Package 'BrainEnrich'

July 31, 2024

Title Revealing Biological Insights from Imaging-Derived Features through Transcriptomic Enrichment

Version 1.0

Date 2024-05-24

Description What the package does (one paragraph).

License AGPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Suggests knitr,
rmarkdown

Config/testthat/edition 3

Imports DOSE,
clue,
dplyr,
matrixStats,
parallel,
pbapply,
purrr,
tibble,
tidyr,
broom,
ggplot2,
ggseg,
parameters,
pls,
BiocManager,
methods,
stats,
data.table,
utils

Depends R (>= 4.0.0)

LazyData true

URL https://zh1peng.github.io/BrainEnrich/

Contents

aggregate_geneSet *Aggregate Gene Set Scores*

Description

Function to aggregate geneList based on geneSet, evaluating one geneSet at a time. The function supports multiple aggregation methods as specified by the user.

Usage

```
aggregate_geneSet(
  geneList,
  geneSet,
 method = c("mean", "median", "meanabs", "meansqr", "maxmean", "ks_orig", "ks_weighted",
    "ks_pos_neg_sum", "sign_test", "rank_sum", "custom")
)
```

Arguments

`geneList`	A matrix of genes by models, with each column representing a true or null model.
`geneSet`	A vector containing names of genes in the gene set of interest.
`method`	A character string specifying the method to use for aggregation. Options include 'mean', 'median', 'meanabs', 'meansqr', 'maxmean', 'sig_n', 'sign_test', 'rank_sum', 'ks_orig', 'ks_weighted', 'ks_sum'. Default is 'mean'.

Value

Returns a numeric score based on the specified aggregation method.

`aggregate_geneSetList` *Aggregate Gene Set List in Parallel*

Description

This function aggregates gene sets in parallel using the `parLapply` function from the `parallel` package, ensuring cross-platform compatibility.

Usage

```
aggregate_geneSetList(geneList, geneSetList, method, n_cores = 1)
```

Arguments

`geneList`	A list of genes.
`geneSetList`	A list of gene sets to be aggregated.
`method`	aggregation method used.
`n_cores`	Number of cores to use for parallel processing. Default is 1.

Value

A list of aggregated gene set scores.

`aggregate_geneSetList_matching_coexp` *Aggregate Gene Set List with Matching Coexpression in Parallel*

Description

This function aggregates gene set scores in parallel using `pblapply` from the `pbapply` package.

Usage

```
aggregate_geneSetList_matching_coexp(
  geneList.true,
  geneSetList,
  sampled_geneSetList,
  method,
  n_cores = 1
)
```

Arguments

`geneList.true`	A m x 1 matrix of true gene sets. Ensure to include `drop=FALSE` when subsetting.
`geneSetList`	A list of gene sets.
`sampled_geneSetList`	A list of sampled gene sets.
`method`	The method to be used for aggregation.
`n_cores`	Number of cores to use for parallel processing. Default is 1. If set to 0, it uses all available cores minus one.

Value

A list of aggregated gene set scores.

`brainenrich` *Perform Brain Gene Set Analysis*

Description

This function performs a gene set analysis using brain data.

Usage

```
brainenrich(
  brain_data,
  gene_data,
  annoData,
  cor_method = c("pearson", "spearman", "pls1c", "pls1w", "custom"),
 aggre_method = c("mean", "median", "meanabs", "meansqr", "maxmean", "ks_orig",
  "ks_weighted", "ks_pos_neg_sum", "local_fdr", "sign_test", "rank_sum", "custom"),
  null_model = c("spin_brain", "resample_gene", "coexp_matched"),
  minGSSize = 10,
  maxGSSize = 200,
  n_cores = 0,
  n_perm = 5000,
  perm_id = NULL,
  coord.l = NULL,
  coord.r = NULL,
  seed = NULL,
  threshold_type = c("sd", "percentile", "none"),
  threshold_val = 1,
  pvalueCutoff = 0.05,
  pAdjustMethod = "fdr",
  matchcoexp_tol = 0.05,
  matchcoexp_max_iter = 1e+06
)
```

Arguments

brain_data — A data frame of brain data. Region by 1 column.

gene_data — A data frame of gene expression data.

annoData — An environment containing annotation data.

cor_method — A character string specifying the correlation method. Default is 'pearson'. Other options include 'spearman', 'pls1c', 'pls1w', 'custom'.

aggre_method — A character string specifying the aggregation method. Default is 'mean'. Other options include 'median', 'meanabs', 'meansqr', 'maxmean', 'ks_orig', 'ks_weighted', 'ks_pos_neg_sum', 'local_fdr', 'sign_test', 'rank_sum', 'custom'.

null_model — A character string specifying the null model. Default is 'spin_brain'. Other options include 'resample_gene', 'coexp_matched'.

minGSSize — An integer specifying the minimum gene set size. Default is 10.

maxGSSize — An integer specifying the maximum gene set size. Default is 200.

n_cores — An integer specifying the number of cores to use. Default is 0.

n_perm — An integer specifying the number of permutations. Default is 5000.

perm_id — A matrix of permutation IDs. Default is NULL.

coord.l — A matrix of left hemisphere coordinates. Default is NULL.

coord.r — A matrix of right hemisphere coordinates. Default is NULL.

seed — An integer specifying the seed for reproducibility of spinning brain. Default is NULL.

threshold_type — A character string specifying the threshold type for core genes. Default is 'sd'. Other option is 'percentile'.

threshold_val — A numeric value specifying the threshold value for core genes. Default is 1.

pvalueCutoff — A numeric value specifying the p-value cutoff for output. Default is 0.05.

pAdjustMethod — A character string specifying the method for p-value adjustment. Default is 'fdr'.

matchcoexp_tol — A numeric value specifying the tolerance for matched co-expression. Lower value means better matching but will take much more iterations. Default is 0.05.

matchcoexp_max_iter — An integer specifying the maximum number of iterations for matched co-expression. Default is 1000000.

Value

A gseaResult object containing the enrichment results.

brainscore *Calculate Brain Scores for Gene Sets*

Description

This function calculates scores for gene sets based on brain data. It includes options for different null models.

Usage

```
brainscore(
  brain_data,
  gene_data,
  annoData,
  cor_method = c("pearson", "spearman", "pls1c", "pls1w", "custom"),
  aggre_method = c("mean", "median", "meanabs", "meansqr", "maxmean", "ks_orig",
    "ks_weighted", "ks_pos_neg_sum", "sign_test", "rank_sum", "custom"),
  null_model = c("none", "spin_brain", "resample_gene", "coexp_matched"),
  minGSSize = 10,
  maxGSSize = 200,
  n_cores = 0,
  n_perm = 5000,
  perm_id = NULL,
  coord.l = NULL,
  coord.r = NULL,
  seed = NULL,
  matchcoexp_tol = 0.05,
  matchcoexp_max_iter = 1e+06
)
```

Arguments

`brain_data` A data frame of brain data. Region by 1 column.

`gene_data` A data frame of gene expression data.

`annoData` An environment containing annotation data.

`cor_method` A character string specifying the correlation method. Default is 'pearson'. Other options include 'spearman', 'pls1c', 'pls1w', 'custom'.

`aggre_method` A character string specifying the aggregation method. Default is 'mean'. Other options include 'median', 'meanabs', 'meansqr', 'maxmean', 'ks_orig', 'ks_weighted', 'ks_pos_neg_sum', 'sign_test', 'rank_sum', 'custom'.

`null_model` A character string specifying the null model. Default is 'none'. Other options include 'spin_brain', 'resample_gene', 'coexp_matched'.

`minGSSize` An integer specifying the minimum gene set size. Default is 10.

`maxGSSize` An integer specifying the maximum gene set size. Default is 200.

`n_cores` An integer specifying the number of cores to use for parallel processing. Default is 0 (no parallel processing).

`n_perm` An integer specifying the number of permutations for null models. Default is 5000.

`perm_id` A matrix of permutation indices for ’spin_brain’ null model. Default is NULL. Either perm_id or any of coord.l or coord.r must be provided if choosing spin_brain mode.

`coord.l` A matrix of coordinates for the left hemisphere for ’spin_brain’ null model. Default is NULL.

`coord.r` A matrix of coordinates for the right hemisphere for ’spin_brain’ null model. Default is NULL.

`seed` An integer specifying the seed for reproducibility of spinning brain. Default is NULL.

`matchcoexp_tol` A numeric value specifying the tolerance for matching co-expression in ’coexp_matched’ null model. Default is 0.05.

`matchcoexp_max_iter`
An integer specifying the maximum iterations for matching co-expression in ’coexp_matched’ null model. Default is 1000000.

Value

A data frame containing the gene set scores with regions as rows and gene sets as columns.

brainscore.lm_test *Perform Brain Score Linear Model Test*

Description

This function performs a linear model test on brain score data with the option to use various null models for comparison. It calculates gene set scores, performs linear modeling, calculates p-values, and identifies core genes.

Usage

```
brainscore.lm_test(
  pred_df,
  cov_df,
  brain_data,
  gene_data,
  annoData,
  cor_method = c("pearson", "spearman", "pls1c", "pls1w", "custom"),
  aggre_method = c("mean", "median", "meanabs", "meansqr", "maxmean", "ks_orig",
    "ks_weighted", "ks_pos_neg_sum", "sign_test", "rank_sum", "custom"),
  null_model = c("spin_brain", "resample_gene", "coexp_matched"),
  minGSSize = 10,
  maxGSSize = 200,
  n_cores = 0,
  n_perm = 5000,
  perm_id = NULL,
  coord.l = NULL,
  coord.r = NULL,
  seed = NULL,
  threshold_type = c("sd", "percentile", "none"),
  threshold_value = 1,
  pvalueCutoff = 0.05,
  pAdjustMethod = "fdr",
  matchcoexp_tol = 0.05,
  matchcoexp_max_iter = 1e+06
)
```

Arguments

`pred_df` Data frame of predictor variables.

`cov_df` Data frame of covariate variables.

`brain_data` Data frame of brain imaging data.

`gene_data` Data frame of gene expression data.

`annoData` Environment containing annotation data.

`cor_method` Character string specifying the correlation method. Default is 'pearson'. Other options include 'spearman', 'pls1c', 'pls1w', 'custom'.

`aggre_method` Character string specifying the aggregation method. Default is 'mean'. Other options include 'median', 'meanabs', 'meansqr', 'maxmean', 'ks_orig', 'ks_weighted', 'ks_pos_neg_sum', 'sign_test', 'rank_sum', 'custom'.

`null_model` Character string specifying the null model method. Default is 'spin_brain'. Other options include 'resample_gene', 'coexp_matched'.

`minGSSize` Integer specifying the minimum gene set size. Default is 10.

`maxGSSize` Integer specifying the maximum gene set size. Default is 200.

`n_cores` Integer specifying the number of cores to use for parallel processing. Default is 0.

`n_perm` Integer specifying the number of permutations. Default is 5000.

`perm_id` Optional permutation ID.

`coord.l` Optional left hemisphere coordinates.

`coord.r` Optional right hemisphere coordinates.

`seed` Optional random seed for generating perm_id.

`threshold_type` Character string specifying the threshold type for core genes. Default is 'sd'. Other options include 'percentile'.

`threshold_value`
Numeric value specifying the threshold level. Default is 1.

`pvalueCutoff` Numeric value specifying the p-value cutoff for significant results. Default is 0.05.

`pAdjustMethod` Character string specifying the method for p-value adjustment. Default is 'fdr'.

`matchcoexp_tol` Numeric value specifying the tolerance for matched coexpression. Default is 0.05.

`matchcoexp_max_iter`
Integer specifying the maximum number of iterations for matched coexpression. Default is 1000000.

Value

A data frame containing the results of the linear model test, including p-values, adjusted p-values, q-values, descriptions, and core genes.

brainscore.simulate *Perform Brain Score Simulation*

Description

This function performs simulations on brain score data using different methods for comparison. It calculates gene set scores, performs linear modeling, and returns the simulation results.

Usage

```
brainscore.simulate(
  pred_df,
  cov_df,
  brain_data,
  gene_data,
  annoData,
  sim_n = 1000,
  subsample_size = 100,
  sim_type = c("randomize_pred", "spin_brain", "resample_gene"),
  cor_method = c("pearson", "spearman", "pls1c", "pls1w", "custom"),
  aggre_method = c("mean", "median", "meanabs", "meansqr", "maxmean", "ks_orig",
    "ks_weighted", "ks_pos_neg_sum", "sign_test", "rank_sum", "custom"),
  minGSSize = 10,
  maxGSSize = 200,
  n_cores = 0,
  n_perm = 5000,
  perm_id = NULL
)
```

Arguments

pred_df Data frame of predictor variables.

cov_df Data frame of covariate variables.

brain_data Data frame of brain imaging data.

gene_data Data frame of gene expression data.

annoData Environment containing annotation data.

sim_n Integer specifying the number of simulations. Default is 1000.

subsample_size Integer or vector specifying the subsample sizes. Default is 100.

sim_type Character string specifying the simulation type. Default is 'randomize_pred'. Other options include 'spin_brain', 'resample_gene', 'coexp_matched'.

cor_method Character string specifying the correlation method. Default is 'pearson'. Other options include 'spearman', 'pls1c', 'pls1w', 'custom'.

aggre_method Character string specifying the aggregation method. Default is 'mean'. Other options include 'median', 'meanabs', 'meansqr', 'maxmean', 'ks_orig', 'ks_weighted', 'ks_pos_neg_sum', 'sign_test', 'rank_sum', 'custom'.

minGSSize Integer specifying the minimum gene set size. Default is 10.

maxGSSize Integer specifying the maximum gene set size. Default is 200.

`n_cores`	Integer specifying the number of cores to use for parallel processing. Default is 0.
`n_perm`	Integer specifying the number of permutations. Default is 5000.
`perm_id`	Optional permutation ID.

Value

A list of data frames containing the results of the simulations.

`brain_data` *Brain Data PC1 for Left Hemisphere*

Description

This dataset contains PC1 data filtered for regions starting with 'L_' in the Desikan atlas.

Usage

```
brain_data
```

Format

A data frame with rows as regions and columns as effect sizes of case-control comparisons on regional cortical thickness between bipolar disorders and healthy controls.

Source

read.csv('data-raw/desikan_PC1_data.csv')

`calculate_pvals` *Calculate P-Values*

Description

This function calculates p-values based on the provided true statistics and null statistics lists. It supports two methods for p-value calculation: 'standard' and 'split_pos_neg'.

Usage

```
calculate_pvals(
  statList.true,
  statList.null,
  method = c("standard", "split_pos_neg")
)
```

Arguments

`statList.true`	A named list of true statistics.
`statList.null`	A named list of null statistics corresponding to the true statistics.
`method`	The method to be used for p-value calculation. Either 'standard' or 'split_pos_neg'. Default is 'standard'.

Value

A list of calculated p-values.

`coord_dk_lh` *Desikan Centroid Coordinates for Left Hemisphere*

Description

This dataset contains the centroid coordinates for the Desikan atlas regions in the left hemisphere.

Usage

```
coord_dk_lh
```

Format

A data frame with rows as regions and columns as coordinates (x, y, z).

Source

read.csv('data-raw/desikan_centroid.csv')

`corr_brain_gene` *Calculate correlations or associations between gene and brain data*

Description

Calculate correlations or associations between gene and brain data

Usage

```
corr_brain_gene(
  gene_data,
  brain_data,
  method = c("pearson", "spearman", "pls1c", "pls1w", "custom"),
  r2z = TRUE
)
```

Arguments

`gene_data`	A data frame or matrix of gene expression data.
`brain_data`	A data frame or matrix of brain data.
`method`	The method to be used for correlation/association. Can be 'pearson', 'spearman', 'pls1c', 'pls1w', or a custom function provided by the user.
`r2z`	Logical, indicating whether to convert correlation coefficients to Fisher's Z scores. Only applicable to 'pearson'.

Value

A matrix with correlation or association coefficients between gene data and brain data.

filter_geneSetList *Filter Gene Set List*

Description

This function filters a list of gene sets based on the background genes and specified size constraints.

Usage

```
filter_geneSetList(bg_genes, geneSetList, minGSSize, maxGSSize)
```

Arguments

bg_genes	A vector of background gene symbols to be used for filtering.
geneSetList	A list of gene sets to be filtered.
minGSSize	Minimum gene set size for filtering.
maxGSSize	Maximum gene set size for filtering.

Value

A filtered list of gene sets that meet the size constraints and background genes criteria.

find_core_genes *Find Core Genes Influencing Aggregated Score or LM Coefficients between molecular profile and behavioral data*

Description

This function performs a Leave-One-Out (LOO) analysis on gene sets to determine core genes that influence the aggregated score. It can utilize parallel processing to enhance computation efficiency and supports two types of analysis: one that considers only gene sets and another that includes predictor and covariate data frames.

Usage

```
find_core_genes(
  geneList,
  geneSetList,
  pred_df = NULL,
  cov_df = NULL,
  aggre_method,
  n_cores = 1,
  threshold_type = c("sd", "percentile"),
  threshold_value = 1
)
```

Arguments

`geneList` A matrix of genes by subs, each column representing a subject / a group-level result.

`geneSetList` A list of gene sets, each containing names of genes.

`pred_df` Optional data frame of a predictor. If NULL, it is perfomred for group-level enrichment.

`cov_df` Optional data frame of covariates. If NULL, it is perfomred for group-level enrichment.

`aggre_method` The aggregation method used to compute the scores.

`n_cores` The number of cores to use for parallel processing; defaults to 1. Uses all available cores minus one if set to 0.

`threshold_type` The method to determine significance ('sd' for standard deviation, 'percentile' for percentile threshold).

`threshold_value` Numeric value specifying the threshold level; meaning depends on `threshold_type`.

Value

A list of core genes for each gene set.

`generate_null_brain_data` *Generate Null Brain Data*

Description

This function generates null brain datasets based on permutations provided in perm_id. It rearranges the brain data according to the permutations and outputs the shuffled datasets.

Usage

```
generate_null_brain_data(brain_data, perm_id)
```

Arguments

`brain_data` A matrix representing brain data, where each row corresponds to a region.

`perm_id` A matrix of permutations, where each column represents a permutation and each row corresponds to an index in `brain_data`.

Value

A matrix of null brain data with the same dimensions as `brain_data` but with permuted rows according to `perm_id`.

get_annoData *Load gene sets annotation data*

Description

This function loads annotation data from RDS files located in the `inst/extdata/geneSets` directory of the package. If the specified file does not exist locally, it will be downloaded from the GitHub repository.

Usage

```
get_annoData(
  type = c("CellTypes_Lake2018", "CellTypes_Martins2021", "CellTypes_Seidlitz2020",
    "DGN", "GO_BP", "GO_CC", "GO_MF", "KEGG", "Reactome", "SynGO", "WikiPathways")
)
```

Arguments

`type` A character string specifying the type of gene set to load. Options are:

"CellTypes_Lake2018" Cell types data from Lake2018

"CellTypes_Martins2021" Cell types data from Martins2021

"CellTypes_Seidlitz2020" Cell types data from Seidlitz2020

"DGN" DisGeNET gene sets

"GO_BP" Gene Ontology Biological Process

"GO_CC" Gene Ontology Cellular Component

"GO_MF" Gene Ontology Molecular Function

"KEGG" KEGG gene sets

"Reactome" Reactome gene sets

"SynGO" SynGO gene sets

"WikiPathways" WikiPathways gene sets

Value

A data frame containing the annotation data.

Examples

```
## Not run:
annoData <- get_annoData("GO_BP")

## End(Not run)
```

get_geneExp *Get Gene Expression Data*

Description

This function retrieves gene expression data based on specified parameters. #' This function loads gene expression data from CSV files located in the `inst/extdata/geneExp` directory of the package. If the specified file does not exist locally, it will be downloaded from the GitHub repository.

Usage

```
get_geneExp(
  atlas = c("desikan", "schaefer100", "schaefer200", "schaefer300"),
  rdonor = c("r0.2", "r0.4", "r0.6"),
  hem = c("L", "R", "B")
)
```

Arguments

`atlas`	A character string specifying the atlas to use. Options are "desikan", "schaefer100", "schaefer200", "schaefer300".
`rdonor`	A character string specifying the donor resolution to use. Options are "r0.2", "r0.4", "r0.6".
`hem`	A character string specifying the hemisphere to use. Options are "L" (Left), "R" (Right), "B" (Both).

Details

The data is obtained from the ENIGMA-TOOLBOX. Please cite the ENIGMA-TOOLBOX: Larivière, S., Paquola, C., Park, B. Y., Royer, J., Wang, Y., Benkarim, O., ... & Bernhardt, B. C. (2021). The ENIGMA Toolbox: multiscale neural contextualization of multisite neuroimaging datasets. Nature Methods, 18(7), 698-700.

Value

A matrix containing the gene expression data.

Examples

```
## Not run:
geneExpMatrix <- get_geneExp("desikan", "r0.4", "L")

## End(Not run)
```

get_geneSetList *Get Gene Set List*

Description

This function retrieves a gene set list from annotation data. It optionally converts gene identifiers to gene symbols.

Usage

```
get_geneSetList(annoData)
```

Arguments

annoData	Annotation data to retrieve gene sets from.

Value

A list of gene sets.

get_termDescription *Get Gene Set Descriptions*

Description

This function retrieves descriptions for gene sets from annotation data.

Usage

```
get_termDescription(term, annoData, strip_prefix = "")
```

Arguments

term	term to search from annoData (can be a vector of terms).
annoData	An environment containing annotation data.
strip_prefix	A character string to remove from the beginning of each term.

Value

A character vector of gene set descriptions.

identify_core_genes *Identify core genes based on a specified threshold method*

Description

Identify core genes based on a specified threshold method

Usage

```
identify_core_genes(
  changes,
  threshold_type = c("sd", "percentile"),
  threshold_value = 1
)
```

Arguments

changes	Named vector of changes from LOO analysis.
threshold_type	Character string indicating the method to determine the threshold ("sd" or "percentile").
threshold_value	Numeric value indicating the percentile (if method is "percentile") or the number of standard deviations (if method is "sd").

Value

Vector of core genes or NA if no core genes are identified.

perm_id_dk_lh_5000 *Permutation index for left Desikan regions (5000 permutations)*

Description

Permutation index for left Desikan regions (5000 permutations)

Usage

```
perm_id_dk_lh_5000
```

Format

A matrix with rows as regions and columns as permutated indices.

Source

rotate_parcellation(coord.l = coord_dk_lh, nrot = 5000) plot_brain *Plot Brain Data*

Description

This function creates a brain plot using the ggplot2 and ggseg packages.

Usage

```
plot_brain(
  df2plot,
  ats = c("dx", "dk", "aseg"),
  what2plot = "statistic",
  filterby = c("p.value", "p.adj", "none"),
  title2show = "",
  limit2show = c(-15, 15),
  legend2show = "Stat",
  hide_legend = FALSE,
  hem = "both",
  low = "steelblue1",
  mid = "white",
  high = "firebrick1",
  sufix2remove = "_thickavg"
)
```

Arguments

df2plot	A data frame containing the data to plot.
ats	A character string indicating the atlas to use ('dx', 'dk', 'aseg').
what2plot	A character string indicating the variable to plot ('statistic').
filterby	A character string indicating the filter to apply ('p.value', 'p.adj', 'none').
title2show	A character string indicating the title of the plot.
limit2show	A numeric vector of length 2 indicating the limits for the color scale.
legend2show	A character string indicating the legend title.
hide_legend	A logical value indicating whether to hide the legend.
hem	A character string indicating which hemisphere to plot ('both', 'left', 'right').
low	A character string indicating the color for the low end of the scale.
mid	A character string indicating the color for the midpoint of the scale.
high	A character string indicating the color for the high end of the scale.
sufix2remove	A character string indicating the suffix to remove from labels.

Value

A ggplot2 object.

resample_gene *Resample Gene List*

Description

Generates a set of permuted gene lists from the original gene list, ensuring uniqueness in the permuted sets.

Usage

```
resample_gene(geneList.true, n_perm = 5000)
```

Arguments

geneList.true	A matrix of gene expression data.
n_perm	Number of permutations to generate.

Value

A matrix of permuted gene lists.

resample_geneSetList_matching_coexp *Resample Gene Sets with Specified Constraints*

Description

This function resamples gene sets based on specific constraints like matching co-expression patterns. The methodology implemented is informed by Wei et al. (2022) on statistical testing in transcriptomic-neuroimaging studies. It is important to note that restricting null models to a subset of genes can be problematic. The empirical statistics sampled from the full gene pool differ from those derived from a restricted pool. Therefore, usage of this approach should be with caution.

Usage

```
resample_geneSetList_matching_coexp(
  gene_data,
  geneSetList,
  tol = 0.01,
  max_iter = 1e+06,
  n_perm = 5000,
  n_cores = 1
)
```

Arguments

`gene_data` A matrix or data frame representing gene expression data.

`geneSetList` A list of gene sets to be resampled.

`tol` A numeric value indicating the tolerance for matching co-expression patterns (default = 0.01).

`max_iter` An integer indicating the maximum number of iterations for the sampling process (default = 1000000).

`n_perm` An integer indicating the number of permutations to generate (default = 5000).

`n_cores` An integer indicating the number of cores to use for parallel processing (default = 1).

Value

A list of resampled gene sets based on the specified constraints.

References

Wei, Y., de Lange, S. C., Pijnenburg, R., Scholtens, L. H., Ardesch, D. J., Watanabe, K., Posthuma, D., & van den Heuvel, M. P. (2022). Statistical testing in transcriptomic-neuroimaging studies: A how-to and evaluation of methods assessing spatial and gene specificity. Human Brain Mapping, 43(3), 885–901. https://doi.org/10.1002/hbm.25711

`rotate_parcellation` *Rotate Parcellation*

Description

Generate a permutation map from a set of cortical regions of interest to itself, while (approximately) preserving contiguity and hemispheric symmetry. The function is based on a rotation of the FreeSurfer projection of coordinates of a set of regions of interest on the sphere. #' This function is modified from the original version available at: https://github.com/frantisekvasa/rotate_parcellation

Usage

```
rotate_parcellation(
  coord.l = NULL,
  coord.r = NULL,
  nrot = 5000,
  method = c("hungarian", "vasa"),
  seed = NULL
)
```

Arguments

`coord.l` Coordinates of left hemisphere regions on the sphere (array of size n(LH regions) x 3). Can be NULL if only right hemisphere is used.

`coord.r` Coordinates of right hemisphere regions on the sphere (array of size n(RH regions) x 3). Can be NULL if only left hemisphere is used.

`nrot` Number of rotations (default = 5000).

`method`	Method to match rotated and unrotated regions; options are 'vasa' (faster, can be suboptimal) or 'hungarian' (default, slower, optimal).
`seed`	Seed for reproducibility.

Details

Modifications include:

- Added support for scenarios where only one hemisphere's coordinates are provided.
- Improved handling of coordinate dimensions and conditional concatenation of reference and rotation indices.
- Included importFrom directives for required functions from `matrixStats` and `clue`.
- Ensured the function generates `nrot + 100` permutations, removes duplicates, and returns exactly `nrot` unique permutations.

Value

Array of permutations, from set of regions to itself (array of size n(total regions) x nrot).

`sample_gs_matching_coexp` *Sample Gene Sets Matching Co-Expression*

Description

This function samples gene sets that closely match the co-expression profile of a target gene set. The methodology implemented is informed by Wei et al. (2022) on statistical testing in transcriptomic-neuroimaging studies.

Usage

```
sample_gs_matching_coexp(
  gs,
  coexp_matrix,
  tol = 0.01,
  max_iter = 1e+06,
  n_target = 5000
)
```

Arguments

`gs`	The target gene set for which similar co-expression profiles are sought.
`coexp_matrix`	A co-expression matrix, typically calculated as the correlation matrix of gene expression data.
`tol`	Tolerance for the difference between the co-expression of the target and the sampled gene sets.
`max_iter`	Maximum number of iterations to attempt finding matches.
`n_target`	Number of gene sets to sample.

Value

A list of gene sets that closely match the target gene set's co-expression profile.

References

Wei, Y., de Lange, S. C., Pijnenburg, R., Scholtens, L. H., Ardesch, D. J., Watanabe, K., Posthuma, D., & van den Heuvel, M. P. (2022). Statistical testing in transcriptomic-neuroimaging studies: A how-to and evaluation of methods assessing spatial and gene specificity. Human Brain Mapping, 43(3), 885–901. https://doi.org/10.1002/hbm.25711

`simple_lm` *Perform Linear Regression with Multiple Predictors and Covariates*

Description

This function fits linear models for specified dependent variables using given predictors and covariates. It returns a data frame containing model summaries.

Usage

```
simple_lm(
  dependent_df,
  pred_df,
  cov_df,
  stat2return = c("all", "tval", "pval", "tval_list")
)
```

Arguments

`dependent_df` A data frame containing the dependent variables.

`pred_df` A data frame containing the predictor variables.

`cov_df` A data frame containing the covariate variables.

`stat2return` A character string specifying which statistic to return ("statistic", "p.value", or "full"). Default is "full". "statistic" returns only the t-value for permutation purposes, "p.value" returns only the p-value for simulation analysis, and "full" returns all information for the parametric test.

Value

A data frame containing model summaries. Depending on `stat2return`, the output can include different statistics:

- If `stat2return` is "all", the output includes unstandardized and standardized coefficients, standard errors, t-values, confidence intervals, p-values, adjusted p-values, and significance markers.
- If `stat2return` is "tval", the output includes only the t-values.
- If `stat2return` is "tval", the output includes only the t-values as a list.
- If `stat2return` is "pval", the output includes only the p-values.

swap_geneList *Swap Gene List*

Description

This function swaps the original values of a gene set (`orig_gs`) with the sampled gene sets (`sampled_gs`) within the given `geneList.true`. The resulting `geneList.null` is in the same format as generated by other approaches.

Usage

```
swap_geneList(geneList.true, orig_gs, sampled_gs)
```

Arguments

`geneList.true`	A matrix representing the true gene list, with dimensions m x 1.
`orig_gs`	A vector of original gene set identifiers.
`sampled_gs`	A list of sampled gene sets, each being a vector of gene identifiers.

Value

A matrix where each column represents a null gene list generated by swapping `orig_gs` with each set in `sampled_gs`.

Index