Package 'DesignLibrary'

August 9, 2018

Type Package

Title Library of Research Designs

Version 0.1.0

Description A simple interface to build designs
using the package 'DeclareDesign'. In one line of code, users can specify
the parameters of individual designs and diagnose their properties.
The designers can also be used to compare performance of a given design
across a range of combinations of parameters, such as effect size, sample
size, and assignment probabilities.

URL https://declaredesign.org/library/,
https://github.com/DeclareDesign/DesignLibrary

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends DeclareDesign (>= 0.10.0), R (>= 3.4.0), randomizr (>=
0.16.1), fabricatr (>= 0.4.0), estimatr (>= 0.10.0)

Imports devtools, rlang, methods

Suggests knitr, testthat, systemfit, ggplot2, broom, rmarkdown

Enhances Matching

RoxygenNote 6.1.0

VignetteBuilder knitr

NeedsCompilation no

Author Graeme Blair [aut],
Jasper Cooper [aut, cre],
Alexander Coppock [aut],
Macartan Humphreys [aut],
Clara Bicalho [aut],
Neal Fultz [aut],
Lily Medina [aut]

Maintainer Jasper Cooper <jjc2247@columbia.edu>

Repository CRAN

Date/Publication 2018-08-09 18:30:04

R topics documented:

`block_cluster_two_arm_designer` *Create a two-arm design with blocks and clusters*

Description

Builds a two-arm design with blocks and clusters.

Usage

```
block_cluster_two_arm_designer(N_blocks = 1, N_clusters_in_block = 100,
  N_i_in_cluster = 1, sd_block = 0.5, sd_cluster = 0.5,
  sd_i_0 = sqrt(max(0, 1 - sd_block^2 - sd_cluster^2)),
  sd_i_1 = sd_i_0, rho = 1, prob = 0.5, control_mean = 0,
  ate = 0, treatment_mean = control_mean + ate)
```

Arguments

`N_blocks`	An integer. Number of blocks. Defaults to 1 for no blocks.
`N_clusters_in_block`	An integer. Number of clusters in each block. This is the total `N` when `N_blocks` and `N_i_in_cluster` are at default values.
`N_i_in_cluster`	An integer. Individuals per cluster. Defaults to 1 for no clusters.
`sd_block`	A nonnegative number. Standard deviation of block level shocks.
`sd_cluster`	A nonnegative number. Standard deviation of cluster level shock.
`sd_i_0`	A nonnegative number. Standard deviation of individual level shock in control. For small `sd_block` and `sd_cluster`, `sd_i_0` defaults to make total variance = 1.

sd_i_1	A nonnegative number. Standard deviation of individual level shock in treatment. Defaults to sd_i_0.
rho	A number in [-1,1]. Correlation in individual shock between potential outcomes for treatment and control.
prob	A number in [0,1]. Treatment assignment probability.
control_mean	A number. Average outcome in control.
ate	A number. Average treatment effect. Alternative to specifying treatment_mean. Note that ate is an argument for the designer but it does not appear as an argument in design code (design code uses control_mean and treatment_mean only.) only.
treatment_mean	A number. Average outcome in treatment. Note: if treatment_mean is not provided then it is calculated from ate. If both ate and treatment_mean are provided then only treatment_mean is used.

Details

Units are assigned to treatment using complete block cluster random assignment. Treatment effects can be specified either by providing control_mean and treatment_mean or by specifying an ate. Estimation uses differences in means accounting for blocks and clusters.

Total N is given by N_blocks*N_clusters_in_block*N_i_in_cluster

Normal shocks can be specified at the individual, cluster, and block levels. If individual level shocks are not specified and cluster and block level variances sum to less than 1, then individual level shocks are set such that total variance in outcomes equals 1.

Key limitations: The designer assumes covariance between potential outcomes at individual level only.

Value

A block cluster two-arm design.

Author(s)

DeclareDesign Team

Examples

```
# Generate a design using default arguments:
block_cluster_two_arm_design <- block_cluster_two_arm_designer()
```

`cluster_sampling_designer` *Create a design for cluster random sampling*

Description

Builds a cluster sampling design of a population with `N_clusters` containing `N_subjects_per_cluster`. Estimations sample `n_clusters` each comprising `n_subjects_per_cluster` units. Outcomes within clusters have ICC approximately equal to `ICC`.

Usage

```
cluster_sampling_designer(N_clusters = 1000,
  N_subjects_per_cluster = 50, n_clusters = 100,
  n_subjects_per_cluster = 10, icc = 0.2)
```

Arguments

`N_clusters`	An integer. Total number of clusters in the population.
`N_subjects_per_cluster`	An integer of vector of integers of length `N_clusters`. Total number of subjects per cluster in the population.
`n_clusters`	An integer. Number of clusters to sample.
`n_subjects_per_cluster`	An integer. Number of subjects to sample per cluster.
`icc`	A number in [0,1]. Intra-cluster Correlation Coefficient (ICC).

Details

Key limitations: The design assumes clusters draw with equal probability (rather than, for example, proportionate to size).

Value

A cluster sampling design.

Author(s)

DeclareDesign Team

Examples

```
# To make a design using default arguments:
cluster_sampling_design <- cluster_sampling_designer()
# A design with varying cluster size
cluster_sampling_design <- cluster_sampling_designer(
  N_clusters = 10, N_subjects_per_cluster = 3:12,
  n_clusters = 5,  n_subjects_per_cluster = 2)
```

get_design_code *Get the code from a design*

Description

Get the code from a design

Usage

```
get_design_code(design)
```

Arguments

design	A design that has code as an attribute.

match.call.defaults *Argument matching with defaults*

Description

This is a version of match.call which also includes default arguments.

Usage

```
match.call.defaults(definition = sys.function(sys.parent()),
  call = sys.call(sys.parent()), expand.dots = TRUE,
  envir = parent.frame(2L))
```

Arguments

definition	a function, by default the function from which match.call is called. See details.
call	an unevaluated call to the function specified by definition, as generated by call.
expand.dots	ogical. Should arguments matching ... in the call be included or left as a ... argument?
envir	an environment, from which the ... in call are retrieved, if any.

Value

An object of class call.

Author(s)

Neal Fultz

References

http://stackoverflow.com/questions/14397364/match-call-with-default-arguments/

Examples

```
foo <- function(x=NULL,y=NULL,z=4, dots=TRUE, ...) {
  match.call.defaults(expand.dots=dots)
}

foo(4,nugan='hand')
foo(dots=FALSE,who='ami')
```

---

mediation_analysis_designer

Create a design for mediation analysis

Description

A mediation analysis design that examines the effect of treatment (Z) on mediator (M) and the effect of mediator (M) on outcome (Y) (given Z=0) as well as direct effect of treatment (Z) on outcome (Y) (given M=0). Analysis is implemented using an interacted regression model. Note this model is not guaranteed to be unbiased despite randomization of Z because of possible violations of sequential ignorability.

Usage

```
mediation_analysis_designer(N = 200, a = 1, b = 0.4, c = 0,
  d = 0.5, rho = 0)
```

Arguments

N	An integer. Size of sample.
a	A number. Parameter governing effect of treatment (Z) on mediator (M).
b	A number. Effect of mediator (M) on outcome (Y) when Z=0.
c	A number. Interaction between mediator (M) and (Z) for outcome (Y).
d	A number. Direct effect of treatment (Z) on outcome (Y), when M = 0.
rho	A number in [-1,1]. Correlation between mediator (M) and outcome (Y) error terms. Non zero correlation implies a violation of sequential ignorability.

Value

A mediation analysis design.

Author(s)

DeclareDesign Team

Examples

```
# Generate a mediation analysis design using default arguments:
mediation_1 <- mediation_analysis_designer()
get_estimands(mediation_1)
## Not run:
diagnose_design(mediation_1, sims = 1000)

## End(Not run)

# A design with a violation of sequential ignorability and heterogeneous effects:
mediation_2 <- mediation_analysis_designer(a = 1, rho = .5, c = 1, d = .75)
get_estimands(mediation_2)
## Not run:
diagnose_design(mediation_2, sims = 1000)

## End(Not run)
```

---

multi_arm_designer *Create a design with multiple experimental arms*

Description

This designer creates a design `m_arms` experimental arms, each assigned with equal probabilities.

Usage

```
multi_arm_designer(N = 30, m_arms = 3, means = rep(0, m_arms),
  sds = rep(1, m_arms), conditions = 1:m_arms, fixed = NULL)
```

Arguments

`N`	An integer. Sample size.
`m_arms`	An integer. Number of arms.
`means`	A numeric vector of length `m_arms`. Average outcome in each arm.
`sds`	A nonnegative numeric vector of length `m_arms`. Standard deviations for each of the arms.
`conditions`	A vector of length `m_arms`. The names of each arm. It can be numeric or a character without blank spaces.
`fixed`	A character vector. Names of arguments to be fixed in design. By default `m_arms` and `conditions` are always fixed.

Value

A function that returns a design.

Author(s)

DeclareDesign Team

Examples

```
# To make a design using default arguments:
design <- multi_arm_designer()


# A design with different mean and sd in each arm
design <- multi_arm_designer(means = c(0, 0.5, 2), sd =  c(1, 0.1, 0.5))

design <- multi_arm_designer(N = 80, m_arms = 4, means = 1:4,
                             fixed = c("means", "sds"))
```

`pretest_posttest_designer` *Create a pretest-posttest design*

Description

Produces designs in which an outcome Y is observed pre- and post-treatment. The design allows for individual post-treatment outcomes to be correlated with pre-treatment outcomes and for at-random missingness in the observation of post-treatment outcomes.

Usage

```
pretest_posttest_designer(N = 100, ate = 0.25, sd_1 = 1, sd_2 = 1,
  rho = 0.5, attrition_rate = 0.1)
```

Arguments

`N`	An integer. Size of sample.
`ate`	A number. Average treatment effect.
`sd_1`	Non negative number. Standard deviation of period 1 shocks.
`sd_2`	Non negative number. Standard deviation of period 2 shocks.
`rho`	A number in [-1,1]. Correlation in outcomes between pre- and post-test.
`attrition_rate`	A number in [0,1]. Proportion of respondents in pre-test data that appear in post-test data.

Value

A pretest-posttest design.

Author(s)

DeclareDesign Team

Examples

```
# Generate a pre-test post-test design using default arguments:
pretest_posttest_design <- pretest_posttest_designer()
```

`randomized_response_designer`
Create a randomized response design

Description

Produces a (forced) randomized response design that measures the share of individuals with a given trait `prevalence_trait` in a population of size `N`. Probability of forced response ("Yes") is given by `prob_forced_yes`, and rate at which individuals with trait lie is given by `withholding_rate`.

Usage

```
randomized_response_designer(N = 1000, prob_forced_yes = 0.6,
  prevalence_rate = 0.1, withholding_rate = 0.5)
```

Arguments

`N`	An integer. Size of sample.
`prob_forced_yes`	A number. Probability of a forced yes.
`prevalence_rate`	A number. Probability that individual has the sensitive trait.
`withholding_rate`	A number. Probability that an individual with the sensitive trait hides it.

Details

`randomized_response_designer` employs a specific variation of randomized response designs in which respondents are required to report a fixed answer to the sensitive question with a given probability (see Blair, Imai, and Zhou (2015) for alternative applications and estimation strategies).

Value

A randomized response design.

Author(s)

DeclareDesign Team

Examples

```
# Generate a randomized response design using default arguments:
randomized_response_design <- randomized_response_designer()
```

`regression_discontinuity_designer`

Create a regression discontinuity design

Description

Builds a design with sample from population of size `N`. The average treatment effect local to the cutpoint is equal to `tau`. It allows for specification of the order of the polynomial regression (`poly_order`), cutoff value on the running variable (`cutoff`), and size of bandwidth around the cutoff (`bandwidth`).

Usage

```
regression_discontinuity_designer(N = 1000, tau = 0.15, cutoff = 0.5,
  bandwidth = 0.5, poly_order = 4)
```

Arguments

`N`	An integer. Size of population to sample from.
`tau`	A number. Difference in potential outcomes functions at the threshold.
`cutoff`	A number in (0,1). Threshold on running variable beyond which units are treated.
`bandwidth`	A number. Bandwidth around threshold from which to include units.
`poly_order`	An integer. Order of the polynomial regression used to estimate the jump at the cutoff.

Value

A regression discontinuity design.

Author(s)

DeclareDesign Team

Examples

```
# Generate a regression discontinuity design using default arguments:
regression_discontinuity_design <- regression_discontinuity_designer()
```

simple_factorial_designer *Create a simple factorial design*

Description

Builds a two-by-two factorial design in which assignments to each factor are independent of each other.

Usage

```
simple_factorial_designer(N = 100, prob_A = 0.5, prob_B = 0.5,
  w_A = 0.5, w_B = 0.5, outcome_means = rep(0, 4),
  mean_A0B0 = outcome_means[1], mean_A0B1 = outcome_means[2],
  mean_A1B0 = outcome_means[3], mean_A1B1 = outcome_means[4],
  outcome_sds = rep(1, 4))
```

Arguments

N	An integer. Size of sample.
prob_A	A number in [0,1]. Probability of assignment to treatment A.
prob_B	A number in [0,1]. Probability of assignment to treatment B.
w_A	A number. Weight placed on A=1 condition in definition of "average effect of B" estimand.
w_B	A number. Weight placed on B=1 condition in definition of "average effect of A" estimand.
outcome_means	A vector of length 4. Average outcome in each A,B condition, in order AB = 00, 01, 10, 11. Values overridden by mean_A0B0, mean_A0B1, mean_A1B0, if provided mean_A1B1.
mean_A0B0	A number. Mean outcome in A=0, B=0 condition.
mean_A0B1	A number. Mean outcome in A=0, B=1 condition.
mean_A1B0	A number. Mean outcome in A=1, B=0 condition.
mean_A1B1	A number. Mean outcome in A=1, B=1 condition.
outcome_sds	A non-negative 4-vector. Standard deviation in each condition, in order AB = 00, 01, 10, 11.

Details

Three types of estimand are declared: weighted averages of the average treatment effects of each treatment over the two conditions of the other treatment and the difference in treatment effects of each over conditions of the other.

Treatment A is assigned first and then Treatment B within blocks defined by treatment A. Thus eg if there are 6 units 3 are guaranteed to receive treatment A but the number receiving treatment B is stochastic.

Units are assigned to treatment using complete random assignment. Potential outcomes follow a normal distribution.

See multi_arm_designer for a factorial design with non independent assignments.

Value

A two-by-two factorial design.

Author(s)

DeclareDesign Team

Examples

```
design <- simple_factorial_designer(outcome_means = c(0,0,0,1))

# A design biased for the specified estimands:
design <- simple_factorial_designer(outcome_means = c(0,0,0,1), prob_A = .8, prob_B = .2)
## Not run:
diagnose_design(design)

## End(Not run)

# A design with estimands that "match" the assignment:
design <- simple_factorial_designer(outcome_means = c(0,0,0,1),
                                    prob_A = .8, prob_B = .2,
                                    w_A = .8, w_B = .2)
## Not run:
diagnose_design(design)

## End(Not run)

# Compare power with and without interactions, given same average effects in each arm
designs <- redesign(simple_factorial_designer(),
                    outcome_means = list(c(0,0,0,1), c(0,.5,.5,1)))
## Not run:
diagnose_design(designs)

## End(Not run)
```

---

simple_spillover_designer

Create a simple design with spillovers

Description

Builds a design with N_groups groups each containing N_i_group individuals. Potential outcomes exhibit spillovers: if any individual in a group receives treatment, the effect is spread equally among members of the group.

Usage

```
simple_spillover_designer(N_groups = 80, N_i_group = 3, sd = 0.2,
  gamma = 2)
```

Arguments

N_groups	An integer. Number of groups.
N_i_group	Number of units in each group. Can be scalar or vector of length N_groups.
sd	A number. Standard deviation of individual level shock.
gamma	A number. Parameter that controls whether spillovers within groups substitute or complement each other.

Details

Parameter gamma controls interactions between spillover effects.For gamma=1 for ever $1 given to a member of a group, each member receives $1N_i_group no matter how many others are already treated. For gamma>1 (<1) for ever $1 given to a member of a group, each member receives an amount that depends negatively (positively) on the number already treated.

The default estimand is the average difference across subjects between no one treated and only that subject treated.

Value

A simple spillover design.

Author(s)

DeclareDesign Team

Examples

```
# Generate a simple spillover design using default arguments:
simple_spillover_design <- simple_spillover_designer()
```

---

simple_two_arm_designer

Create a simple two arm design

Description

Builds a design with one treatment and one control arm. Treatment effects can be specified either by providing control_mean and treatment_mean or by specifying an ate.

Usage

```
simple_two_arm_designer(N = 100, prob = 0.5, control_mean = 0,
  control_sd = 1, ate = 1, treatment_mean = control_mean + ate,
  treatment_sd = control_sd, rho = 1)
```

Arguments

N	An integer. Sample size.
prob	A number in [0,1]. Probability of assignment to treatment.
control_mean	A number. Average outcome in control.
control_sd	A positive number. Standard deviation in control.
ate	A number. Average treatment effect.
treatment_mean	A number. Average outcome in treatment. Overrides ate if both specified.
treatment_sd	A non-negative number. Standard deviation in treatment. By default equals control_sd.
rho	A number in [-1,1]. Correlation between treatment and control outcomes.

Details

Units are assigned to treatment using complete random assignment. Potential outcomes follow a normal distribution.

Value

A simple two-arm design.

Author(s)

DeclareDesign Team

Examples

```
#Generate a simple two-arm design using default arguments
simple_two_arm_design <- simple_two_arm_designer()
```

---

two_arm_attrition_designer

Create design with risk of attrition or post treatment conditioning

---

Description

Creates a two arm design with application for when estimand of interest is conditional on a post treatment outcome (the effect on Y given R) or data is conditionally observed (Y given R). See 'Details' for more information on the data generating process.

Usage

```
two_arm_attrition_designer(N = 100, a_R = 0, b_R = 1, a_Y = 0,
  b_Y = 1, rho = 0)
```

Arguments

N	An integer. Size of sample.
a_R	A number. Constant in equation relating treatment to responses.
b_R	A number. Slope coefficient in equation relating treatment to responses.
a_Y	A number. Constant in equation relating treatment to outcome.
b_Y	A number. Slope coefficient in equation relating treatment to outcome.
rho	A number in [0,1]. Correlation between shocks in equations for R and Y.

Details

The data generating process is of the form:

`R ~ (a_R + b_R*Z > u_R)`

`Y ~ (a_Y + b_Y*Z > u_Y)`

where `u_R` and `u_Y` are joint normally distributed with correlation `rho`.

Value

A post-treatment design.

Author(s)

DeclareDesign Team

Examples

```
# To make a design using default argument (missing completely at random):
two_arm_attrition_design <- two_arm_attrition_designer()
## Not run:
diagnose_design(two_arm_attrition_design)

## End(Not run)
# Attrition can produce bias even for unconditional ATE even when not
# associated with treatment
## Not run:
diagnose_design(two_arm_attrition_designer(b_R = 0, rho = .3))

## End(Not run)
# Here the linear estimate using R=1 data is unbiased for
# "ATE on Y (Given R)" with b_R = 0 but not when  b_R = 1
## Not run:
diagnose_design(redesign(two_arm_attrition_design, b_R = 0:1, rho = .2))

## End(Not run)
```

Index