Package ‘OptimalRerandExpDesigns’

October 12, 2022

Type Package

Title Optimal Rerandomization Experimental Designs

Version 1.1

Date 2021-01-25

Author Adam Kapelner, Michael Sklar, Abba M. Krieger and David Azriel

Maintainer Adam Kapelner <kapelner@qc.cuny.edu>

Description This is a tool to find the optimal rerandomization threshold in non-sequential experiments. We offer three procedures based on assumptions made on the residuals distribution: (1) normality assumed (2) excess kurtosis assumed (3) entire distribution assumed. Illustrations are included. Also included is a routine to unbiasedly estimate Frobenius norms of variance-covariance matrices. Details of the method can be found in "Optimal Rerandomization via a Criterion that Provides Insurance Against Failed Experiments" Adam Kapelner, Abba M. Krieger, Michael Sklar and David Azriel (2020) <arXiv:1905.03337>.

License GPL-3

Depends R (>= 3.2.0), ggplot2 (>= 3.0), momentchi2 (>= 0.1.5), GreedyExperimentalDesign (>= 1.3)

Imports stats

RoxygenNote 7.1.0


NeedsCompilation no

Repository CRAN

Date/Publication 2021-01-28 12:50:06 UTC

R topics documented:

  complete_randomization_plus_one_min_one ............................. 2
  complete_randomization_with_forced_balance_plus_one_min_one .......... 3
  compute_objective_val_plus_one_min_one_enc ............................ 3
  frob_norm_sq .......................................................... 4
complete_randomization_plus_one_min_one

Implements the complete randomization design (CRD) AKA Bernoulli Trial

Description

Implements the complete randomization design (CRD) AKA Bernoulli Trial

Usage

complete_randomization_plus_one_min_one(n, r)

Arguments

n  number of observations

r  number of randomized designs you would like

Value

a matrix where each column is one of the r designs

Author(s)

Adam Kapelner
complete_randomization_with_forced_balance_plus_one_min_one

Implements the balanced complete randomization design (BCRD)

Description
Implements the balanced complete randomization design (BCRD)

Usage
complete_randomization_with_forced_balance_plus_one_min_one(n, r)

Arguments
\n n \quad \text{number of observations}
 r \quad \text{number of randomized designs you would like}

Value
a matrix where each column is one of the \( r \) designs

Author(s)
Adam Kapelner

compute_objective_val_plus_one_min_one_enc

Returns the objective value given a design vector as well an an objective function. This is code duplication since this is implemented within Java. This is only to be run if...

Description
Returns the objective value given a design vector as well an an objective function. This is code duplication since this is implemented within Java. This is only to be run if...

Usage
compute_objective_val_plus_one_min_one_enc(
  X,
  indic_T,
  objective = "abs_sum_diff",
  inv_cov_X = NULL
)
**Arguments**

- **X**: The n x p design matrix
- **indic_T**: The n-length binary allocation vector
- **objective**: The objective function to use. Default is `abs_sum_diff`.
- **inv_cov_X**: Optional: the inverse sample variance covariance matrix. Use this argument if you will be doing many calculations since passing this in will cache this data.

**Value**

A vector of computed objective values.

**Author(s)**

Adam Kapelner

---

**frob_norm_sq**

*Naive Frobenius Norm Squared*

**Description**

Compute naive / vanilla squared Frobenius Norm of matrix A

**Usage**

`frob_norm_sq(A)`

**Arguments**

- **A**: The matrix of interest

**Value**

The Frobenius Norm of A squared.

**Author(s)**

Adam Kapelner
frob_norm_sq_debiased  Debiased Frobenius Norm Squared Var-Cov matrix

Description
Compute debiased Frobenius Norm of matrix Sigmahat (Appendix 5.8). Note that for $S \leq 2$, it returns the naive estimate.

Usage
```r
frob_norm_sq_debiased(
  Sigmahat,
  s,
  n,
  frob_norm_sq_bias_correction_min_samples = 10
)
```

Arguments
- `Sigmahat` The var-cov matrix of interest
- `s` The number of vectors Sigmahat was generated from
- `n` The length of each vector
- `frob_norm_sq_bias_correction_min_samples` This estimate suffers from high variance when there are not enough samples. Thus, we only implement the correction beginning at this number of samples otherwise we return the naive estimate. Default is 10.

Value
The unbiased estimate of the Frobenius Norm of a variance-covariance matrix squared.

Author(s)
Adam Kapelner

frob_norm_sq_debiased_times_matrix

Debiased Frobenius Norm Squared Constant Times Var-Cov matrix

Description
Compute debiased Frobenius Norm of matrix $P$ times Sigmahat (Appendix 5.9). Note that for $S \leq 2$, it returns the naive estimate.
Usage
frob_norm_sq_debiased_times_matrix(
  Sigmahat,
  A,
  s,
  n,
  frob_norm_sq_bias_correction_min_samples = 10
)

Arguments
- Sigmahat: The var-cov matrix of interest
- A: The matrix that multiplies Sigmahat
- s: The number of vectors Sigmahat was generated from
- n: The length of each vector
- frob_norm_sq_bias_correction_min_samples: 
  This estimate suffers from high variance when there are not enough samples. Thus, we only implement the correction beginning at this number of samples otherwise we return the naive estimate. Default is 10.

Value
The unbiased estimate of the Frobenius Norm of A times a variance-covariance matrix quantity squared.

Author(s)
- Adam Kapelner

Description
Generate Base Assignments and Sorts

Usage
generate_W_base_and_sort(
  X,
  max_designs = 25000,
  imbalance_function = "mahal_dist",
  r = 0,
  max_max_iters = 5
)
Arguments

X
- The data as an $n \times p$ matrix.

max_designs
- The maximum number of designs. Default is 25,000.

imbalance_function
- A string indicating the imbalance function. Currently, "abs_sum_difference" and "mahal_dist" are the options with the latter being the default.

r
- An experimental feature that adds lower imbalance vectors to the base set using the GreedyExperimentalDesign package. This controls the number of vectors to search through on each iteration.

max_max_iters
- An experimental feature that adds lower imbalance vectors to the base set using the GreedyExperimentalDesign package. The maximum number of iterations to use for the greedy search.

Value

A list including all arguments plus a matrix $W_{\text{base\_sorted}}$ whose $\text{max\_designs}$ rows are $n$-length allocation vectors and the allocation vectors are in

Author(s)

Adam Kapelner

Examples

n = 100
p = 10
X = matrix(rnorm(n * p), nrow = n, ncol = p)
X = apply(X, 2, function(xj){(xj - mean(xj)) / sd(xj)})
S = 1000

W_base_obj = generate_W_base_and_sort(X, max_designs = S)
W_base_obj

OptimalRerandExpDesigns

Optimal Rerandomization Threshold Search for Experimental Design

Description

A tool to find the optimal rerandomization threshold in non-sequential experiments

Author(s)

Adam Kapelner <kapelner@qc.cuny.edu>

References

Kapelner, A
optimal_rerandomization_exact

Find the Optimal Rerandomization Design Exactly

Description

Finds the optimal rerandomization threshold based on a user-defined quantile and a function that generates the non-linear component of the response

Usage

optimal_rerandomization_exact(
  W_base_object,
  estimator = "linear",
  q = 0.95,
  skip_search_length = 1,
  smoothing_degree = 1,
  smoothing_span = 0.1,
  z_sim_fun,
  N_z = 1000,
  dot_every_x_iters = 100
)

Arguments

W_base_object An object that contains the assignments to begin with sorted by
estimator "linear" for the covariate-adjusted linear regression estimator (default).
q The tail criterion’s quantile of MSE over z’s. The default is 95%.
skip_search_length In the exhaustive search, how many designs are skipped? Default is 1 for full
  exhaustive search through all assignments provided for in W_base_object.
smoothing_degree The smoothing degree passed to loess.
smoothing_span The smoothing span passed to loess.
z_sim_fun This function returns vectors of numeric values of size n. No default is provided.
N_z The number of times to simulate z’s within each strategy.
dot_every_x_iters Print out a dot every this many iterations. The default is 100. Set to NULL for no
  printout.

Value

A list containing the optimal design threshold, strategy, and other information.
optimal_rerandomization_normality_assumed

Author(s)
Adam Kapelner

Examples

```r
n = 100
p = 10
X = matrix(rnorm(n * p), nrow = n, ncol = p)
X = apply(X, 2, function(xj){(xj - mean(xj)) / sd(xj)})
S = 25000

W_base_obj = generate_W_base_and_sort(X, max_designs = S)
design = optimal_rerandomization_exact(W_base_obj, z_sim_fun = function(){rnorm(n)},
skip_search_length = 10)
design
```

---

optimal_rerandomization_normality_assumed

Find the Optimal Rerandomization Design Under the Gaussian Approximation

Description
Finds the optimal rerandomization threshold based on a user-defined quantile and a function that generates the non-linear component of the response

Usage

```r
optimal_rerandomization_normality_assumed(
    W_base_object, 
estimator = "linear", 
    q = 0.95, 
    skip_search_length = 1, 
    dot_every_x_iters = 100
)
```

Arguments

- `W_base_object` An object that contains the assignments to begin with sorted by estimator "linear" for the covariate-adjusted linear regression estimator (default).
- `q` The tail criterion’s quantile of MSE over z’s. The default is 95%.
- `skip_search_length` In the exhaustive search, how many designs are skipped? Default is 1 for full exhaustive search through all assignments provided for in `W_base_object`. 
dot_every_x_iters

Print out a dot every this many iterations. The default is 100. Set to NULL for no printout.

Value

A list containing the optimal design threshold, strategy, and other information.

Author(s)

Adam Kapelner

Examples

n = 100
p = 10
X = matrix(rnorm(n * p), nrow = n, ncol = p)
X = apply(X, 2, function(xj){(xj - mean(xj)) / sd(xj)})
S = 25000

W_base_obj = generate_W_base_and_sort(X, max_designs = S)
design = optimal_rerandomization_normality_assumed(W_base_obj,
skip_search_length = 10)
design
Arguments

- **W_base_object** An object that contains the assignments to begin with sorted by imbalance.
- **estimator** "linear" for the covariate-adjusted linear regression estimator (default).
- **q** The tail criterion's quantile of MSE over z's. The default is 95%.
- **c_val** The c value used (see Equation 8 in the paper). The default is NULL corresponding to \( q \text{norm}(q) \).
- **skip_search_length** In the exhaustive search, how many designs are skipped? Default is 1 for full exhaustive search through all assignments provided for in \( W_\text{base}\_object \).
- **binary_search** If TRUE, a binary search is employed to find the optimal threshold instead of an exhaustive search. Default is FALSE.
- **excess_kurtosis_z** An estimate of the excess kurtosis in the measure on z. Default is 0.
- **use_frob_norm_sq_unbiased_estimator** If TRUE, this would use the debiased Frobenius norm estimator instead of the naive. Default is TRUE.
- **frob_norm_sq_bias_correction_min_samples** The bias-corrected estimate suffers from high variance when there are not enough samples. Thus, we only implement the correction beginning at this number of vectors. Default is 10 and this parameter is only applicable if use_frob_norm_sq_unbiased_estimator is TRUE.
- **smoothing_degree** The smoothing degree passed to \( \text{loess} \).
- **smoothing_span** The smoothing span passed to \( \text{loess} \).
- **dot_every_x_iters** Print out a dot every this many iterations. The default is 100. Set to NULL for no printout.

Value

A list containing the optimal design threshold, strategy, and other information.

Author(s)

Adam Kapelner

Examples

```r
n = 100
p = 10
X = matrix(rnorm(n * p), nrow = n, ncol = p)
```
X = apply(X, 2, function(xj){(xj - mean(xj)) / sd(xj)})
S = 25000

W_base_obj = generate_W_base_and_sort(X, max_designs = S)
design = optimal_rerandomization_tail_approx(W_base_obj,
  skip_search_length = 10)
design

---

plot.optimal_rerandomization_obj

Plots a summary of a optimal_rerandomization_obj object

Description

Plots a summary of a optimal_rerandomization_obj object

Usage

## S3 method for class 'optimal_rerandomization_obj'
plot(x, ...)

Arguments

x The optimal_rerandomization_obj object to be summarized in the plot
...
The option advanced = TRUE can be passed here for optimal rerandomization
results from algorithm type "approx" to see how all the terms in the criterion
behave. You can pass s_min which controls the minimum number of vectors the
plot begins at. Below a certain number, the criterion is unstable. Also, title,
subtitle, xlab and ylab can be passed here.

Value

No return value, called for side effects

Author(s)

Adam Kapelner
**plot.W_base_object**

Plots a summary of the imbalances in a \( W_{\text{base\_object}} \) object

**Description**

Plots a summary of the imbalances in a \( W_{\text{base\_object}} \) object

**Usage**

```r
## S3 method for class 'W_base_object'
plot(x, ...)
```

**Arguments**

- `x`: The \( W_{\text{base\_object}} \) object to be summarized in the plot
- `...`: title, subtitle, xlab, bins can be specified here to be passed to the ggplot plotting function. Also log10 can be set to FALSE to not log the x-axis.

**Value**

No return value, called for side effects

**Author(s)**

Adam Kapelner

**print.optimal_rerandomization_obj**

Prints a summary of a \( \text{optimal\_rerandomization\_obj} \) object

**Description**

Prints a summary of a \( \text{optimal\_rerandomization\_obj} \) object

**Usage**

```r
## S3 method for class 'optimal_rerandomization_obj'
print(x, ...)
```

**Arguments**

- `x`: The \( \text{optimal\_rerandomization\_obj} \) object to be summarized in the console
- `...`: Other parameters to pass to the default print function
Prints a summary of a \texttt{W\_base\_object} object

\textbf{Arguments}

\begin{itemize}
  \item \texttt{x} \hspace{1cm} The \texttt{W\_base\_object} object to be summarized in the console
  \item \texttt{...} \hspace{1cm} Other parameters to pass to the default print function
\end{itemize}

\textbf{Value}

No return value, called for side effects

\textbf{Author(s)}

Adam Kapelner

---

\textbf{summary.optimal\_rerandomization\_obj}

Prints a summary of a \texttt{optimal\_rerandomization\_obj} object

\textbf{Description}

Prints a summary of a \texttt{optimal\_rerandomization\_obj} object

\textbf{Usage}

\begin{verbatim}
## S3 method for class 'optimal\_rerandomization\_obj'
summary(object, ...)
\end{verbatim}
**summary.W_base_object**

**Arguments**

- **object**
  - The `optimal_rerandomization_obj` object to be summarized in the console
- ... Other parameters to pass to the default summary function

**Author(s)**

Adam Kapelner

---

**summary.W_base_object**  *Prints a summary of a W_base_object object*

---

**Description**

Prints a summary of a `W_base_object` object

**Usage**

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

**Arguments**

- **object**
  - The `W_base_object` object to be summarized in the console
- ... Other parameters to pass to the default summary function

**Author(s)**

Adam Kapelner
Index

complete_randomization_plus_one_min_one, 2
complete_randomization_with_forced_balance_plus_one_min_one, 3
compute_objective_val_plus_one_min_one_enc, 3

frob_norm_sq, 4
frob_norm_sq_debiased, 5
frob_norm_sq_debiased_times_matrix, 5

generate_W_base_and_sort, 6

optimal_rerandomization_exact, 8
optimal_rerandomization_normality_assumed, 9
optimal_rerandomization_tail_approx, 10
OptimalRerandExpDesigns, 7

plot.optimal_rerandomization_obj, 12
plot.W_base_object, 13
print.optimal_rerandomization_obj, 13
print.W_base_object, 14

summary.optimal_rerandomization_obj, 14
summary.W_base_object, 15