Package 'MOST'

October 12, 2022

Type Package

Title Multiphase Optimization Strategy

Version 0.1.2

Depends R (>= 2.15.0)

Encoding UTF-8

Description Provides functions similar to the 'SAS' macros previously provided to accompany Collins, Dziak, and Li (2009) <DOI:10.1037/a0015826> and Dziak, Nahum-Shani, and Collins (2012) <DOI:10.1037/a0026972>, papers which outline practical benefits and challenges of factorial and fractional factorial experiments for scientists interested in developing biological and/or behavioral interventions, especially in the context of the multiphase optimization strategy (see Collins, Kugler & Gwadz 2016) <DOI:10.1007/s10461-015-1145-4>. The package currently contains three functions. First, RelativeCosts1() draws a graph of the relative cost of complete and reduced factorial designs versus other alternatives. Second, RandomAssignmentGenerator() returns a dataframe which contains a list of random numbers that can be used to conveniently assign participants to conditions in an experiment with many conditions. Third, FactorialPowerPlan() estimates the power, detectable effect size, or required sample size of a factorial or fractional factorial experiment, for main effects or interactions, given several possible choices of effect size metric, and allowing pretests and clustering.

License GPL (>= 2)

NeedsCompilation no

RoxygenNote 7.2.0

Author Linda Collins [aut],
Liying Huang [aut],
John Dziak [aut, cre]

Maintainer John Dziak <dziakj1@gmail.com>

Repository CRAN

Date/Publication 2022-06-23 22:20:08 UTC

R topics documented:

`FactorialPowerPlan` *sample size, power and effect size calculations for a factorial or fractional factorial experiment*

Description

There are three ways to use this function:

1. Estimate power available from a given sample size and a given effect size.
2. Estimate sample size needed for a given power and a given effect size.
3. Estimate effect size detectable from a given power at a given sample size.

That is, there are three main pieces of information: power, sample size, and effect size. The user provides two of them, and this function calculates the third.

Usage

```
FactorialPowerPlan(
  alpha = 0.05,
  assignment = "unclustered",
  change_score_icc = NULL,
  cluster_size = NULL,
  cluster_size_sd = NULL,
  d_main = NULL,
  effect_size_ratio = NULL,
  icc = NULL,
  model_order = 1,
  nclusters = NULL,
  nfactors = 1,
  ntotal = NULL,
  power = NULL,
  pre_post_corr = NULL,
  pretest = "none",
  raw_coef = NULL,
  raw_main = NULL,
  sigma_y = NULL,
  std_coef = NULL
)
```

Arguments

alpha	Two sided Type I error level for the test to be performed(default=0.05).
assignment	One of three options: (default=unclustered) 1. "independent" or equivalently "unclustered" 2. "within" or equivalently "within_clusters" 3. "between" or equivalently "between_clusters" Clusters in this context are preexisting units within which responses may be dependent (e.g., clinics or schools). A within-cluster experiment involves randomizing individual members, while a between-cluster experiment involves randomizing clusters as whole units (see Dziak, Nahum-Shani, and Collins, 2012) <DOI:10.1037/a0026972>
change_score_icc	The intraclass correlation of the change scores (posttest minus pretest). Relevant only if assignment is between clusters and there is a pretest.
cluster_size	The mean number of members in each cluster. Relevant only if assignment is between clusters or within clusters.
cluster_size_sd	Relevant only if assignment is between clusters. The standard deviation of the number of members in each cluster (the default is 0 which means that the clusters are expected to be of equal size).
d_main	Effect size measure: standardized mean difference raw_main/sigma_y.
effect_size_ratio	Effect size measure: signal to noise ratio raw_coef^2/sigma_y^2.
icc	Relevant only if assignment is between clusters or within clusters. The intraclass correlation of the variable of interest in the absence of treatment.
model_order	The highest order term to be included in the regression model in the planned analysis (1=main effects, 2=two-way interactions, 3=three-way interactions, etc.); must be >= 1 and <= nfactors (default=1).
nclusters	The total number of clusters available (for between clusters or within clusters assignment).
nfactors	The number of factors (independent variables) in the planned experiment(default=1).
ntotal	The total sample size available (for unclustered assignment. For clustered assignment, use "cluster_size" and "nclusters."
power	If specified: The desired power of the test. If returned in the output list: The expected power of the test.
pre_post_corr	Relevant only if there is a pretest. The correlation between the pretest and the posttest.
pretest	One of three options: 1. "no" or "none" for no pretest. 2. "covariate" for pretest to be entered as a covariate in the model. 3. "repeated" for pretest to be considered as a repeated measure.

The option "yes" is also allowed and is interpreted as "repeated." The option "covariate" is not allowed if assignment is between clusters. This is because predicting power for covariate-adjusted cluster-level randomization is somewhat complicated, although it can be approximated in practice by using the formula for the repeated-measures cluster-level randomization (see simulations in Dziak, Nahum-Shani, and Collins, 2012).

`raw_coef` Effect size measure: unstandardized effect-coded regression coefficient.

`raw_main` Effect size measure: unstandardized mean difference.

`sigma_y` The assumed standard deviation of the response variable after treatment, within each treatment condition (i.e., adjusting for treatment but not adjusting for post-test). This statement must be used if the effect size argument used is either "raw_main" or "raw_coef".

`std_coef` Effect size measure: standardized effect-coded regression coefficient raw_coef/sigma_y.

Value

A list with power, sample size and effect size.

Examples

```
FactorialPowerPlan(assignment="independent",
                   model_order=2,
                   nfactors=5,
                   ntotal=300,
                   raw_main=3,
                   sigma_y=10)

FactorialPowerPlan(assignment="independent",
                   model_order=2,
                   nfactors=5,
                   ntotal=300,
                   pre_post_corr=.6,
                   pretest="covariate",
                   raw_main=3,
                   sigma_y=10)
```

RandomAssignmentGenerator

Random Assignment Generator for a Factorial Experiment with Many Conditions

---

Description

This function provides a list of random numbers that can be used to assign participants to conditions (cells) in an experiment with many conditions, such as a factorial experiment. The randomization is restricted as follows: if the number of participants available is a multiple of the number of conditions, then cell sizes will be balanced; otherwise, they will be as near balanced as possible.

Usage

```
RandomAssignmentGenerator(N, C)
```

Arguments

N	The total number of participants to be randomized.
C	The total number of conditions for the experiment you are planning. Note that f or a complete factorial experiment having k factors, there will be 2^k conditions.

Value

A dataframe with 1 variable ranList with N observations, each observation of ranList provides a random number for each participant. This will be a number from 1 to C. For example, if the 4th number in the list is 7, the 4th subject is randomly assigned to experiment condition 7. Random numbers will be generated so that the experiment is approximately balanced.

Examples

```
result <- RandomAssignmentGenerator(35,17)
print(result)
```

RelativeCosts1 *The relative cost of reduced factorial designs*

Description

Draw a graph of the relative cost of complete factorial, fractional factorial, and unbalanced reduced factorial designs, as presented by Collins, Dziak and Li (2009; https://www.ncbi.nlm.nih.gov/pubmed/19719358). For purposes of illustration, a normally distributed response variable, dichotomous factors, and negligible interactions are assumed in this function.

Usage

```
RelativeCosts1(
  number_of_factors,
  desired_fract_resolution = 4,
  min_target_d_per_factor = 0.2,
  condition_costlier_than_subject = 1,
  max_graph_ratio = 5
)
```

Arguments

number_of_factors	The number of factors to be tested.
desired_fract_resolution	The desired resolution of the fractional factorial experiment to be compared. The default value is set to be 4.

`min_target_d_per_factor`
: The minimum Cohen's d (standardized difference, i.e., response difference between levels on a given factor, divided by response standard deviation) that is desired to be detected with 80 The default value is set to be 0.2.

`condition_costlier_than_subject`
: The default value is set to be 1.

`max_graph_ratio`
: The default value is set to be 5.

Examples

```
RelativeCosts1(number_of_factors = 9,
               desired_fract_resolution = 4,
               min_target_d_per_factor = .2,
               condition_costlier_than_subject=1,
               max_graph_ratio = 4)
```

Index