Package 'mlpwr'

May 8, 2026

Title A Power Analysis Toolbox to Find Cost-Efficient Study Designs

Version 1.1.1

Description We implement a surrogate modeling algorithm to guide simulation-based sample size planning. The method is described in detail in our paper (Zimmer & Debelak (2023) <doi:10.1037/met0000611>). It supports multiple study design parameters and optimization with respect to a cost function. It can find optimal designs that correspond to a desired statistical power or that fulfill a cost constraint. We also provide a tutorial paper (Zimmer et al. (2023) <doi:10.3758/s13428-023-02269-0>).

License GPL (>= 3)

LazyData true

URL https://github.com/flxzimmer/mlpwr

BugReports https://github.com/flxzimmer/mlpwr/issues

Imports utils, stats, DiceKriging, digest, ggplot2, randtoolbox, rlist, rgenoud

Suggests knitr, lme4, lmerTest, mirt, pwr, rmarkdown, simr, sn, tidyr, WeightSVM

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

Depends R (>= 3.5.0)

NeedsCompilation no

Author Felix Zimmer [aut, cre] (ORCID: <https://orcid.org/0000-0002-8127-0007>), Rudolf Debelak [aut] (ORCID: <https://orcid.org/0000-0001-8900-2106>), Marc Egli [ctb]

Maintainer Felix Zimmer <felix.zimmer@mail.de>

Repository CRAN

Date/Publication 2024-10-03 23:00:30 UTC

Contents

`example.simfun` *Example simulation functions*

Description

These simulation functions can be loaded as examples or templates. They correspond to the functions in the 'simulation_functions' vignette. The vignette contains the code and a description of the simfuns.

Usage

```
example.simfun(name)
```

Arguments

`name`	Name of the simulation function as a character. Currently available are: 'ttest','anova','glm2','irt1','irt2','r

Value

The returned object is the simulation function.

Examples

```
simfun = example.simfun('ttest')
simfun(400)
```

extensions_results *Results of the extensions vignette*

Description

Since the Monte Carlo simulations required to generate the results in the extensions vignette can be time-consuming, we have precomputed them and included them as a data file.

Usage

```
extensions_results
```

Format

A list with designresult objects created in the vignette "extensions"

find.design *Find optimal study designs*

Description

Perform a surrogate modeling approach to search for optimal study design parameters. For further guidance on how to use the package and the `find.design` function specifically, see the Readme.md file.

Usage

```
find.design(
  simfun,
  boundaries,
  power = NULL,
  evaluations = 4000,
  ci = NULL,
  ci_perc = 0.95,
  time = NULL,
  costfun = NULL,
  cost = NULL,
  surrogate = NULL,
  n.startsets = 4,
  init.perc = 0.2,
  setsize = NULL,
  continue = NULL,
  dat = NULL,
  silent = FALSE,
  autosave_dir = NULL,
```

```
  control = list(),
  goodvals = "high",
  aggregate_fun = mean,
  noise_fun = "bernoulli",
  integer = TRUE,
  use_noise = TRUE
)
```

Arguments

simfun	function to generate hypothesis test results with. Takes design parameters as input and outputs a logical (result of the hypothesis test). The function can take the designs through one argument as a vector or through multiple arguments. For example, function(x) where x is later used with x=c(n,k) for two design parameters n and k is valid. Also valid is a definition using function(n,k).
boundaries	list containing lower and upper bounds of the design space. The list should consist of named vectors, each containing the upper and lower bound for the respective design parameter dimensions. For one design parameter dimension, can also be a vector containing the upper and lower bounds.
power	numeric; desired statistical power
evaluations	integer; number of simfun evaluations to be performed before termination
ci	numeric; desired width of the confidence interval at the predicted value on termination.
ci_perc	numeric; specifying the desired confidence interval, e.g. 95% or 99%.
time	integer; seconds until termination
costfun	function that takes a vector of design parameters as input and outputs a cost, e.g. monetary costs. Necessary for simfuns with multiple input dimensions.
cost	numeric; cost threshold. Design parameter set with highest power is searched among sets that fulfill this cost threshold.
surrogate	character; which surrogate model should be used. The default is 'logreg' for one design parameter and 'gpr' for multiple design parameters. The current options are: 'gpr', 'svr', 'logreg', 'reg' for one-dimensional designs and 'gpr' and 'svr' for multi-dimensional designs.
n.startsets	integer; number of startsets used per dimension of simfun
init.perc	numeric; percentage of evaluations used for the initialization phase
setsize	The number of draws from the simfun in each iteration
continue	Object of class designresult as created by the find.design function. Will be used to continue the search, using all collected simulation results so far.
dat	list of data from a previous design result.
silent	logical; suppresses output during the search.
autosave_dir	character; file location for saving the dat object after each update. The dat object is saved in autosave_dir/dat_autosave.Rdata. It can be loaded for example using load(paste0(autosave_dir,"/dat_autosave.Rdata")).

control	list specifying arguments passed to the surrogate models. For example, list(covtype='gauss') can be used with the gpr surrogate to use a different covariance structure than the default.
goodvals	character indicating whether higher or lower criterion values are preferable given equal cost; the default is "high" for statistical power, the other option is "low".
aggregate_fun	function to aggregate results of the evaluations of the simulation function; the default is mean, as for statistical power.
noise_fun	function to calculate the noise or variance of the aggregated results of the Monte Carlo evaluations; can also be the character value "bernoulli" (default) to indicate the variance of the Bernoulli distribution used for statistical power. This function is $p(1-p)/n$, where p is the statistical power and n is the number of performed evaluations.
integer	logical indicating whether the design parameters are integers or not; the default is TRUE, which is suitable for sample size, for example.
use_noise	logical indicating whether noise variance should be used; the default is TRUE.

Value

function returns an object of class designresult

Examples

```
## T-test example:

# Load a simulation function
simfun <- example.simfun('ttest')
# Perform the search
ds <- find.design(simfun = simfun, boundaries = c(100,300), power = .95)
# Output the results
summary(ds)
# Plot results
plot(ds)

## Two-dimensional simulation function:

simfun <- example.simfun('anova')
# Perform the search
ds <- find.design(simfun = simfun,
 costfun = function(n,n.groups) 5*n+20*n.groups,
 boundaries = list(n = c(10, 150), n.groups = c(5, 30)),
 power = .95)
# Output the results
summary(ds)
# Plot results
plot(ds)


##  Mixed model example with a custom, two-dimensional simulation function:

library(lme4)
```

```
library(lmerTest)

# Simulation function
simfun_multilevel <- function(n.per.school,n.schools) {

  # generate data
  group = rep(1:n.schools,each=n.per.school)
  pred = factor(rep(c("old","new"),n.per.school*n.schools),levels=c("old","new"))
  dat = data.frame(group = group, pred = pred)

  params <- list(theta = c(.5,0,.5), beta = c(0,1),sigma = 1.5)
 names(params$theta) = c("group.(Intercept)","group.prednew.(Intercept)","group.prednew")
  names(params$beta) = c("(Intercept)","prednew")
 dat$y <- simulate.formula(~pred + (1 + pred | group), newdata = dat, newparams = params)[[1]]

  # test hypothesis
  mod <- lmer(y ~ pred + (1 + pred | group), data = dat)
  pvalue <- summary(mod)[["coefficients"]][2,"Pr(>|t|)"]
  pvalue < .01
}
# Cost function
costfun_multilevel <- function(n.per.school, n.schools) {
  100 * n.per.school + 200 * n.schools
}
# Perform the search, can take a few minutes to run
ds <- find.design(simfun = simfun_multilevel, costfun = costfun_multilevel,
boundaries = list(n.per.school = c(5, 25), n.schools = c(10, 30)), power = .95,
evaluations = 1000)
# Output the results
summary(ds)
# Plot results
plot(ds)
```

`plot.designresult` *Plot search result*

Description

Plot a one- or two-dimensional graph of the result.

Usage

```
## S3 method for class 'designresult'
plot(
  x,
  design = NULL,
  adderrorbars = NULL,
  addribbon = NULL,
```

```
  trim = TRUE,
  type = "heat",
  color.width = 0.15,
  color.gradient = "diverging",
  ...
)
```

Arguments

x	Object of class designresult as created by the find.design function.
design	Specify a design as a list. Can be used to make a 1D plot for a two-dimensional simfun. Set NA for the dimension that should be plotted and set a value for all others. For example: design=list(n=NA,k=9)
adderrorbars	logical. Plots error bars in the 1D plot if TRUE. Default is FALSE (also if specified as NULL).
addribbon	logical. Adds ribbon in the 1D plot if TRUE. Default is TRUE (also if specified as NULL).
trim	logical. Option to trim the plotting area for the 2D line plot. The trimmed area is the area where the line is plotted. Default is TRUE.
type	character indicating the type of the 2D plot. Can be 'heat'(default) or 'line'.
color.width	numeric. Option for the diverging color map in the 2D plot. Width of the blue-white color band.
color.gradient	character indicating whether the 2D plot should have a "diverging" color gradient (white-blue-white, default) or a "linear" color gradient (blue-red)
...	additional arguments to be passed.

Value

A ggplot object

Examples

```
#Load a simulation function
simfun = example.simfun('ttest')
# Perform the search
ds = find.design(simfun = simfun, boundaries = c(100,300), power = .95)
# Plot results
plot(ds)
```

print.summary.designresult *Print Summary of the search result*

Description

Print Summary of the search result

Usage

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

Arguments

x	Object of class designresult as created by the find.design function
...	additional arguments to be passed

Value

An object of class summary.designresult

Examples

```
#Load a simulation function
simfun = example.simfun('ttest')
# Perform the search
ds = find.design(simfun = simfun, boundaries = c(100,300), power = .95)
# Output the results
summary(ds)
```

simulations_data *Display info on the generated data*

Description

Output a data frame that, for each set of design parameters, includes the cost, the estimated power and SE, the surrogate model estimated power and SE, and the number of performed evaluations.

Usage

```
simulations_data(ds)
```

Arguments

ds	Object of class designresult as created by the find.design function

Details

The raw power is estimated using the ratio of significant hypothesis tests among the performed evaluations. The raw SE is estimated using $p(1-p)/n$ with p being the raw power and n being the number of performed evaluations.

For the estimation of SE using a surrogate model, GPR is employed because it is capable of variance estimation, unlike the other surrogate models.

Value

A data frame

Examples

```
# Load a simulation function
simfun <- example.simfun('ttest')
# Perform the search
ds <- find.design(simfun = simfun, boundaries = c(100,300), power = .95)
# Display more info on the collected data
simulations_data(ds)
```

---

summary.designresult *Summary of the search result*

Description

Produce summary statistics of the results and the algorithm run.

Usage

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

Arguments

object	Object of class designresult as created by the find.design function
...	additional arguments to be passed

Value

An object of class summary.designresult

Examples

```
#Load a simulation function
simfun = example.simfun('ttest')
# Perform the search
ds = find.design(simfun = simfun, boundaries = c(100,300), power = .95)
# Output the results
summary(ds)
```

Index