Package 'PUMP'

March 12, 2025

Type Package

Title Power Under Multiplicity Project

Version 1.0.4

Description
Estimates power, minimum detectable effect size (MDES) and sample size requirements. The context is multilevel randomized experiments with multiple outcomes. The estimation takes into account the use of multiple testing procedures. Development of this package was supported by a grant from the Institute of Education Sciences (R305D170030). For a full package description, including a detailed technical appendix, see <doi:10.18637/jss.v108.i06>.

URL https://github.com/MDRCNY/PUMP, https://mdrcny.github.io/PUMP/

BugReports https://github.com/MDRCNY/PUMP/issues

Depends R (>= 3.5.0)

Imports dplyr, ggthemes, ggplot2, ggpubr, glue, future, lme4, magrittr, methods, mvtnorm, parallel, purrr, randomizr, readr, rlang, stats, stringr, tibble, tidyr, tidyselect

Suggests furrr, here, kableExtra, knitr, PowerUpR (>= 1.1.0), testthat

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Author Luke Miratrix [aut, cre] (<https://orcid.org/0000-0002-0078-1906>), Kristen Hunter [aut] (<https://orcid.org/0000-0002-5678-4620>), Zarni Htet [aut], Kristin Porter [aut], MDRC [cph], Institute of Education Sciences [fnd]

Maintainer Luke Miratrix <luke_miratrix@gse.harvard.edu>

Repository CRAN

Date/Publication 2025-03-12 12:00:02 UTC

Contents

calc_df . . . 2
check_cor . . . 3
convert_params . . . 5
gen_base_sim_data . . . 5
gen_cluster_ids . . . 6
gen_corr_matrix . . . 7
gen_sim_data . . . 7
gen_T.x . . . 8
gen_Yobs . . . 9
get_power_results . . . 10
parse_d_m . . . 10
plot.pumpgridresult . . . 11
plot.pumpresult . . . 12
power_curve . . . 14
print_context . . . 15
pumpgridresult . . . 15
pumpresult . . . 16
pump_info . . . 19
pump_mdes . . . 19
pump_mdes_grid . . . 22
pump_power . . . 25
pump_power_grid . . . 28
pump_sample . . . 30
pump_sample_grid . . . 33
transpose_power_table . . . 36
update.pumpgridresult . . . 37
update.pumpresult . . . 37
update_grid . . . 38

Index **39**

`calc_df` *Calculate degrees of freedom (support function)*

Description

Given sample sizes, return the used degrees of freedom (frequently conservative) for the design and model.

Usage

```
calc_df(d_m, J, K, nbar, numCovar.1, numCovar.2, numCovar.3, validate = TRUE)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
validate	logical; whether or not to validate if output df is <= 0.

Value

scalar; degrees of freedom for the context.

---

check_cor *Check correlation of test statistics (simulation function)*

Description

Estimates the pairwise correlations between test statistics for all outcomes.

Takes in two options: - a pumpresult object OR - a list of necessary data-generating parameters - the context (d_m) - Tbar

Note that this function can take several minutes to run.

Usage

```
check_cor(
  pump.object = NULL,
  rho.V = NULL,
  rho.w0 = NULL,
  rho.w1 = NULL,
  rho.X = NULL,
  rho.u0 = NULL,
  rho.u1 = NULL,
  rho.C = NULL,
  rho.r = NULL,
  d_m = NULL,
```

```
  model.params.list = NULL,
  Tbar = 0.5,
  n.sims = 100
)
```

Arguments

pump.object	A pumpresult object.
rho.V	matrix; correlation matrix of level 3 covariates.
rho.w0	matrix; correlation matrix of level 3 random effects.
rho.w1	matrix; correlation matrix of level 3 random impacts.
rho.X	matrix; correlation matrix of level 2 covariates.
rho.u0	matrix; correlation matrix of level 2 random effects.
rho.u1	matrix; correlation matrix of level 2 random impacts.
rho.C	matrix; correlation matrix of level 1 covariates.
rho.r	matrix; correlation matrix of level 1 residuals.
d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
model.params.list	list; model parameters such as ICC, R2, etc. See simulation vignette for details.
Tbar	scalar; the proportion of samples that are assigned to the treatment.
n.sims	numeric; Number of simulated datasets to generate. More datasets will achieve a more accurate result but also increase computation time.

Value

matrix; M x M correlation matrix between test statistics.

Examples

```
pp <- pump_power( d_m = "d3.2_m3ff2rc",
                  MTP = "BF",
                  MDES = rep( 0.10, 2 ),
                  M = 2,
                  J = 4, # number of schools/block
                  K = 10, # number RA blocks
                  nbar = 50,
                  Tbar = 0.50, # prop Tx
                  alpha = 0.05, # significance level
                  numCovar.1 = 5, numCovar.2 = 3,
                  R2.1 = 0.1, R2.2 = 0.7,
                  ICC.2 = 0.05, ICC.3 = 0.4,
                  rho = 0.4, # how correlated test statistics are
                  tnum = 200
)
cor.tstat <- check_cor(
    pump.object = pp, n.sims = 4
)
est.cor <- mean(cor.tstat[lower.tri(cor.tstat)])
```

convert_params *Converts model params into DGP params (simulation function)*

Description

Converts user-provided parameters such as ICC and omega into data-generating parameters for the multilevel random effects model used to produce simulated data, such as variance values and covariate coefficients.

This function is beyond the main scope of calculating power, and is instead used for simulating data. For more info on use, see the simulation vignette.

Usage

```
convert_params(param.list)
```

Arguments

param.list	list; model parameters such as ICC, R2, etc.

Value

list; data-generating parameters.

gen_base_sim_data *Generate base simulated multi-level data (simulation function)*

Description

Generates simulated data for multi-level RCTs for pump-supported designs and models for both unobserved potential outcomes. This function does not generate treatment assignments or observed outcomes–see gen_sim_data() for that.

This method takes in a list of necessary data-generating parameters, following the rest of the package.

This function is beyond the main scope of calculating power, and is instead used for simulating data. For more info on use, see the simulation vignette.

Usage

```
gen_base_sim_data(
  param.list,
  pump.object = NULL,
  return.as.dataframe = TRUE,
  no.list = TRUE,
  dgp.params = FALSE
)
```

Arguments

param.list	list; model parameters such as ICC, R2, etc. See simulation vignette for details.
pump.object	A pumpresult object.
return.as.dataframe	TRUE means return list of dataframes, one for each outcome. FALSE means return components of the covariates, etc., in a list.
no.list	Only relevant if return.as.dataframe=TRUE. no.list=TRUE means if M=1 return the dataframe, not a list of length 1. FALSE means return a list of length 1, even if there is only 1 outcome.
dgp.params	TRUE means param.list is already converted to DGP parameters, FALSE means it needs to be converted via 'convert_params()'.

Value

list; potential outcomes given control y0, treatment y1, covariates V.k, X.jk, C.ijk, or list of dataframes if return.as.dataframe = TRUE.

See Also

gen_sim_data

---

gen_cluster_ids *Generates school and district assignments (simulation function)*

Description

Generates simple default schools and districts IDs for individual students for the purpose of simulations. This assumes equal sized schools in equal sized districts.

This function is beyond the main scope of calculating power, and is instead used for simulating data. For more info on use, see the simulation vignette.

Usage

```
gen_cluster_ids(nbar, J, K)
```

Arguments

nbar	scalar; number of individuals per school.
J	scalar; number of schools per district.
K	scalar; number of districts.

Value

list; school and district assignments (S.id, D.id) for each individual.

gen_corr_matrix *Generate correlation matrix (simulation function)*

Description

Generate correlation matrix (simulation function)

Usage

```
gen_corr_matrix(M, rho.scalar)
```

Arguments

M	scalar; dimension of matrix.
rho.scalar	scalar; rho value.

Value

matrix; M x M correlation matrix with rho.scalar as diagonal.

gen_sim_data *Generate simulated multi-level data (simulation function)*

Description

Generates simulated data for multi-level RCTs for pump-supported designs and models for both unobserved and observed potential outcomes.

Takes in two options:

- a pumpresult object OR

- a list of necessary data-generating parameters - the context (d_m) - Tbar (proportion assigned to treatment)

This function is beyond the main scope of calculating power, and is instead used for simulating data. For more info on use, see the simulation vignette.

Usage

```
gen_sim_data(
  d_m = NULL,
  param.list = NULL,
  Tbar = 0.5,
  pump.object = NULL,
  return.as.dataframe = TRUE,
  no.list = TRUE,
  include_POs = FALSE
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
param.list	list; model parameters such as ICC, R2, etc. See simulation vignette for details.
Tbar	scalar; the proportion of samples that are assigned to the treatment.
pump.object	A pumpresult object.
return.as.dataframe	TRUE means return list of dataframes, one for each outcome. FALSE means return components of the covariates, etc., in a list.
no.list	Only relevant if return.as.dataframe=TRUE. no.list=TRUE means if M=1 return the dataframe, not a list of length 1. FALSE means return a list of length 1, even if there is only 1 outcome.
include_POs	Include columns for the potential outcomes in addition to the observed outcome.

Value

list; potential outcomes, covariates, observed outcomes, and treatment assignment.

Examples

```
pp <- pump_power( d_m = "d3.2_m3ff2rc",
                  MTP = "BF",
                  MDES = rep( 0.10, 3 ),
                  M = 3,
                  J = 3, # number of schools/block
                  K = 21, # number RA blocks
                  nbar = 258,
                  Tbar = 0.50, # prop Tx
                  alpha = 0.05, # significance level
                  numCovar.1 = 5, numCovar.2 = 3,
                  R2.1 = 0.1, R2.2 = 0.7,
                  ICC.2 = 0.05, ICC.3 = 0.4,
                  rho = 0.4,
                  tnum = 200
)
sim.data <- gen_sim_data(pump.object = pp)
```

---

gen_T.x *Generate treatment assignment vector (simulation function)*

Description

Given a RCT design and supporting information, generates treatment assignments for each student.

This function is beyond the main scope of calculating power, and is instead used for simulating data. For more info on use, see the simulation vignette.

Usage

```
gen_T.x(d_m, S.id, D.id, Tbar)
```

Arguments

d_m	string; design and model.
S.id	vector; school assignments.
D.id	vector; district assignments.
Tbar	scalar; probability of treatment assignment.

Value

vector; treatment assignments for each unit.

---

gen_Yobs *Generate observed outcomes (simulation function)*

Description

Takes in a full dataset of both observed and latent potential outcomes and the treatment assignment vector, and returns only the observed outcomes.

This function is beyond the main scope of calculating power, and is instead used for simulating data. For more info on use, see the simulation vignette.

Usage

```
gen_Yobs(full.data, T.x)
```

Arguments

full.data	data.frame; full dataset of potential outcomes.
T.x	vector; binary assignment to treat/control.

Value

vector; observed outcomes

get_power_results *Calculates different definitions of power (support function)*

Description

This function takes in a matrix of adjusted p-values and unadjusted p-values and outputs different types of power.

This function is mostly for internal use, but may be of interest to users who wish to calculate power on their own.

Usage

```
get_power_results(
  adj.pval.mat,
  unadj.pval.mat,
  ind.nonzero,
  alpha,
  drop.zero.outcomes = TRUE,
  adj = TRUE
)
```

Arguments

adj.pval.mat	matrix; adjusted p-values, columns are outcomes
unadj.pval.mat	matrix; unadjusted p-values, columns are outcomes
ind.nonzero	vector; which outcomes correspond to nonzero effects.
alpha	scalar; the family wise error rate (FWER).
drop.zero.outcomes	logical; whether to report power results for outcomes with MDES = 0.
adj	logical; whether p-values are unadjusted or not.

Value

data frame; power results for individual, minimum, complete power.

parse_d_m *Return characteristics of a given context/d_m code (support function)*

Description

Returns number of levels and model at each level. See pump_info()$Context to get a list of supported d_ms.

Usage

```
parse_d_m(d_m, d_only = FALSE)
```

Arguments

d_m	string; context to parse.
d_only	TRUE/FALSE; TRUE means only look at design, ignore model if present.

Value

list; list of features including number of levels, level of randomization, etc.

Examples

```
supported <- pump_info(comment = FALSE)$Context
parse_d_m( supported$d_m[4] )
```

---

plot.pumpgridresult *Plot a pumpgridresult object (result function)*

Description

Plots grid results across values of a single parameter, specified by the user using var.vary, for a single definition of power, specified by power.definition.

If multiple things vary in the grid, the outcome (power, mdes, or sample size) will be averaged (marginalized) across the other varying factors. This treats the grid as a multifactor simulation, with this showing the "main effect" of the specified parameter.

Usage

```
## S3 method for class 'pumpgridresult'
plot(
  x,
  power.definition = NULL,
  var.vary = NULL,
  color = "MTP",
  lines = TRUE,
  include.title = FALSE,
  ...
)
```

Arguments

x	pumpgridresult object.
power.definition	string; definition of power to plot. If NULL, plot all definitions as a facet wrap.
var.vary	string; variable to vary on X axis. If NULL, and only one thing varies, then it will default to single varying parameter.
color	string; Group lines by this element to make an interaction plot (default "MTP", giving one curve for each MTP).
lines	logical; TRUE means connect dots with lines on the plots. FALSE means no lines.
include.title	logical; whether to include/exclude title (if planning a facet wrap, for example).
...	additional parameters.

Value

plot; a ggplot object of outcome across parameter values.

Examples

```
g <- pump_power_grid( d_m = "d3.2_m3ff2rc", MTP = c( "HO", "BF" ),
 MDES = 0.10, J = seq(5, 10, 1), M = 5, K = 7, nbar = 58,
 Tbar = 0.50, alpha = 0.15, numCovar.1 = 1,
 numCovar.2 = 1, R2.1 = 0.1, R2.2 = 0.7,
 ICC.2 = 0.25, ICC.3 = 0.25, rho = 0.4, tnum = 200)
plot(g, power.definition = 'min1')
```

---

plot.pumpresult *Plot a pumpresult object (result function)*

Description

For the object returned by pump_power(), visualizes different definitions of power across MTPs. For the object returned by pump_mdes() or pump_sample(), plot a power curve as a function of MDES or sample size, respectively. This latter call will calculate power over a passed range from low to high to generate this curve.

Several of the passed parameters only apply to the mdes or sample versions, and are for controlling the grid search and plot.

For pump_power, will include standard errors of uncertainty on calculated power. These depend on number of iterations (tnum) used in the simulation.

Usage

```
## S3 method for class 'pumpresult'
plot(
  x,
  type = "power",
  all = TRUE,
  low = NULL,
  high = NULL,
  grid.size = 5,
  breaks = grid.size,
  include_SE = TRUE,
  ...
)
```

Arguments

x	pumpresult object.
type	string; "power" or "search". Specifies whether to plot the default power graph, or the search path. The search path is only valid for MDES and SS results.
all	Logical. If TRUE, merge in the search path from the original search to the estimated power curve, for MDES or sample plots.
low	Low range of x-axis and curve calculation for sample or MDES plots. (Optional.)
high	High range of x-axis and curve calculation. (Optional.)
grid.size	If calculating curve for sample or MDES plot, how many grid points?
breaks	If plotting a curve for sample or MDES, where to put the grid points?
include_SE	Include (approximate) SEs on the power estimates, if they are naturally calculated.
...	additional parameters, such as, in case of sample or mdes objects, tnum for setting number of replicates or all (logical) for determining whether to include original points in the estimated curve, or include.points (logical) for including points on the plot itself.

Value

plot; a ggplot object of power across different definitions.

Examples

```
pp1 <- pump_power(d_m = "d2.2_m2rc", MTP = 'HO',
 nbar = 50, J = 20, M = 8, numZero = 5,
 MDES = 0.30, Tbar = 0.5, alpha = 0.05, two.tailed = FALSE,
 numCovar.1 = 1, numCovar.2 = 1, R2.1 = 0.1, R2.2 = 0.7,
 ICC.2 = 0.05, rho = 0.2, tnum = 200)

plot(pp1)
```

```
J <- pump_sample(d_m = "d2.1_m2fc",
   MTP = 'HO', power.definition = 'D1indiv',
   typesample = 'J', target.power = 0.6,
   nbar = 50, M = 3, MDES = 0.125,
   Tbar = 0.5, alpha = 0.05,
   numCovar.1 = 1, R2.1 = 0.1, ICC.2 = 0.05,
   rho = 0.2, tnum = 500)
plot(J)
plot(J, type = "search")
```

---

power_curve *Obtain a power curve for a range of sample size or MDES values*

Description

This is used to see how power changes as a function of sample size or MDES. It takes a fit pumpresult and calculates a power curve based on that scenario coupled with a passed range of values to make the curve over.

Usage

```
power_curve(
  x,
  all = FALSE,
  low = NULL,
  high = NULL,
  grid.size = 5,
  tnum = 2000
)
```

Arguments

x	a pumpresult object.
all	logical; if TRUE, merge in the search path from the original search.
low	scalar; low range for curve.
high	scalar; high range for the curve.
grid.size	scalar; number of points to calculate power for.
tnum	scalar; number of iterations to calculate power at each grid point.

Value

data.frame of power results.

print_context *Print context (design, model, parameter values) of pumpresult or pumpgridresult (result function)*

Description

Print out the context (design and model, with parameter values) of given pump result or pump grid result object. The "***" denotes varying values in the printout.

Usage

```
print_context(
  x,
  insert_results = FALSE,
  insert_control = FALSE,
  include_SE = TRUE,
  ...
)
```

Arguments

x	A pumpresult object or pumpgridresult object.
insert_results	Include actual results in the printout.
insert_control	Include the optimizer control parameter information.
include_SE	Include standard errors in the printout.
...	Extra arguments to pass to print.pumpresult.

Value

No return value; prints results.

pumpgridresult *Result object for results of grid power calculations*

Description

The pumpgridresult object is an S3 class that holds the results from ‘pump_power_grid()‘, ‘pump_sample_grid()‘, and ‘pump_mdes_grid()‘.

It has several methods that pull different information from this object, and some printing methods for getting nicely formatted results.

Usage

```
is.pumpgridresult(x)

## S3 method for class 'pumpgridresult'
print(x, header = TRUE, include_SE = FALSE, ...)

## S3 method for class 'pumpgridresult'
summary(object, include_SE = FALSE, ...)
```

Arguments

x	a pumpgridresult object (except for is.pumpgridresult, where it is a generic object to check).
header	logical; FALSE means skip some header info on the result, just print the data.frame of actual results.
include_SE	logical; TRUE means include standard errors and df.
...	extra options passed to print.pumpgridresult
object	object to summarize.

Value

is.pumpgridresult: TRUE if object is a pumpgridresult object.

print: No return value; prints results.

summary: No return value; prints results.

---

pumpresult *pumpresult object for results of power calculations*

Description

The pumpresult object is an S3 class that holds the results from ‘pump_power()‘, ‘pump_sample()‘, and ‘pump_mdes()‘.

It has several methods that pull different information from this object, and some printing methods for getting nicely formatted results.

Pump result objects are also data.frames, so they can be easily manipulated and combined. The return values from the ‘grid‘ functions will just return data frames in general.

Returns whether call was power, mdes, or sample.

Calls the print_context method with results and control both set to TRUE.

Usage

```
params(x, ...)

d_m(x, ...)

design(x, ...)

search_path(x, ...)

pump_type(x)

is.pumpresult(x)

## S3 method for class 'pumpresult'
x[...]

## S3 method for class 'pumpresult'
x[[...]]

## S3 method for class 'pumpresult'
dim(x, ...)

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

## S3 method for class 'pumpresult'
print(x, n = 10, header = TRUE, search = FALSE, include_SE = TRUE, ...)

## S3 method for class 'pumpresult'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	a pumpresult object (except for is.pumpresult, where it is a generic object to check).
...	additional arguments to be passed to the as.data.frame.list methods.
object	Object to summarize.
n	Number of lines of search path to print, max.
header	FALSE means skip some header info on the result, just print the data.frame of actual results.
search	FALSE means don't print the search path for a result for mdes or sample.
include_SE	TRUE means include standard errors given design (if any) in the printout. Default to TRUE.
row.names	NULL or a character vector giving the row names for the data frame.
optional	logical. If TRUE, setting row names and converting column names is optional.

Value

params: List of design parameters used.

d_m: Context (d_m) used (as string).

design (the randomization and levels) as string.

search_path: Dataframe describing search path, if it was saved in the pumpresult object.

pump_type: power, mdes, or sample, as a string.

is.pumpresult: TRUE if object is a pumpresult object.

‘[‘: pull out rows and columns of the dataframe.

‘[[‘: pull out single element of dataframe.

dim: Dimension of pumpresult (as matrix)

summary: No return value; prints results.

print: No return value; prints results.

as.data.frame: pumpresult object as a clean dataframe (no more attributes from pumpresult).

See Also

update

update_grid

print_context

print_context

Examples

```
pp <- pump_power(d_m = "d3.2_m3ff2rc",
  MTP = 'HO', nbar = 50, J = 30, K = 10,
  M = 5, MDES = 0.125, Tbar = 0.5, alpha = 0.05,
  numCovar.1 = 1, numCovar.2 = 1,
  R2.1 = 0.1, R2.2 = 0.1, ICC.2 = 0.2, ICC.3 = 0.2,
  omega.2 = 0, omega.3 = 0.1, rho = 0.5, tnum = 1000)

print(pp)
params(pp)
print_context(pp)
d_m(pp)
pump_type(pp)
is.pumpresult(pp)
as.data.frame(pp)
dim(pp)
summary(pp)
transpose_power_table(pp)

J <- pump_sample(d_m = "d2.1_m2fc",
  MTP = 'HO', power.definition = 'D1indiv',
  typesample = 'J', target.power = 0.7,
  nbar = 50, M = 3, MDES = 0.125,
  Tbar = 0.5, alpha = 0.05, numCovar.1 = 1,
```

```
  R2.1 = 0.1, ICC.2 = 0.05, rho = 0.2, tnum = 1000)

search_path(J)
power_curve(J)
```

---

pump_info *Provides details about supported package features (core function)*

Description

List user options: designs and models (d_m), including what parameters are relevant for each context; multiple testing procedures; types of power; design and model parameters.

Usage

```
pump_info(
  topic = c("all", "context", "adjustment", "power", "parameters"),
  comment = TRUE
)
```

Arguments

topic	string; what kind of info. One of: all, context, adjustment, power, parameters.
comment	logical; prints out long description of each design and method.

Value

list; a list of data frames with information about each topic.

See Also

For more detailed information about user choices, see the manuscript <doi:10.18637/jss.v108.i06>, which includes a detailed Technical Appendix including information about the designs and models and parameters.

---

pump_mdes *Estimate the minimum detectable effect size (MDES) (core function)*

Description

The user chooses the context (d_m), MTP, power definition, and choices of all relevant design parameters.

The functions performs a search algorithm, and returns the MDES value within the specified tolerance. For a list of choices for specific parameters, see pump_info().

Usage

```
pump_mdes(
  d_m,
  MTP = NULL,
  numZero = NULL,
  propZero = NULL,
  M = 1,
  nbar,
  J = 1,
  K = 1,
  Tbar,
  alpha = 0.05,
  two.tailed = TRUE,
  target.power = 0.8,
  power.definition,
  tol = 0.02,
  numCovar.1 = 0,
  numCovar.2 = 0,
  numCovar.3 = 0,
  R2.1 = 0,
  R2.2 = 0,
  R2.3 = 0,
  ICC.2 = 0,
  ICC.3 = 0,
  omega.2 = 0,
  omega.3 = 0,
  rho = NULL,
  rho.matrix = NULL,
  B = 1000,
  max.steps = 20,
  tnum = 1000,
  start.tnum = round(tnum/10),
  final.tnum = 4 * tnum,
  parallel.WY.cores = 1,
  updateProgress = NULL,
  give.optimizer.warnings = FALSE,
  verbose = FALSE
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
MTP	string, or vector of strings; multiple testing procedure(s). See pump_info() for list of choices.
numZero	scalar; additional number of outcomes assumed to be zero. Please provide NumZero + length(MDES) = M, if length(MDES) is not 1.

propZero	scalar; proportion of outcomes assumed to be zero (alternative specification to numZero). length(MDES) should be 1 or equal to (1-propZero)*M.
M	scalar; the number of hypothesis tests (outcomes), including zero outcomes.
nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
Tbar	scalar; the proportion of samples that are assigned to the treatment.
alpha	scalar; the family wise error rate (FWER).
two.tailed	scalar; TRUE/FALSE for two-tailed or one-tailed power calculation.
target.power	target power for search algorithm.
power.definition	see pump_info() for possible power definitions.
tol	tolerance for target power, defaults to 0.01 (1 This parameter controls when the search is done: when estimated power (checked with 'final.tnum' iterations) is within 'tol', the search stops.
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
R2.1	scalar, or vector of length M; percent of variation explained by level 1 covariates for each outcome.
R2.2	scalar, or vector of length M; percent of variation explained by level 2 covariates for each outcome.
R2.3	scalar, or vector of length M; percent of variation explained by level 3 covariates for each outcome.
ICC.2	scalar, or vector of length M; level 2 (school) intraclass correlation.
ICC.3	scalar, or vector length M; level 3 (district) intraclass correlation.
omega.2	scalar, or vector of length M; ratio of variance of level 2 average impacts to variance of level 2 random intercepts.
omega.3	scalar, or vector of length M; ratio of variance of level 3 average impacts to variance of level 3 random intercepts.
rho	scalar; assumed correlation between all pairs of test statistics.
rho.matrix	matrix; alternate specification allowing a full matrix of correlations between test statistics. Must specify either rho or rho.matrix, but not both.
B	scalar; the number of permutations for Westfall-Young procedures.
max.steps	how many steps allowed before terminating.

tnum	max number of samples for first iteration of search algorithm.
start.tnum	number of samples to start search (this will increase with each step).
final.tnum	number of samples for final draw.
parallel.WY.cores	number of cores to use for parallel processing of WY-SD.
updateProgress	function to update progress bar (only used for PUMP shiny app).
give.optimizer.warnings	whether to return verbose optimizer warnings.
verbose	TRUE/FALSE; Print out diagnostics of time, etc.

Value

a pumpresult object containing MDES results.

See Also

For more detailed information about this function and the user choices, see the manuscript <doi:10.18637/jss.v108.i06>, which includes a detailed Technical Appendix including information about the designs and models and parameters.

Examples

```
mdes <-  pump_mdes(
  d_m = "d3.1_m3rr2rr",
  MTP = 'HO',
  power.definition = 'D1indiv',
  target.power = 0.6,
  J = 30,
  K = 15,
  nbar = 50,
  M = 3,
  Tbar = 0.5, alpha = 0.05,
  two.tailed = FALSE,
  numCovar.1 = 1, numCovar.2 = 1,
  R2.1 = 0.1, R2.2 = 0.1,
  ICC.2 = 0.2, ICC.3 = 0.2,
  omega.2 = 0.1, omega.3 = 0.1,
  rho = 0.5, tnum = 2000)
```

---

pump_mdes_grid *Run pump_mdes on varying values of parameters (grid function)*

Description

See pump_power_grid() for more details.

Usage

```
pump_mdes_grid(
  d_m,
  MTP = NULL,
  M = 1,
  target.power = 0.8,
  power.definition = NULL,
  tol = 0.01,
  propZero = NULL,
  numZero = NULL,
  nbar,
  J = 1,
  K = 1,
  Tbar = 0.5,
  alpha = 0.05,
  numCovar.1 = NULL,
  numCovar.2 = NULL,
  numCovar.3 = NULL,
  R2.1 = NULL,
  R2.2 = NULL,
  R2.3 = NULL,
  ICC.2 = NULL,
  ICC.3 = NULL,
  omega.2 = NULL,
  omega.3 = NULL,
  rho = NULL,
  verbose = FALSE,
  drop.unique.columns = TRUE,
  ...
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
MTP	string, or vector of strings; multiple testing procedure(s). See pump_info() for list of choices.
M	scalar; the number of hypothesis tests (outcomes), including zero outcomes.
target.power	target power for search algorithm.
power.definition	see pump_info() for possible power definitions.
tol	tolerance for target power, defaults to 0.01 (1 This parameter controls when the search is done: when estimated power (checked with ‘final.tnum‘ iterations) is within ‘tol‘, the search stops.
propZero	scalar; proportion of outcomes assumed to be zero (alternative specification to numZero). length(MDES) should be 1 or equal to (1-propZero)*M.

numZero	scalar; additional number of outcomes assumed to be zero. Please provide NumZero + length(MDES) = M, if length(MDES) is not 1.
nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
Tbar	scalar; the proportion of samples that are assigned to the treatment.
alpha	scalar; the family wise error rate (FWER).
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
R2.1	scalar, or vector of length M; percent of variation explained by level 1 covariates for each outcome.
R2.2	scalar, or vector of length M; percent of variation explained by level 2 covariates for each outcome.
R2.3	scalar, or vector of length M; percent of variation explained by level 3 covariates for each outcome.
ICC.2	scalar, or vector of length M; level 2 (school) intraclass correlation.
ICC.3	scalar, or vector length M; level 3 (district) intraclass correlation.
omega.2	scalar, or vector of length M; ratio of variance of level 2 average impacts to variance of level 2 random intercepts.
omega.3	scalar, or vector of length M; ratio of variance of level 3 average impacts to variance of level 3 random intercepts.
rho	scalar; assumed correlation between all pairs of test statistics.
verbose	TRUE/FALSE; Print out diagnostics of time, etc.
drop.unique.columns	logical; drop all parameter columns that did not vary across the grid.
...	extra arguments passed to the underlying pump_power, pump_sample, or pump_mdes functions.

Value

a pumpgridresult object containing MDES results.

See Also

Other grid functions: pump_power_grid(), pump_sample_grid()

Examples

```
g <- pump_mdes_grid(d_m = "d3.2_m3ff2rc", MTP = "HO",
  target.power = c( 0.50, 0.80 ), power.definition = "D1indiv",
  tol = 0.05, M = 5, J = c( 3, 9 ), K = 7, nbar = 58,
  Tbar = 0.50, alpha = 0.15, numCovar.1 = 1, numCovar.2 = 1,
  R2.1 = 0.1, R2.2 = 0.7, ICC.2 = 0.05, ICC.3 = 0.9,
  rho = 0.4, tnum = 200)
```

---

pump_power *Estimate power across definitions (core function)*

Description

The user chooses the context (d_m), MTP, MDES, and choices of all relevant design parameters.

The functions returns power for all definitions of power for any MTP. For a list of choices for specific parameters, see pump_info().

Usage

```
pump_power(
  d_m,
  MTP = NULL,
  MDES,
  numZero = NULL,
  propZero = NULL,
  M = 1,
  nbar,
  J = 1,
  K = 1,
  Tbar,
  alpha = 0.05,
  two.tailed = TRUE,
  numCovar.1 = 0,
  numCovar.2 = 0,
  numCovar.3 = 0,
  R2.1 = 0,
  R2.2 = 0,
  R2.3 = 0,
  ICC.2 = 0,
  ICC.3 = 0,
  omega.2 = 0,
  omega.3 = 0,
  rho = NULL,
  rho.matrix = NULL,
  tnum = 10000,
  B = 1000,
  parallel.WY.cores = 1,
```

```
  drop.zero.outcomes = TRUE,
  updateProgress = NULL,
  validate.inputs = TRUE,
  long.table = FALSE,
  verbose = FALSE,
  exact.where.possible = TRUE
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
MTP	string, or vector of strings; multiple testing procedure(s). See pump_info() for list of choices.
MDES	scalar or vector; the desired MDES values for each outcome. Please provide a scalar, a vector of length M, or vector of values for non-zero outcomes.
numZero	scalar; additional number of outcomes assumed to be zero. Please provide NumZero + length(MDES) = M, if length(MDES) is not 1.
propZero	scalar; proportion of outcomes assumed to be zero (alternative specification to numZero). length(MDES) should be 1 or equal to (1-propZero)*M.
M	scalar; the number of hypothesis tests (outcomes), including zero outcomes.
nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
Tbar	scalar; the proportion of samples that are assigned to the treatment.
alpha	scalar; the family wise error rate (FWER).
two.tailed	scalar; TRUE/FALSE for two-tailed or one-tailed power calculation.
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
R2.1	scalar, or vector of length M; percent of variation explained by level 1 covariates for each outcome.
R2.2	scalar, or vector of length M; percent of variation explained by level 2 covariates for each outcome.
R2.3	scalar, or vector of length M; percent of variation explained by level 3 covariates for each outcome.
ICC.2	scalar, or vector of length M; level 2 (school) intraclass correlation.

ICC.3	scalar, or vector length M; level 3 (district) intraclass correlation.
omega.2	scalar, or vector of length M; ratio of variance of level 2 average impacts to variance of level 2 random intercepts.
omega.3	scalar, or vector of length M; ratio of variance of level 3 average impacts to variance of level 3 random intercepts.
rho	scalar; assumed correlation between all pairs of test statistics.
rho.matrix	matrix; alternate specification allowing a full matrix of correlations between test statistics. Must specify either rho or rho.matrix, but not both.
tnum	scalar; the number of test statistics to draw. Increasing tnum increases precision and computation time.
B	scalar; the number of permutations for Westfall-Young procedures.
parallel.WY.cores	number of cores to use for parallel processing of WY-SD.
drop.zero.outcomes	whether to report power results for outcomes with MDES = 0. If ALL MDES = 0, then the first outcome will not be dropped.
updateProgress	function to update progress bar (only used for PUMP shiny app).
validate.inputs	TRUE/FALSE; whether or not to check whether parameters are valid given the choice of d_m.
long.table	TRUE for table with power as rows, correction as columns, and with more verbose names. See 'transpose_power_table'.
verbose	TRUE/FALSE; Print out diagnostics of time, etc.
exact.where.possible	TRUE/FALSE; whether to do exact calculations when M=1, or use simulation. Default is TRUE.

Value

a pumpresult object containing power results.

See Also

For more detailed information about this function and the user choices, see the manuscript <doi:10.18637/jss.v108.i06>, which includes a detailed Technical Appendix including information about the designs and models and parameters.

Examples

```
pp <- pump_power(
   d_m = "d3.2_m3ff2rc",
   MTP = 'HO',
   nbar = 50,
   J = 30,
   K = 10,
   M = 5,
```

```
MDES = 0.125,
Tbar = 0.5, alpha = 0.05,
numCovar.1 = 1, numCovar.2 = 1,
R2.1 = 0.1, R2.2 = 0.1,
ICC.2 = 0.2, ICC.3 = 0.2,
omega.2 = 0, omega.3 = 0.1,
rho = 0.5)
```

---

pump_power_grid *Run pump_power on varying values of parameters (grid function)*

Description

This extension of 'pump_power()' will take lists of parameter values and run 'pump_power()' on all combinations of these values.

It can only assume the same MDES value for all outcomes due to this. (I.e., a vector of MDES values will be interpreted as a sequence of calls to pump_power, one for each MDES value given).

Each parameter in the parameter list can be a list, not scalar. It will cross all combinations of the list.

Usage

```
pump_power_grid(
  d_m,
  MTP = NULL,
  MDES,
  M = 1,
  nbar,
  J = 1,
  K = 1,
  propZero = NULL,
  numZero = NULL,
  Tbar,
  alpha = 0.05,
  numCovar.1 = NULL,
  numCovar.2 = NULL,
  numCovar.3 = NULL,
  R2.1 = NULL,
  R2.2 = NULL,
  R2.3 = NULL,
  ICC.2 = NULL,
  ICC.3 = NULL,
  omega.2 = NULL,
  omega.3 = NULL,
  rho = NULL,
  long.table = FALSE,
```

```
  verbose = FALSE,
  drop.unique.columns = TRUE,
  ...
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
MTP	string, or vector of strings; multiple testing procedure(s). See pump_info() for list of choices.
MDES	vector of numeric; This is *not* a list of MDES for each outcome, but rather a list of MDES to explore. Each value will be assumed held constant across all M outcomes.
M	scalar; the number of hypothesis tests (outcomes), including zero outcomes.
nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
propZero	Proportion of outcomes that have 0 impact (this will be used to override numZero, only one can be defined)
numZero	scalar; additional number of outcomes assumed to be zero. Please provide NumZero + length(MDES) = M, if length(MDES) is not 1.
Tbar	scalar; the proportion of samples that are assigned to the treatment.
alpha	scalar; the family wise error rate (FWER).
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
R2.1	scalar, or vector of length M; percent of variation explained by level 1 covariates for each outcome.
R2.2	scalar, or vector of length M; percent of variation explained by level 2 covariates for each outcome.
R2.3	scalar, or vector of length M; percent of variation explained by level 3 covariates for each outcome.
ICC.2	scalar, or vector of length M; level 2 (school) intraclass correlation.
ICC.3	scalar, or vector length M; level 3 (district) intraclass correlation.
omega.2	scalar, or vector of length M; ratio of variance of level 2 average impacts to variance of level 2 random intercepts.

omega.3	scalar, or vector of length M; ratio of variance of level 3 average impacts to variance of level 3 random intercepts.
rho	scalar; assumed correlation between all pairs of test statistics.
long.table	TRUE for table with power as rows, correction as columns, and with more verbose names. See 'transpose_power_table'.
verbose	logical; TRUE means print out some text as calls processed. FALSE do not.
drop.unique.columns	logical; drop all parameter columns that did not vary across the grid.
...	extra arguments passed to the underlying pump_power, pump_sample, or pump_mdes functions.

Value

a pumpgridresult object containing power results.

See Also

Other grid functions: pump_mdes_grid(), pump_sample_grid()

Examples

```
g <- pump_power_grid( d_m = "d3.2_m3ff2rc", MTP = c( "HO", "BF" ),
 MDES = 0.10, J = seq(5, 10, 1), M = 5, K = 7, nbar = 58,
 Tbar = 0.50, alpha = 0.15, numCovar.1 = 1,
 numCovar.2 = 1, R2.1 = 0.1, R2.2 = 0.7,
 ICC.2 = 0.25, ICC.3 = 0.25, rho = 0.4, tnum = 1000)
```

---

pump_sample *Estimate the required sample size (core function)*

Description

The user chooses the context (d_m), MTP, type of sample size, MDES, power definition, and choices of all relevant design parameters.

The functions performs a search algorithm, and returns the sample size value within the specified tolerance. For a list of choices for specific parameters, see pump_info().

Usage

```
pump_sample(
  d_m,
  MTP = NULL,
  typesample,
  MDES,
  M = 1,
  numZero = NULL,
```

```
  nbar = NULL,
  J = NULL,
  K = NULL,
  target.power,
  power.definition,
  alpha,
  two.tailed = TRUE,
  Tbar,
  numCovar.1 = 0,
  numCovar.2 = 0,
  numCovar.3 = 0,
  R2.1 = 0,
  R2.2 = 0,
  R2.3 = 0,
  ICC.2 = 0,
  ICC.3 = 0,
  rho = NULL,
  rho.matrix = NULL,
  omega.2 = 0,
  omega.3 = 0,
  B = 1000,
  max.steps = 20,
  tnum = 1000,
  start.tnum = tnum/10,
  final.tnum = 4 * tnum,
  parallel.WY.cores = 1,
  updateProgress = NULL,
  max_sample_size_nbar = 10000,
  max_sample_size_JK = 1000,
  tol = 0.01,
  give.optimizer.warnings = FALSE,
  verbose = FALSE
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
MTP	string, or vector of strings; multiple testing procedure(s). See pump_info() for list of choices.
typesample	string; type of sample size to calculate: "nbar", "J", or "K".
MDES	scalar or vector; the desired MDES values for each outcome. Please provide a scalar, a vector of length M, or vector of values for non-zero outcomes.
M	scalar; the number of hypothesis tests (outcomes), including zero outcomes.
numZero	scalar; additional number of outcomes assumed to be zero. Please provide NumZero + length(MDES) = M, if length(MDES) is not 1.

nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
target.power	target power for search algorithm.
power.definition	see pump_info() for possible power definitions.
alpha	scalar; the family wise error rate (FWER).
two.tailed	scalar; TRUE/FALSE for two-tailed or one-tailed power calculation.
Tbar	scalar; the proportion of samples that are assigned to the treatment.
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
R2.1	scalar, or vector of length M; percent of variation explained by level 1 covariates for each outcome.
R2.2	scalar, or vector of length M; percent of variation explained by level 2 covariates for each outcome.
R2.3	scalar, or vector of length M; percent of variation explained by level 3 covariates for each outcome.
ICC.2	scalar, or vector of length M; level 2 (school) intraclass correlation.
ICC.3	scalar, or vector length M; level 3 (district) intraclass correlation.
rho	scalar; assumed correlation between all pairs of test statistics.
rho.matrix	matrix; alternate specification allowing a full matrix of correlations between test statistics. Must specify either rho or rho.matrix, but not both.
omega.2	scalar, or vector of length M; ratio of variance of level 2 average impacts to variance of level 2 random intercepts.
omega.3	scalar, or vector of length M; ratio of variance of level 3 average impacts to variance of level 3 random intercepts.
B	scalar; the number of permutations for Westfall-Young procedures.
max.steps	how many steps allowed before terminating.
tnum	max number of samples for first iteration of search algorithm.
start.tnum	number of samples to start search (this will increase with each step).
final.tnum	number of samples for final draw.
parallel.WY.cores	number of cores to use for parallel processing of WY-SD.
updateProgress	function to update progress bar (only used for PUMP shiny app).

`max_sample_size_nbar`
: scalar; default upper bound for nbar for search algorithm.

`max_sample_size_JK`
: scalar; default upper bound for J or K for search algorithm.

`tol`
: tolerance for target power, defaults to 0.01 (1 This parameter controls when the search is done: when estimated power (checked with 'final.tnum' iterations) is within 'tol', the search stops.

`give.optimizer.warnings`
: whether to return verbose optimizer warnings.

`verbose`
: TRUE/FALSE; Print out diagnostics of time, etc.

Value

a pumpresult object containing sample size results.

See Also

For more detailed information about this function and the user choices, see the manuscript <doi:10.18637/jss.v108.i06>, which includes a detailed Technical Appendix including information about the designs and models and parameters.

Examples

```
J <- pump_sample(
  d_m = 'd2.1_m2fc',
  MTP = 'HO',
  power.definition = 'D1indiv',
  typesample = 'J',
  target.power = 0.8,
  nbar = 50,
  M = 3,
  MDES = 0.125,
  Tbar = 0.5, alpha = 0.05,
  numCovar.1 = 1,
  R2.1 = 0.1, ICC.2 = 0.05, rho = 0.2,
  tnum = 1000)
```

---

pump_sample_grid *Run pump_sample on varying values of parameters (grid function)*

Description

See pump_power_grid() for further details.

Usage

```
pump_sample_grid(
  d_m,
  MTP = NULL,
  M = 1,
  target.power,
  power.definition,
  tol = 0.01,
  MDES = NULL,
  propZero = NULL,
  numZero = NULL,
  typesample,
  nbar = NULL,
  J = NULL,
  K = NULL,
  Tbar,
  alpha,
  numCovar.1 = NULL,
  numCovar.2 = NULL,
  numCovar.3 = NULL,
  R2.1 = NULL,
  R2.2 = NULL,
  R2.3 = NULL,
  ICC.2 = NULL,
  ICC.3 = NULL,
  omega.2 = NULL,
  omega.3 = NULL,
  rho = NULL,
  verbose = FALSE,
  drop.unique.columns = TRUE,
  ...
)
```

Arguments

d_m	string; a single context, which is a design and model code. See pump_info() for list of choices.
MTP	string, or vector of strings; multiple testing procedure(s). See pump_info() for list of choices.
M	scalar; the number of hypothesis tests (outcomes), including zero outcomes.
target.power	target power for search algorithm.
power.definition	see pump_info() for possible power definitions.
tol	tolerance for target power, defaults to 0.01 (1 This parameter controls when the search is done: when estimated power (checked with ‘final.tnum‘ iterations) is within ‘tol‘, the search stops.

MDES	scalar or vector; the desired MDES values for each outcome. Please provide a scalar, a vector of length M, or vector of values for non-zero outcomes.
propZero	Proportion of outcomes that have 0 impact (this will be used to override numZero, only one can be defined)
numZero	scalar; additional number of outcomes assumed to be zero. Please provide NumZero + length(MDES) = M, if length(MDES) is not 1.
typesample	string; type of sample size to calculate: "nbar", "J", or "K".
nbar	scalar; the harmonic mean of the number of level 1 units per level 2 unit (students per school). Note that this is not the total number of level 1 units, but instead the number of level 1 units nested within each level 2 unit, so the total number of level 1 units is nbar x J x K.
J	scalar; the harmonic mean of number of level 2 units per level 3 unit (schools per district). Note that this is not the total number of level 2 units, but instead the number of level 2 units nested within each level 3 unit, so the total number of level 2 units is J x K.
K	scalar; the number of level 3 units (districts).
Tbar	scalar; the proportion of samples that are assigned to the treatment.
alpha	scalar; the family wise error rate (FWER).
numCovar.1	scalar; number of level 1 (individual) covariates.
numCovar.2	scalar; number of level 2 (school) covariates.
numCovar.3	scalar; number of level 3 (district) covariates.
R2.1	scalar, or vector of length M; percent of variation explained by level 1 covariates for each outcome.
R2.2	scalar, or vector of length M; percent of variation explained by level 2 covariates for each outcome.
R2.3	scalar, or vector of length M; percent of variation explained by level 3 covariates for each outcome.
ICC.2	scalar, or vector of length M; level 2 (school) intraclass correlation.
ICC.3	scalar, or vector length M; level 3 (district) intraclass correlation.
omega.2	scalar, or vector of length M; ratio of variance of level 2 average impacts to variance of level 2 random intercepts.
omega.3	scalar, or vector of length M; ratio of variance of level 3 average impacts to variance of level 3 random intercepts.
rho	scalar; assumed correlation between all pairs of test statistics.
verbose	TRUE/FALSE; Print out diagnostics of time, etc.
drop.unique.columns	logical; drop all parameter columns that did not vary across the grid.
...	extra arguments passed to the underlying pump_power, pump_sample, or pump_mdes functions.

Value

a pumpgridresult object containing sample results.

See Also

Other grid functions: pump_mdes_grid(), pump_power_grid()

Examples

```
g <- pump_sample_grid(d_m = "d3.2_m3ff2rc", typesample = "J",
  MTP = "HO", MDES = 0.10, target.power = c( 0.50, 0.80 ),
  power.definition = "min1", tol = 0.03,
  M = 5, K = 7, nbar = 58, Tbar = 0.50,
  alpha = 0.15, numCovar.1 = 1, numCovar.2 = 1,
  R2.1 = 0.1, R2.2 = 0.7, ICC.2 = 0.25, ICC.3 = 0.25,
  rho = 0.4, tnum = 400)
```

---

transpose_power_table *Convert power table from wide to long (result function)*

Description

Transform table returned from pump_power to a long format table or to a wide format table.

Usage

```
transpose_power_table(power_table, M = NULL)
```

Arguments

power_table	pumpresult object for a power result (not mdes or sample). (It can also take a raw dataframe of the wide table to convert to long, as an internal helper method.)
M	scalar; set if power_table is a data.frame without set number of outcomes. Usually ignore this.

Value

data.frame of power results in long format.

update.pumpgridresult *Update a pump grid call, tweaking some parameters (core function)*

Description

Works on objects returned by ‘update_grid()‘; calls ‘update_grid()‘.

Usage

```
## S3 method for class 'pumpgridresult'
update(object, ...)
```

Arguments

object	A pumpgridresult object.
...	Additional arguments, i.e., the arguments you would pass to the ‘pump_power()‘, ‘pump_mdes()‘ and ‘pump_sample()‘, that will replace the existing parameters of the object.

See Also

[update_grid()]

update.pumpresult *Update a pump call, tweaking some parameters (core function)*

Description

Works on objects returned by pump_power(), pump_mdes(), or pump_sample(). One of the optional parameters can be a ‘type = something‘ argument, where the "something" is either "power", "sample", or "mdes", if the call should be shifted to a different pump call (pump_power, pump_sample, or pump_mdes, respectively).

Usage

```
## S3 method for class 'pumpresult'
update(object, type = NULL, ...)
```

Arguments

object	pump result object.
type	string; can be "power", "mdes" or "sample", sets the type of the updated call (can be different from original).
...	parameters as specified in ‘pump_power‘, ‘pump_mdes‘, and ‘pump_sample‘ that should be overwritten.

Value

a pumpresult object: results of a new call using parameters of old object with newly specified parameters replaced.

Examples

```
ss <- pump_sample( d_m = "d2.1_m2fc", MTP = "HO",
  typesample = "J", nbar = 200, power.definition = "min1",
  M = 5, MDES = 0.05, target.power = 0.5, tol = 0.05,
  Tbar = 0.50, alpha = 0.05, numCovar.1 = 5, R2.1 = 0.1,
  ICC.2 = 0.15, rho = 0, final.tnum = 1000 )

up <- update(ss, nbar = 40, tnum = 2000 )
```

---

update_grid *Update a single pump call to a grid call (grid function)*

Description

Take a pumpresult and provide lists of parameters to explore various versions of the initial scenario.

Usage

```
update_grid(x, ...)
```

Arguments

x	pump result object.
...	list of parameters to expand into a grid.

Value

a pumpgridresult object; result of calling corresponding grid.

Examples

```
pp <- pump_power(d_m = "d2.1_m2fc", MTP = "HO",
  nbar = 200, J = 20, MDES = 0.2, M = 3,
  Tbar = 0.50, alpha = 0.05, numCovar.1 = 5,
  R2.1 = 0.1, ICC.2 = 0.05, rho = 0, tnum = 500)

gd <- update_grid( pp, J = c( 10, 20, 30 ) )
```

Index

∗ **grid functions**
    pump_mdes_grid, 22
    pump_power_grid, 28
    pump_sample_grid, 33
∗ **pump_info**
    parse_d_m, 10
[.pumpresult *(*pumpresult*)*, 16
[[.pumpresult *(*pumpresult*)*, 16

as.data.frame.pumpresult *(*pumpresult*)*, 16

calc_df, 2
check_cor, 3
convert_params, 5

d_m *(*pumpresult*)*, 16
design *(*pumpresult*)*, 16
dim.pumpresult *(*pumpresult*)*, 16

gen_base_sim_data, 5
gen_cluster_ids, 6
gen_corr_matrix, 7
gen_sim_data, 7
gen_T.x, 8
gen_Yobs, 9
get_power_results, 10

is.pumpgridresult *(*pumpgridresult*)*, 15
is.pumpresult *(*pumpresult*)*, 16

params *(*pumpresult*)*, 16
parse_d_m, 10
plot.pumpgridresult, 11
plot.pumpresult, 12
power_curve, 14
print.pumpgridresult *(*pumpgridresult*)*, 15
print.pumpresult *(*pumpresult*)*, 16
print_context, 15
pump_info, 19
pump_mdes, 19
pump_mdes_grid, 22, *30*, *36*
pump_power, 25
pump_power_grid, *24*, 28, *36*
pump_sample, 30
pump_sample_grid, *24*, *30*, 33
pump_type *(*pumpresult*)*, 16
pumpgridresult, 15
pumpresult, 16

search_path *(*pumpresult*)*, 16
summary.pumpgridresult *(*pumpgridresult*)*, 15
summary.pumpresult *(*pumpresult*)*, 16

transpose_power_table, 36

update.pumpgridresult, 37
update.pumpresult, 37
update_grid, 38