Package: Randomization (via r-universe)

November 23, 2026

Type Package

Title Randomization of Designs

Version 0.1

Date 2025-11-27

Maintainer Krämer Konrad <konrad.kraemer@kit.edu>

Description Randomization of Designs of Experimental Plans.

License GPL-3

Depends R (>= 4.1.0)

Imports pwr, Rcpp

LinkingTo Rcpp, RcppThread

Suggests tinytest

Encoding UTF-8

Repository https://konrad1991.r-universe.dev

Date/Publication 2026-11-19 14:09:53 UTC

RemoteUrl https://github.com/ComPlat/OpenStats

RemoteRef HEAD

RemoteSha 661bccc166a47416219bc5f51b37ba1ee91e084e

RemoteSubdir Randomization

Contents

calc_n_one_way_anova 2
random_assign 3
random_finite_assign 4

Index **6**

`calc_n_one_way_anova` *Calculates the sample size using power analysis*

Description

A power analysis (for a one way anova) is conducted for the defined predictors. Thereby, the number of repetions per predictor level is calculated.

Usage

```
randomization_one_way_anova(
  predictors,
  primary_factor,
  cohens_f,
  sig_level,
  desired_power
)
```

Arguments

`predictors`	A named list which contains all predictors.
`primary_factor`	a character vector defining the name of the predictor which is used in the power analysis.
`cohens_f`	cohens_f
`sig_level`	The significance level which should be used in the power analysis
`desired_power`	desired_power

Examples

```
predictors <- list(
    cellLines = c("HeLa", "Hek"),
    Treatment = LETTERS[1:4]
    )
res <- Randomization::calc_n_one_way_anova(
  predictors = predictors,
  primary_factor = "Treatment",
  cohens_f = 0.25,
  sig_level = 0.05,
  desired_power = 0.8
)
```

random_assign *Creates a Random Assignment to a Dataframe*

Description

The desired groups are assigned to df using different randomization methods.

Usage

```
random_assign(
  df,
  groups,
  ratios,
  col,
  block_col,
  strata_cols,
  randomization_method,
  n_quantiles,
  seed
)
```

Arguments

df	A data.frame to which the groups should be randomly aissgned
groups	A character vector containing the groups which should be assigned.
ratios	A numeric vector defining the ratios of the groups
col	A character defining the column(name) to which the groups should be assigned
block_col	In case the randomization_method is "block", or "balanced", you have to pass a character vector containing the column defining the blocks.
strata_cols	In case the randomization_method is "block_stratified", you have to pass a character vector containing the column(s) for which the assignment should be stratified
randomization_method	one of these options: "simple", "block", "block_stratified", or "balanced"
n_quantiles	A full number defining how fine granulated you want to split up finite groups
seed	A seed number to initialize the Random Number Generator

Examples

```
predictors <- list(
  cellLines = c("HeLa", "Hek"),
  Treatment = LETTERS[1:4]
)
design <- Randomization:::completely_randomised_design(predictors, 10)
```

```
groups <- paste0("Day", 1:4)
checks <- logical(6L)
res <- Randomization::random_assign(
  design,
  groups = groups,
  c(1, 1, 1, 1),
  "Group",
  randomization_method = "simple",
  seed = 1234
)
```

`random_finite_assign` *Random Assignment of a Finite Population to a Design via Optimization*

Description

Performs an optimization-based random assignment of a finite set of experimental units (given in `groups`) to a predefined experimental design (`design`).

The assignment is initialized randomly and iteratively improved using a greedy swap-based optimization procedure. In each iteration, potential swaps are evaluated according to the chosen loss function.

If `ids` are supplied, they are carried through the permutation and returned alongside the assignment.

Usage

```
random_finite_assign(
  seed,
  groups,
  design,
  max_iter = 50L,
  ridge = 1e-8,
  loss_function = "Default",
  verbose = TRUE,
  ids = NULL
)
```

Arguments

`seed`	A single numeric value used to initialize the random number generator.
`groups`	A `data.frame` containing the finite population to be assigned. The columns represent covariates used for balancing during optimization.
`design`	A `data.frame` defining the experimental design (e.g. treatment structure or block structure). Units from `groups` are assigned to the rows of this design.
`max_iter`	Integer. Maximum number of swap-iterations used in the greedy optimization procedure. Default is `50L`.

ridge	Numeric value used as ridge regularization (e.g. in Mahalanobis-based loss functions). Default is 1e-8.
loss_function	Character string specifying the loss function used for balancing. One of "Default" or "Mahalanobis".
verbose	Logical. If TRUE, prints progress information during optimization. Default is TRUE.
ids	Optional vector of unique identifiers for the units in groups. Must have length nrow(groups). If provided, the returned object contains these identifiers in permuted order.

Value

A list containing the optimized assignment, including:

- assigned: The design data frame with assigned units.
- perm: The final permutation (row indices of groups).
- unit_index: Row indices of groups in assigned order.
- unit_id: The corresponding identifiers (if ids was provided).
- blocks: The block structure used during optimization.
- UNUSED: Units not assigned (if the population size exceeds the number of required design rows).

Examples

```
# Define experimental design
design <- data.frame(
  treatment = rep(c("control", "one", "two"), each = 10L)
)

# Finite population with covariate and explicit ID
groups <- data.frame(
  weight = rnorm(30, mean = 300, sd = 25)
)

animal_ids <- paste0("Animal_", seq_len(nrow(groups)))

res <- random_finite_assign(
  seed = 42,
  groups = groups,
  design = design,
  ids = animal_ids,
  loss_function = "Default"
)

head(res$assigned)
head(res$unit_id)
```

Index

calc_n_one_way_anova, 2

random_assign, 3
random_finite_assign, 4