                                            label_list = factors_levels_names)

n <- 7
correlated_sim <- twoway_simulation_correlated(group_size=n, matrices_obj=effects_treat_time,
                  nsims=20)
##used smaller number of iterations to reduce computation time

twoway_simulation_testing(correlated_sim)
## defaults to parametric analysis of variance

twoway_simulation_testing(correlated_sim, test="rank")
## rank based analysis of variance

## permutation test is another option
```

Index