Package ‘GreedyExperimentalDesign’

December 7, 2020

Type Package

Title Greedy Experimental Design Construction

Version 1.4

Date 2020-12-07

Author Adam Kapelner, David Azriel and Abba M. Krieger

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

Description Computes experimental designs for a
two-arm experiment with covariates via a number of methods.
(0) complete randomization and randomization with forced-balance.
(1) Greedily optimizing a
balance objective function via pairwise switching. This optimization
provides lower variance for the treatment effect estimator (and higher
power) while preserving a design that is close to complete randomization.
We return all iterations of the designs for use in a permutation test.
(2) The second is via numerical optimization
(via 'gurobi' which must be installed, see <https://www.gurobi.com/documentation/9.1/quickstart_windows/r_ins_the_r_package.html>)
a la Bertsimas and Kallus.
(3) rerandomization,
(4) Karp's method for one covariate,
(5) exhaustive enumeration to find the
optimal solution (only for small sample sizes)
(6) Binary pair matching using the 'nbpMatching' library
(7) Binary pair matching plus (1) to further optimize balance
(8) Binary pair matching plus (3) to further optimize balance
(9) Hadamard designs
We also allow for three objective functions:
Mahalanobis distance,
Sum of absolute differences standardized and
Kernel distances via the 'kernlab' library.

License GPL-3

Depends R (>= 3.2.0), rJava (>= 0.9-6), GreedyExperimentalDesignJARs
(>= 1.0)

SystemRequirements Java (>= 7.0)

LinkingTo Rcpp

Imports Rcpp, checkmate, nbpMatching, survey, kernlab, graphics, grDevices, stats

RoxygenNote 7.1.0

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-12-07 16:30:06 UTC

R topics documented:

automobile *Data concerning automobile prices.*

Description

The automobile data frame has 201 rows and 25 columns and concerns automobiles in the 1985 Auto Imports Database. The response variable, price, is the log selling price of the automobile. There are 7 categorical predictors and 17 continuous / integer predictors which are features of the automobiles. 41 automobiles have missing data in one or more of the feature entries. This dataset is true to the original except with a few of the predictors dropped.

Usage

```
data(automobile)
```

Source

K Bache and M Lichman. UCI machine learning repository, 2013. http://archive.ics.uci.edu/ml/datasets/Automobile

binaryMatchExperimentalDesignSearch *Begin a Search for Binary Matching Designs*

Description

This method creates an object of type binary_experimental_design and will find pairs. You can then use the function resultsBinaryMatchSearch to create randomized allocation vectors. For one column in X, we just sort to find the pairs trivially.

Usage

```
binaryMatchExperimentalDesignSearch(X, compute_dist_matrix = NULL)
```

Arguments

`X` The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design.

`compute_dist_matrix` The function that computes the distance matrix between every two observations in `X`, its only argument. The default is `NULL` signifying euclidean squared distance optimized in C++.

Value

An object of type `binary_experimental_design` which can be further operated upon.

Author(s)

Adam Kapelner

---

binaryMatchFollowedByGreedyExperimentalDesignSearch

Begin a Search for Binary Matching Followed by Greedy Switch Designs

---

Description

This method creates an object of type binary_then_greedy_experimental_design and will find optimal matched pairs which are then greedily switched in order to further minimize a balance metric. You can then use the function `resultsBinaryMatchThenGreedySearch` to obtain the randomized allocation vectors. For one column in X, the matching just sorts the values to find the pairs trivially.

Usage

```
binaryMatchFollowedByGreedyExperimentalDesignSearch(
  X,
  compute_dist_matrix = NULL,
  ...
)
```

Arguments

`X` The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design.

`compute_dist_matrix` The function that computes the distance matrix between every two observations in `X`, its only argument. The default is `NULL` signifying euclidean squared distance optimized in C++.

`...` Arguments passed to `initGreedyExperimentalDesignObject`. It is recommended to set `max_designs` otherwise it will default to 10,000.

Value

An object of type `binary_experimental_design` which can be further operated upon.

Author(s)

Adam Kapelner

`binaryMatchFollowedByRerandomizationDesignSearch` *Begin a Search for Binary Matching Followed by Rerandomization*

Description

This method creates an object of type binary_then_rerandomization_experimental_design and will find optimal matched pairs which are then rerandomized in order to further minimize a balance metric. You can then use the function `resultsBinaryMatchThenRerandomizationSearch` to obtain the randomized allocation vectors. For one column in X, the matching just sorts the values to find the pairs trivially.

Usage

```
binaryMatchFollowedByRerandomizationDesignSearch(
  X,
  compute_dist_matrix = NULL,
  ...
)
```

Arguments

`X`	The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design.
`compute_dist_matrix`	The function that computes the distance matrix between every two observations in `X`, its only argument. The default is `NULL` signifying euclidean squared distance optimized in C++.
`...`	Arguments passed to `initGreedyExperimentalDesignObject`. It is recommended to set `max_designs` otherwise it will default to 10,000.

Value

An object of type `binary_experimental_design` which can be further operated upon.

Author(s)

Adam Kapelner

complete_randomization *Implements complete randomization (without forced balance)*

Description

Implements complete randomization (without forced balance)

Usage

```
complete_randomization(n, r, form = "one_zero")
```

Arguments

n	number of observations
r	number of randomized designs you would like
form	Which form should it be in? The default is one_zero for 1/0's or pos_one_min_one for +1/-1's.

Value

a matrix where each column is one of the r designs

Author(s)

Adam Kapelner

complete_randomization_with_forced_balanced *Implements forced balanced randomization*

Description

Implements forced balanced randomization

Usage

```
complete_randomization_with_forced_balanced(n, r, form = "one_zero")
```

Arguments

n	number of observations
r	number of randomized designs you would like
form	Which form should it be in? The default is one_zero for 1/0's or pos_one_min_one for +1/-1's.

Value

a matrix where each column is one of the `r` designs

Author(s)

Adam Kapelner

`compute_gram_matrix` *Gram Matrix Computation*

Description

Computes the Gram Matrix for a user-specified kernel using the library `kernlab`. Note that this function automatically standardizes the columns of the data entered.

Usage

```
compute_gram_matrix(X, kernel_type, params = c())
```

Arguments

`X`	The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design.
`kernel_type`	One of the following: "vanilla", "rbf", "poly", "tanh", "bessel", "laplace", "anova" or "spline".
`params`	A vector of numeric parameters. Each `kernel_type` has different numbers of parameters required. For more information see documentation for the `kernlab` library.

Value

The `n x n` gram matrix for the given kernel on the given data.

Author(s)

Adam Kapelner

compute_objective_val *Computes Objective Value From Allocation Vector*

Description

Returns the objective value given a design vector as well an an objective function. This is sometimes duplicated in Java. However, within Java, tricks are played to make optimization go faster so Java's objective values may not always be the same as the true objective function (e.g. logs or constants dropped).

Usage

```
compute_objective_val(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` and the other option is `mahal_dist`.
`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.

Author(s)

Adam Kapelner

compute_randomization_metrics *Computes Randomization Metrics (explained in paper) about a design algorithm*

Description

Computes Randomization Metrics (explained in paper) about a design algorithm

Usage

```
compute_randomization_metrics(designs)
```

Arguments

`designs`	A matrix where each column is one design.

Value

A list of resulting data: the probability estimates for each pair in the design of randomness where estmates close to ~0.5 represent random assignment, then the entropy metric the distance metric, the maximum eigenvalue of the allocation var-cov matrix (operator norm) and the squared Frobenius norm (the sum of the squared eigenvalues)

Author(s)

Adam Kapelner

---

generate_stdzied_design_matrix

Generates a design matrix with standardized predictors.

Description

This function is useful for debugging.

Usage

```
generate_stdzied_design_matrix(n = 50, p = 1, covariate_gen = rnorm, ...)
```

Arguments

n	Number of rows in the design matrix
p	Number of columns in the design matrix
covariate_gen	The function to use to draw the covariate realizations (assumed to be iid). This defaults to `rnorm` for $N(0,1)$ draws.
...	Optional arguments to be passed to the `covariate_dist` function.

Value

THe design matrix

Author(s)

Adam Kapelner

GreedyExperimentalDesign *Greedy Experimental Design Search*

Description

A tool to find a priori experimental designs with good balance greedily

Author(s)

Adam Kapelner <kapelner@qc.cuny.edu>

References

Kapelner, A

greedy_orthogonalization_curation *Curate More Orthogonal Vectors Greedily*

Description

This function takes a set of allocation vectors and pares them down one-by-one by eliminating the vector that can result in the largest reduction in Avg[|r_ij|]. It is recommended to begin with a set of unmirrored vectors for speed. Then add the mirrors later for whichever subset you wish.

Usage

```
greedy_orthogonalization_curation(W, Rmin = 2, verbose = FALSE)
```

Arguments

`W`	A matrix in $-1, 1^R x n$ which have R allocation vectors for an experiment of sample size n.
`Rmin`	The minimum number of vectors to consider in a design. The default is the true bottom, two.
`verbose`	Default is `FALSE` but if not, it will print out a message for each iteration.

Value

A list with two elements: (1) `avg_abs_rij_by_R` which is a data frame with R - Rmin + 1 rows and columns R and average absolute r_ij and (2) `Wsorted` which provides the collection of vectors in sorted by best average absolute r_ij in row order from best to worst.

Author(s)

Adam Kapelner

greedy_orthogonalization_curation2
Curate More Orthogonal Vectors Greedily

Description

This function takes a set of allocation vectors and pares them down one-by-one by eliminating the vector that can result in the largest reduction in Avg[|r_ij|]. It is recommended to begin with a set of unmirrored vectors for speed. Then add the mirrors later for whichever subset you wish.

Usage

```
greedy_orthogonalization_curation2(W, R0 = 100, verbose = FALSE)
```

Arguments

`W`	A matrix in $-1, 1^R x n$ which have R allocation vectors for an experiment of sample size n.
`R0`	The minimum number of vectors to consider in a design. The default is the true bottom, two.
`verbose`	Default is `FALSE` but if not, it will print out a message for each iteration.

Value

A list with two elements: (1) `avg_abs_rij_by_R` which is a data frame with R - Rmin + 1 rows and columns R and average absolute r_ij and (2) `Wsorted` which provides the collection of vectors in sorted by best average absolute r_ij in row order from best to worst.

Author(s)

Adam Kapelner

hadamardExperimentalDesign
Create a Hadamard Design

Description

This method returns unique designs according to a Hadamard matrix

Usage

```
hadamardExperimentalDesign(X, strict = TRUE, form = "zero_one")
```

Arguments

`X` The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). The measurements aren't used to compute the Hadamard designs, only the number of rows.

`strict` Hadamard matrices are not available for all $n$.

`form` Which form should it be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Value

An matrix of dimension $R$ x $n$ where $R$ is the number of Hadamard allocations.

Author(s)

Adam Kapelner

`initGreedyExperimentalDesignObject` *Begin A Greedy Pair Switching Search*

Description

This method creates an object of type greedy_experimental_design and will immediately initiate a search through $1_T$ space for forced balance designs.

Usage

```
initGreedyExperimentalDesignObject(
  X = NULL,
  max_designs = 10000,
  objective = "mahal_dist",
  Kgram = NULL,
  wait = FALSE,
  start = TRUE,
  max_iters = Inf,
  semigreedy = FALSE,
  diagnostics = FALSE,
  num_cores = 1
)
```

Arguments

`X` The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design. This parameter must be specified unless you choose objective type `"kernel"` in which case, the `Kgram` parameter must be specified.

max_designs The maximum number of designs to be returned. Default is 10,000. Make this large so you can search however long you wish as the search can be stopped at any time by using the stopSearch method

objective The objective function to use when searching design space. This is a string with valid values "mahal_dist" (the default), "abs_sum_diff" or "kernel".

Kgram If the objective = kernel, this argument is required to be an n x n matrix whose entries are the evaluation of the kernel function between subject i and subject j. Default is NULL.

wait Should the R terminal hang until all max_designs vectors are found? The deafult is FALSE.

start Should we start searching immediately (default is TRUE).

max_iters Should we impose a maximum number of greedy switches? The default is Inf which a flag for “no limit.”

semigreedy Should we use a fully greedy approach or the quicker semi-greedy approach? The default is FALSE corresponding to the fully greedy approach.

diagnostics Returns diagnostic information about the iterations including (a) the initial starting vectors, the switches at every iteration and information about the objective function at every iteration (default is FALSE due to speed concerns).

num_cores The number of CPU cores you wish to use during the search. The default is 1.

Value

An object of type greedy_experimental_design_search which can be further operated upon

Author(s)

Adam Kapelner

Examples

```
 ## Not run:
library(MASS)
data(Boston)
 #pretend the Boston data was an experiment setting
#first pull out the covariates
 X = Boston[, 1 : 13]
 #begin the greedy design search
ged = initGreedyExperimentalDesignObject(X,
max_designs = 1000, num_cores = 3, objective = "abs_sum_diff")
#wait
ged

## End(Not run)
```

initKarpExperimentalDesignObject

Begin Karp Search

Description

This method creates an object of type karp_experimental_design and will immediately initiate a search through $1_T$ space. Note that the Karp search only works for one covariate (i.e. $p=1$) and the objective "abs_sum_diff".

Usage

```
initKarpExperimentalDesignObject(
  X,
  wait = FALSE,
  balanced = TRUE,
  start = TRUE
)
```

Arguments

`X`	The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more karp design.
`wait`	Should the `R` terminal hang until all `max_designs` vectors are found? The deafult is `FALSE`.
`balanced`	Should the final vector be balanced? Default and recommended is `TRUE`.
`start`	Should we start searching immediately (default is `TRUE`).

Value

An object of type `karp_experimental_design_search` which can be further operated upon

Author(s)

Adam Kapelner

initOptimalExperimentalDesignObject *Begin a Search for the Optimal Solution*

Description

This method creates an object of type optimal_experimental_design and will immediately initiate a search through $1_T$ space. Since this search takes exponential time, for most machines, this method is futile beyond 28 samples. You've been warned!

Usage

```
initOptimalExperimentalDesignObject(
  X = NULL,
  objective = "mahal_dist",
  Kgram = NULL,
  wait = FALSE,
  start = TRUE,
  num_cores = 1
)
```

Arguments

`X`	The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design.
`objective`	The objective function to use when searching design space. This is a string with valid values "`mahal_dist`" (the default), "`abs_sum_diff`" or "`kernel`".
`Kgram`	If the `objective` = `kernel`, this argument is required to be an `n x n` matrix whose entries are the evaluation of the kernel function between subject i and subject j. Default is `NULL`.
`wait`	Should the `R` terminal hang until all `max_designs` vectors are found? The deafult is `FALSE`.
`start`	Should we start searching immediately (default is `TRUE`).
`num_cores`	The number of CPU cores you wish to use during the search. The default is `1`.

Value

An object of type `optimal_experimental_design_search` which can be further operated upon

Author(s)

Adam Kapelner

`initRerandomizationExperimentalDesignObject` *Begin a Rerandomization Search*

Description

This method creates an object of type rerandomization_experimental_design and will immediately initiate a search through $1_T$ space for forced-balance designs.

Usage

```
initRerandomizationExperimentalDesignObject(
  X = NULL,
  obj_val_cutoff_to_include,
  max_designs = 1000,
  objective = "mahal_dist",
  Kgram = NULL,
  wait = FALSE,
  start = TRUE,
  num_cores = 1
)
```

Arguments

`X` The design matrix with $n$ rows (one for each subject) and $p$ columns (one for each measurement on the subject). This is the design matrix you wish to search for a more optimal design.

`obj_val_cutoff_to_include` Only allocation vectors with objective values lower than this threshold will be returned. If the cutoff is infinity, you are doing BCRD and you should use the `complete_randomization_with_forced_balanced` function instead.

`max_designs` The maximum number of designs to be returned. Default is 10,000. Make this large so you can search however long you wish as the search can be stopped at any time by using the `stopSearch` method

`objective` The objective function to use when searching design space. This is a string with valid values "`mahal_dist`" (the default), "`abs_sum_diff`" or "`kernel`".

`Kgram` If the `objective = kernel`, this argument is required to be an `n x n` matrix whose entries are the evaluation of the kernel function between subject i and subject j. Default is `NULL`.

`wait` Should the `R` terminal hang until all `max_designs` vectors are found? The default is `FALSE`.

`start` Should we start searching immediately (default is `TRUE`).

`num_cores` The number of CPU cores you wish to use during the search. The default is `1`.

Value

An object of type `rerandomization_experimental_design_search` which can be further operated upon.

Author(s)

Adam Kapelner

`plot.greedy_experimental_design_search` *Plots a summary of a* `greedy_experimental_design_search` *object*

Description

Plots a summary of a `greedy_experimental_design_search` object

Usage

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

Arguments

`x`	The `greedy_experimental_design_search` object to be summarized in the plot
`...`	Other parameters to pass to the default plot function

Value

An array of order statistics from plot_obj_val_order_statistic as a list element

Author(s)

Adam Kapelner

plot_obj_val_by_iter *Plots the objective value by iteration*

Description

Plots the objective value by iteration

Usage

```
plot_obj_val_by_iter(res, runs = NULL)
```

Arguments

res	Results from a greedy search object
runs	A vector of run indices you would like to see plotted (default is to plot the first up to 9)

Author(s)

Adam Kapelner

plot_obj_val_order_statistic *Plots an order statistic of the object value as a function of number of searches*

Description

Plots an order statistic of the object value as a function of number of searches

Usage

```
plot_obj_val_order_statistic(
  obj,
  order_stat = 1,
  skip_every = 5,
  type = "o",
  ...
)
```

Arguments

obj	The greedy_experimental_design_search object whose search history is to be visualized
order_stat	The order statistic that you wish to plot. The default is 1 for the minimum.
skip_every	Plot every nth point. This makes the plot generate much more quickly. The default is 5.
type	The type parameter for plot.
...	Other arguments to be passed to the plot function.

Value

An array of order statistics as a list element

Author(s)

Adam Kapelner

---

print.binary_experimental_design *Prints a summary of a* binary_experimental_design *object*

Description

Prints a summary of a binary_experimental_design object

Usage

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

Arguments

x	The binary_experimental_design object to be summarized in the console
...	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

print.binary_then_greedy_experimental_design

Prints a summary of a binary_then_greedy_experimental_design *object*

Description

Prints a summary of a binary_then_greedy_experimental_design object

Usage

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

Arguments

x	The binary_then_greedy_experimental_design object to be summarized in the console
...	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

print.binary_then_rerandomization_experimental_design

Prints a summary of a binary_then_rerandomization_experimental_design *object*

Description

Prints a summary of a binary_then_rerandomization_experimental_design object

Usage

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

Arguments

x	The binary_then_rerandomization_experimental_design object to be summarized in the console
...	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

print.greedy_experimental_design_search

Prints a summary of a `greedy_experimental_design_search` *object*

Description

Prints a summary of a `greedy_experimental_design_search` object

Usage

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

Arguments

`x`	The `greedy_experimental_design_search` object to be summarized in the console
`...`	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

print.karp_experimental_design_search

Prints a summary of a `karp_experimental_design_search` *object*

Description

Prints a summary of a `karp_experimental_design_search` object

Usage

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

Arguments

`x`	The `karp_experimental_design_search` object to be summarized in the console
`...`	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

print.optimal_experimental_design_search

Prints a summary of a `optimal_experimental_design_search` *object*

Description

Prints a summary of a `optimal_experimental_design_search` object

Usage

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

Arguments

`x`	The `optimal_experimental_design_search` object to be summarized in the console
`...`	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

print.rerandomization_experimental_design_search

Prints a summary of a `rerandomization_experimental_design_search` *object*

Description

Prints a summary of a `rerandomization_experimental_design_search` object

Usage

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

Arguments

`x`	The `rerandomization_experimental_design_search` object to be summarized in the console
`...`	Other parameters to pass to the default print function

Author(s)

Adam Kapelner

resultsBinaryMatchSearch

Returns unique allocation vectors that are binary matched

Description

Returns unique allocation vectors that are binary matched

Usage

```
resultsBinaryMatchSearch(
  obj,
  num_vectors = 1000,
  objective = NULL,
  form = "zero_one"
)
```

Arguments

`obj`	The `binary_experimental_design` object where the pairs are computed.
`num_vectors`	How many random allocation vectors you wish to return. The default is 1000.
`objective`	Should we compute all the objective values for each allocation? Default is `NULL` for "no". If non-null, it needs to either be "mahal_dist" or "abs_sum_diff".
`form`	Which form should it be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Author(s)

Adam Kapelner

resultsBinaryMatchThenGreedySearch

Returns unique allocation vectors that are binary matched

Description

Returns unique allocation vectors that are binary matched

Usage

```
resultsBinaryMatchThenGreedySearch(
  obj,
  num_vectors = NULL,
  compute_obj_vals = FALSE,
  form = "zero_one"
)
```

Arguments

- `obj` The `binary_then_greedy_experimental_design` object where the pairs are computed.
- `num_vectors` How many random allocation vectors you wish to return. The default is `NULL` indicating you want all of them.
- `compute_obj_vals` Should we compute all the objective values for each allocation? Default is `FALSE`.
- `form` Which form should it be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Author(s)

Adam Kapelner

resultsBinaryMatchThenRerandomizationSearch

Returns unique allocation vectors that are binary matched

Description

Returns unique allocation vectors that are binary matched

Usage

```
resultsBinaryMatchThenRerandomizationSearch(
  obj,
  num_vectors = NULL,
  compute_obj_vals = FALSE,
  form = "zero_one"
)
```

Arguments

- `obj` The `binary_then_greedy_experimental_design` object where the pairs are computed.
- `num_vectors` How many random allocation vectors you wish to return. The default is `NULL` indicating you want all of them.
- `compute_obj_vals` Should we compute all the objective values for each allocation? Default is `FALSE`.
- `form` Which form should it be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Author(s)

Adam Kapelner

`resultsGreedySearch` *Returns the results (thus far) of the greedy design search*

Description

Returns the results (thus far) of the greedy design search

Usage

```
resultsGreedySearch(obj, max_vectors = 9, form = "one_zero")
```

Arguments

`obj` The `greedy_experimental_design` object that is currently running the search

`max_vectors` The number of design vectors you wish to return. `NULL` returns all of them. This is not recommended as returning over 1,000 vectors is time-intensive. The default is 9.

`form` Which form should it be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Author(s)

Adam Kapelner

Examples

```
 ## Not run:
library(MASS)
data(Boston)
 #pretend the Boston data was an experiment setting
#first pull out the covariates
 X = Boston[, 1 : 13]
 #begin the greedy design search
ged = initGreedyExperimentalDesignObject(X,
max_designs = 1000, num_cores = 2, objective = "abs_sum_diff")
#wait
res = resultsGreedySearch(ged, max_vectors = 2)
design = res$ending_indicTs[, 1] #ordered already by best-->worst
 design
 #what is the balance on this vector?
res$obj_vals[1]
#compute balance explicitly in R to double check
compute_objective_val(X, design) #same as above
#how far have we come?
ged
#we can cut it here
stopSearch(ged)

## End(Not run)
```

`resultsKarpSearch` *Returns the results (thus far) of the karp design search*

Description

Returns the results (thus far) of the karp design search

Usage

```
resultsKarpSearch(obj)
```

Arguments

`obj`	The `karp_experimental_design` object that is currently running the search

Author(s)

Adam Kapelner

`resultsOptimalSearch` *Returns the results (thus far) of the optimal design search*

Description

Returns the results (thus far) of the optimal design search

Usage

```
resultsOptimalSearch(obj, num_vectors = 1, form = "one_zero")
```

Arguments

`obj`	The `optimal_experimental_design` object that is currently running the search
`num_vectors`	How many allocation vectors you wish to return. The default is 1 meaning the best vector. If `Inf`, it means all vectors.
`form`	Which form should it be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Author(s)

Adam Kapelner

resultsRerandomizationSearch *Returns the results (thus far) of the rerandomization design search*

Description

Returns the results (thus far) of the rerandomization design search

Usage

```
resultsRerandomizationSearch(
  obj,
  include_assignments = FALSE,
  form = "one_zero"
)
```

Arguments

`obj` The `rerandomization_experimental_design` object that is currently running the search

`include_assignments` Do we include the assignments (takes time) and default is `FALSE`.

`form` Which form should the assignments be in? The default is `one_zero` for 1/0's or `pos_one_min_one` for +1/-1's.

Author(s)

Adam Kapelner

searchTimeElapsed *Returns the amount of time elapsed*

Description

Returns the amount of time elapsed

Usage

```
searchTimeElapsed(obj)
```

Arguments

`obj` The `experimental_design` object that is currently running the search

Author(s)

Adam Kapelner

standardize_data_matrix *Standardizes the columns of a data matrix.*

Description

Standardizes the columns of a data matrix.

Usage

```
standardize_data_matrix(X)
```

Arguments

`X`	The n x p design matrix

Value

The n x p design matrix with columns standardized

Author(s)

Adam Kapelner

startSearch *Starts the parallelized greedy design search.*

Description

Once begun, this function cannot be run again.

Usage

```
startSearch(obj)
```

Arguments

`obj`	The `experimental_design` object that will be running the search

Author(s)

Adam Kapelner

stopSearch *Stops the parallelized greedy design search.*

Description

Once stopped, it cannot be restarted.

Usage

```
stopSearch(obj)
```

Arguments

`obj`	The `experimental_design` object that is currently running the search

Author(s)

Adam Kapelner

summary.binary_experimental_design *Prints a summary of a `binary_experimental_design` object*

Description

Prints a summary of a `binary_experimental_design` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

summary.binary_then_greedy_experimental_design

Prints a summary of a `binary_then_greedy_experimental_design` *object*

Description

Prints a summary of a `binary_then_greedy_experimental_design` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

summary.binary_then_rerandomization_experimental_design

Prints a summary of a `binary_then_rerandomization_experimental_design` *object*

Description

Prints a summary of a `binary_then_rerandomization_experimental_design` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

`summary.greedy_experimental_design_search` *Prints a summary of a* `greedy_experimental_design_search` *object*

Description

Prints a summary of a `greedy_experimental_design_search` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

`summary.karp_experimental_design_search` *Prints a summary of a* `karp_experimental_design_search` *object*

Description

Prints a summary of a `karp_experimental_design_search` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

summary.optimal_experimental_design_search

Prints a summary of a `optimal_experimental_design_search` *object*

Description

Prints a summary of a `optimal_experimental_design_search` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

summary.rerandomization_experimental_design_search

Prints a summary of a `rerandomization_experimental_design_search` *object*

Description

Prints a summary of a `rerandomization_experimental_design_search` object

Usage

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

Arguments

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

Author(s)

Adam Kapelner

Index