Package ‘sSNAPPY’

April 11, 2023

Title Single Sample directioNAl Pathway Perturbation analYsis

Version 1.2.5

Description A single sample pathway perturbation testing method for RNA-seq data. The method propagates changes in gene expression
down gene-set topologies to compute single-sample directional pathway perturbation scores that reflect potential directions
of changes.Perturbation scores can be used to test significance of pathway perturbation at both individual-sample and
treatment levels.

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.2

Suggests BiocManager, BiocStyle, cowplot, DT, htmltools, knitr,
pander, rmarkdown, spelling, stringr, testthat (>= 3.0.0),
tidyverse

Config/testthat/edition 3

SystemRequirements C++11

LazyData false

Imports dplyr, magrittr, rlang, stats, purrr, BiocParallel, graphite,
Rcpp, tibble, ggplot2, ggraph, igraph, reshape2, org.Hs.eg.db,
SummarizedExperiment, edgeR, methods, ggforce, ggnewscale,
pheatmap, utils

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 4.2.0)

biocViews Software, GeneExpression, GeneSetEnrichment, GeneSignaling

URL `https://wenjun-liu.github.io/sSNAPPY/`

VignetteBuilder knitr

BugReports `https://github.com/Wenjun-Liu/sSNAPPY/issues`

Language en-US

git_url https://git.bioconductor.org/packages/sSNAPPY

git_branch RELEASE_3_16

git_last_commit 483e456

git_last_commit_date 2023-03-28

Date/Publication 2023-04-10

Author Wenjun Liu [aut, cre] (<https://orcid.org/0000-0002-8185-3069>)

Maintainer Wenjun Liu <wenjun.liu@adelaide.edu.au>

R topics documented:

generate_permuted_scores *Permute sample labels to simulate null distribution of perturbation scores*

Description

Simulate null distributions of perturbation scores for each pathway through sample permutation.

Usage

```
generate_permuted_scores(
  expreMatrix,
  numOfTreat,
  NB = 1000,
  gsTopology,
  weight,
  BPPARAM = BiocParallel::bpparam()
)

## S4 method for signature 'matrix'
generate_permuted_scores(
  expreMatrix,
  numOfTreat,
  NB = 1000,
  gsTopology,
  weight,
  BPPARAM = BiocParallel::bpparam()
)

## S4 method for signature 'data.frame'
generate_permuted_scores(
  expreMatrix,
  numOfTreat,
  NB = 1000,
  gsTopology,
  weight,
  BPPARAM = BiocParallel::bpparam()
)

## S4 method for signature 'DGEList'
generate_permuted_scores(
  expreMatrix,
  numOfTreat,
  NB = 1000,
  gsTopology,
  weight,
  BPPARAM = BiocParallel::bpparam()
)

## S4 method for signature 'SummarizedExperiment'
generate_permuted_scores(
  expreMatrix,
  numOfTreat,
  NB = 1000,
  gsTopology,
  weight,
  BPPARAM = BiocParallel::bpparam()
)
```

Arguments

expreMatrix	matrix and data.frame of logCPM, or DGEList/SummarizedExperiment storing gene expression counts. Feature names need to be in entrez IDs
numOfTreat	Number of treatments (including control)

`NB` Number of permutations to perform

`gsTopology` List of pathway topology matrices generated using function `retrieve_topology`

`weight` A vector of gene-wise weights derived from function `weight_ss_fc`

`BPPARAM` The parallel back-end to uses, if not specified, it is defaulted to the one returned by `BiocParallel::bpparam()`.

Details

This `generate_permuted_scores` function is a generic function that can deal with multiple types of inputs. It firstly randomly permute sample labels `NB` times to generate permuted logFCs, which are then used to compute permuted perturbation scores for each pathway.

The function outputs a list that is of the same length as the list storing pathway topology matrices. Each element of the output list is for a pathway and contains a vector of permuted perturbation score of length. The permuted perturbation scores will be used to estimate the null distributions of perturbation scores.

If the input is S4 object of `DGEList` or `SummarizedExperiment`, gene expression matrix will be extracted and converted to a logCPM matrix.

The default number of permutation (`NB`) is set to 1000. If the requested `NB` is larger than the maximum number of permutations possible, `NB` will be set to the largest number of permutations possible instead.

Value

A list where each element is a vector of perturbation scores for a pathway.

Examples

```
#compute weighted single sample logFCs
data(metadata_example)
data(logCPM_example)
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
 factor = "patient", control = "Vehicle")
## Not run:
load(system.file("extdata", "gsTopology.rda", package = "sSNAPPY"))

# simulate the null distribution of scores through sample permutation
permutedScore <- generate_permuted_scores(logCPM_example, numOfTreat = 3,
 NB = 10, gsTopology = gsTopology, weight = ls$weight)

# To see what other parallel back-end can be used:
 BiocParallel::registered()

## End(Not run)
```

`gsAnnotation_df` *gsAnnotation_df: Categorization of KEGG pathways used for community annotation*

Description

gsAnnotation_df: Categorization of KEGG pathways used for community annotation

Usage

```
data(gsAnnotation_df)
```

Format

A `data.frame` with 549 rows and 2 columns containing categorization of 549 KEGG pathways

gs_name Gene-set name

vategory Category

Source

https://www.genome.jp/kegg/

`logCPM_example` *logCPM_example: Normalised logCPM of patient-derived explant models obtained from 5 ER-positive primamry breast cancer patients (GSE80098)*

Description

This data was adopted from a study by Singhal H, et al., which was published as *Genomic agonism and phenotypic antagonism between estrogen and progesterone receptors in breast cancer* in 2016.

Usage

```
data(logCPM_example)
```

Format

A matrix with 7672 rows and 15 columns

Details

In this study, 12 primary malignant breast tissues (8PR+ and 4 PR-) were developed into patient-derived explants and treated with Vehicle, E2, E2+R5020, or R5020 for 24 or 48 hrs.

Raw data for 48-hr Vehicle-, R5020-treated and E2+R5020-treated samples were retrieved from GEO (GSE80098) and pre-processed into raw count. Filtration was sequentially performed to remove undetectable genes and the filtered counts were normalised using conditional quantile normalisation to offset effects of systematic artefacts, such as gene length and GC contents.

To reduce computing time, we randomly sampled half of the genes after filtration and used their logCPM value as the example data.

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE80098

`metadata_example` *metadata_example: Sample metadata for malignant breast cancer tumours PDE from 5 ER+ breast cancer patients (GSE80098)*

Description

metadata_example: Sample metadata for malignant breast cancer tumours PDE from 5 ER+ breast cancer patients (GSE80098)

Usage

```
data(metadata_example)
```

Format

A `data.frame` with 15 rows and 4 columns

patient patient N2-3, P4-6

treatment treatment: Vehicle, R5020 or E2+R5020

PR progesterone receptor status

sample sample name, corresponding to column names of the logCPM matrix

Source

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4928895/

normalise_by_permu *Normalise test perturbation scores by permutation results*

Description

Normalise test perturbation scores by permutation results

Usage

```
normalise_by_permu(permutedScore, testScore, pAdj_method = "fdr")
```

Arguments

`permutedScore` A list. Output of `generate_permuted_scores`

`testScore` A matrix. Output of `weight_ss_fc`

`pAdj_method` Method for adjusting p-values for multiple comparisons. See `?p.adjust` for methods available. Default to FDR.

Details

Normalise the test perturbation scores generated by `weight_ss_fc()` through the permuted perturbation scores derived from the `generate_permuted_scores()` function. The mean absolute deviation(MAD) and median of perturbation scores for each pathway are firstly derived from the permuted perturbation scores. The test perturbation scores are then converted to robust z-scores using MADs and medians calculated.

Value

A `data.frame`

Examples

```
## Not run:
load(system.file("extdata", "gsTopology.rda", package = "sSNAPPY"))
data(metadata_example)
data(logCPM_example)
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
factor = "patient", control = "Vehicle")

# compute raw gene-wise perturbation scores
genePertScore <- raw_gene_pert(ls$logFC, gsTopology)

# sum gene-wise perturbation scores to derive the pathway-level single-sample perturbation scores
pathwayPertScore <- pathway_pert(genePertScore)

# simulate the null distribution of scores through sample permutation
permutedScore <- generate_permuted_scores(logCPM_example, numOfTreat = 3,
 NB = 5, gsTopology = gsTopology, weight = ls$weight)
```

```
# normlise the test perturbation scores using the permutation results
normalisedScores <- normalise_by_permu(permutedScore, pathwayPertScore)

## End(Not run)
```

pathway_pert *Compute Single-sample Pathway-level Perturbation Score*

Description

Sum gene-wise raw perturbation scores within each sample to derive single-sample perturbation scores for each pathway

Usage

```
pathway_pert(genePertScore)
```

Arguments

genePertScore	List of gene-wise raw perturbation score matrices generated using function raw_gene_pert()

Value

A data.frame with 3 columns: tA (single-sample pathway-level perturbation score), sample, and gs_name (gene-set name)

References

Tarca AL, Draghici S, Khatri P, Hassan SS, Mittal P, Kim JS, Kim CJ, Kusanovic JP, Romero R. A novel signaling pathway impact analysis. Bioinformatics. 2009 Jan 1;25(1):75-82.

Examples

```
#compute weighted single sample logFCs
data(metadata_example)
data(logCPM_example)
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
factor = "patient", control = "Vehicle")
# extract all the KEGG pathways
gsTopology <- retrieve_topology(database = "kegg")
# compute raw gene-wise perturbation scores
genePertScore <- raw_gene_pert(ls$logFC, gsTopology)
# sum gene-wise perturbation scores to derive the pathway-level single-sample perturbation scores
pathwayPertScore <- pathway_pert( genePertScore)
```

`plot_community` *Visualise the community structure in significantly perturbed gene-set network*

Description

Visualise the community structure in significantly perturbed gene-set network

Usage

```
plot_community(
  normalisedScores,
  gsTopology,
  gsAnnotation = NULL,
  colorBy = c("robustZ", "pvalue", "community"),
  communityMethod = "cluster_louvain",
  foldGSname = TRUE,
  foldafter = 2,
  layout = "fr",
  markCommunity = "ellipse",
  markAlpha = 0.2,
  edgeAlpha = 0.8,
  up_col = "brown3",
  down_col = "steelblue3",
  scale_edgeWidth = c(0.5, 3),
  edgeLegend = FALSE,
  scale_nodeSize = c(3, 6),
  nodeShape = 16,
  color_lg_title = NULL,
  lb_size = 3,
  lb_color = "black",
  plotIsolated = FALSE,
  ...
)
```

Arguments

`normalisedScores`
A `data.frame` derived from the `normalise_by_permu()` function

`gsTopology` List of pathway topology matrices generated using function `retrieve_topology()`

`gsAnnotation` A `data.frame` containing gene-sets categorisations for pathway annotation. Must contain at least two columns: `c("gs_name", "category")`, where `gs_name` denotes gene-sets names that are matched to names of pathway topology matrices, and `category` records the categorization of each pathway. If customized annotation is not provided, it's assumed that the pathways investigated were from the KEGG database and the inbuilt KEGG pathway annotation information will be used `colorBy` Choose to color nodes either by *community*, *robustZ* or *pvalue*. To color by *robustZ* or *pvalue*, a column must exist in the `normalisedScores` `data.frame` for the chosen parameter

`communityMethod`
A community detection method supported by `igraph`. See details for all methods available.

`foldGSname` `logical`. Should long gene-set names be folded into two lines

`foldafter` The number of words after which gene-set names should be folded. Defaulted to 2

`layout` The layout algorithm to apply. Accepted layouts are `"fr", "dh", "gem", "graphopt", "kk", "lgl", "`

`markCommunity` `character` A `geom_mark_*` method supported by `ggforce` to annotate sets of nodes belonging to the same community. Either `*NULL*, *ellipse*, *circle*, *hull*, *rect*`

`markAlpha` Transparency of annotation areas. Default to 0.2

`edgeAlpha` Transparency of edges. Default to 0.8

`up_col` The color used to label activated gene-sets. Only applicable if `colorBy` is set to be "robustZ"

`down_col` The color used to label inhibited gene-sets. Only applicable if `colorBy` is set to be "robustZ"

`scale_edgeWidth`
A numerical vector of length 2 to be provided to `ggraph::scale_edge_width_continuous()` for specifying the minimum and maximum edge widths after transformation. Defaulted to c(0.5, 3)

`edgeLegend` `logical` Should edge weight legend be shown

`scale_nodeSize` A numerical vector of length 2 to be provided to `ggplot2::scale_size()` for specifying the minimum and maximum node sizes after transformation. Defaulted to c(3,6)

`nodeShape` The shape to use for nodes

`color_lg_title` Title for the color legend

`lb_size` Size of node text labels

`lb_color` Color of node text labels

`plotIsolated` `logical`.Should nodes not connected to any other nodes be plotted. Default to FALSE

`...` Used to pass various potting parameters to `ggforce::geom_mark_*()`

Details

A community detection strategy specified by `communityMethod` will be applied to the pathway-pathway network, and communities will be annotated with the pathway category that had the highest number of occurrence, denoting the main biological processes perturbed in that community.

At the moment, only categorisations of KEGG pathway were built into the package, so if the provided `normalisedScores` contains perturbation scores of pathways derived from other databases, annotation of communities will not be performed unless pathway information is provided through the `gsAnnotation` parameter. The categorisation information needs to be stored in a `data.frame` containing `gs_name` (gene-set names) and `category` (categorisation of the given pathways).

Plotting parameters accepted by `geom_mark_*` could be passed to the function to adjust the annotation area or the annotation label. See `?ggforce::geom_mark_*` for more details.

Value

A ggplot2 object

Examples

```
load(system.file("extdata", "gsTopology.rda", package = "sSNAPPY"))
load(system.file("extdata", "normalisedScores.rda", package = "sSNAPPY"))
#Subset the first 10 rows of the normalisedScores data.frame as an example
subset <- normalisedScores[1:15,]
# Color network plot nodes by the community they were assigned to and mark nodes belonging
# to the same community by ellipses
plot_community(subset, gsTopology, colorBy = "community",layout = "kk",
color_lg_title = "Community")

# Color network plot nodes by pathways' directions of changes and mark nodes belonging
# to the same community by ellipses
plot_community(subset, gsTopology, colorBy = "robustZ",layout = "kk",
color_lg_title = "Direction of pathway perturbation")

# To change the colour and fill of `geom_mark_*` annotation, use any
# `scale_fill_*` and/or `scale_color_*`
# functions supported by `ggplot2`. For example:
p <- plot_community(subset, gsTopology, colorBy = "robustZ",layout = "kk",
markCommunity = "rect",color_lg_title = "Direction of pathway perturbation")
p + ggplot2::scale_color_ordinal() + ggplot2::scale_fill_ordinal()
```

plot_gene_contribution *Plot genes' contribution to a specific pathway's perturbation as heatmap*

Description

Plot genes' contribution to a specific pathway's perturbation as heatmap

Usage

```
plot_gene_contribution(
  genePertScore,
  gsToPlot,
  mapEntrezID = NULL,
  metadata = NULL,
  annotation_attribute = "pathwayPertScore",
  pathwayPertScore = NULL,
  ...
)
```

Arguments

`genePertScore` List of gene-wise perturbation scores generated using function `raw_gene_pert()`

`gsToPlot` `character` Name of the pathway to be plotted

`mapEntrezID` Optional. A `data.frame` matching genes' entrez IDs to other identifiers, such as gene names. Must contain 2 columns: `"entrezid"` and `"mapTo"`

`metadata` A `data.frame` containing sample metadata for heatmap annotation

`annotation_attribute`
`character` Vector specifying attributes to draw annotations for. Default to "pathwayPertScore" (ie. pathway-level perturbation scores)

`pathwayPertScore`
A `data.frame` containing pathway-level perturbation scores for each pathway each treated sample. Output of function `pathway_pert()`

`...` Used to pass various potting parameters to `pheatmap::pheatmap()`

Details

The single-sample pathway-level perturbation score for a given pathway is derived from aggregating all the gene-wise perturbation scores of genes in that pathway. This function visualises individual pathway genes' perturbation scores as a heatmap to demonstrate genes' contribution to a pathway perturbation.

Plotting of the heatmap is done through `pheatmap::pheatmap()` so all plotting parameters accepted by `pheatmap::pheatmap()` could also be passed to this function.

It is recommended to provide the pathway-level perturbation scores derived using the `pathway_pert()` function to visualise the directions of changes at pathway-level as a column annotation, which helps the identification of genes driving or antagonizing the perturbation.

Additional annotation attributes could be specified through the `annotation_attribute` parameter and the specified attributes must be provided by columns of the sample metadata `data.frame` provided through the `metadata` parameter, otherwise the attributes will be ignored.

References

Kolde R (2019). *pheatmap: Pretty Heatmaps*. R package version 1.0.12, https://CRAN.R-project.org/package=pheatmap.

Examples

```
#compute weighted single sample logFCs
data(metadata_example)
data(logCPM_example)

# compute single-sample logFCs for all treated samples
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
factor = "patient", control = "Vehicle")

# extract all the KEGG pathways
gsTopology <- retrieve_topology(database = "kegg")
```

```
# compute raw gene-wise perturbation scores
genePertScore <- raw_gene_pert(ls$logFC, gsTopology)
# sum gene-wise perturbation scores to derive the pathway-level single-sample perturbation scores
pathwayPertScore <- pathway_pert(genePertScore)

# Genes' contribution to the perturbation of Estrogen signaling pathway was
# visuaulised with pathway-level perturbation scores
# and treatments as column annotation attributes.
plot_gene_contribution(genePertScore = genePertScore, gsToPlot =
"Estrogen signaling pathway", metadata = metadata_example,
annotation_attribute = c("pathwayPertScore", "treatment"),
pathwayPertScore = pathwayPertScore)
```

plot_gs2gene *Plot pathways and genes contained in them as a network*

Description

Plot pathways and genes contained in them as a network

Usage

```
plot_gs2gene(
  normalisedScores,
  gsTopology,
  geneFC = NULL,
  mapEntrezID = NULL,
  colorGS_By = c("robustZ", "pvalue"),
  foldGSname = TRUE,
  foldafter = 2,
  layout = "fr",
  edgeAlpha = 0.8,
  upGS_col = "brown3",
  downGS_col = "steelblue3",
  upGene_col = "pink",
  downGene_col = "lightblue",
  GeneNode_size = 3,
  GeneNode_shape = 17,
  GsNode_size = 2,
  GsNode_shape = 16,
  label_Gene = TRUE,
  GeneName_size = 3,
  GsName_size = 6,
  gene_lg_title = "Changes in Gene Expression",
  gs_lg_title = "Pathway Perturbation",
  arc_strength = 0.5
)
```

Arguments

`normalisedScores` A `data.frame` derived from the `normalise_by_permu()` function. Only gene-sets of interest should be included

`gsTopology` List of pathway topology matrices generated using function `retrieve_topology()`

`geneFC` An optional named vector of pathways’ fold changes

`mapEntrezID` Optional. A `data.frame` matching genes’ entrez IDs to other identifier. Must contain 2 columns: `"entrezid"` and `"mapTo"`

`colorGS_By` Choose to color nodes by *robustZ* or *pvalue*. A column must exist in the normalisedScores `data.frame` for the chosen parameter

`foldGSname` `logical`. Should long gene-set names be folded into two lines

`foldafter` The number of words after which gene-set names should be folded. Defaulted to 2

`layout` The layout algorithm to apply. Accept all layout supported by `igraph`.

`edgeAlpha` Transparency of edges. Default to 0.8

`upGS_col` Color for activated gene-sets. Only applicable if `colorGS_By` is set to be "robustZ"

`downGS_col` Color for inhibited gene-sets. Only applicable if `colorGS_By` is set to be "robustZ"

`upGene_col` Color for up-regulated genes. Only applicable if `geneFC` is not NULL

`downGene_col` Color for down-regulated genes. Only applicable if `geneFC` is not NULL

`GeneNode_size` Size for gene nodes

`GeneNode_shape` Shape for gene nodes

`GsNode_size` Size for gene-set nodes

`GsNode_shape` Shape for gene nodes

`label_Gene` `logical` Should gene name be plotted

`GeneName_size` Size of gene name label

`GsName_size` Size of gene-set name label

`gene_lg_title` ‘character. Legend for gene nodes color

`gs_lg_title` `character` Legend for gene-set nodes color

`arc_strength` The bend of edges. 1 approximates a halfcircle while 0 will give a straight line.

Details

Taking the perturbation scores of a list of gene-sets derived from `normalise_by_permu()` as input, this function matches gene-set to their associated genes by utilizing information from pathway topology matrices.

It’s optional to provide genes’ logFCs as a named vector, where the names must be genes’ entrez IDs in the format of "ENTREZID:XXXX". This is because pathway topology matrices retrieved through `retrieve_topology()` always use entrez ID as identifiers.

However, it might not be very informative to label genes with their entrez ID. So users can also choose to proivde a mapEntrezID data.frame to match entrez IDs to their chosen identifiers. The data.frame should contain two columns: "entrezid" and "mapTo".

If geneFC is provided, gene nodes will be colored by genes' directions of changes. Otherwise, all gene nodes will be black.

Since some gene-sets could can contain hundreds of genes, it is not recommended to plot all of those genes. If mapEntrezID data.frame is provided, only genes included in that data.frame will be used in the plot.

It is strongly recommended to filter for genes with the highest magnitude of changes. If all pathway genes have to be plotted, consider setting label_Gene to FALSE to turn off plotting all gene names.

Value

A ggplot2 object

Examples

```
load(system.file("extdata", "gsTopology.rda", package = "sSNAPPY"))
load(system.file("extdata", "normalisedScores.rda", package = "sSNAPPY"))
#Subset pathways significantly perturbed in sample R5020_N2_48
subset <- dplyr::filter(normalisedScores, adjPvalue < 0.05, sample == "R5020_N2_48")

# Color gene-sets nodes by robust z-scores.
plot_gs2gene(subset, gsTopology, colorGS_By = "robustZ", label_Gene = FALSE,
GeneNode_size = 1)
# When genes' fold-changes are not provided, gene nodes are colored in black.

# To color genes by their directions of changes, firstly compute genes' single-sample logFCs
data(logCPM_example)
data(metadata_example)
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
 factor = "patient", control = "Vehicle")
# Provide fold-changes of sample R5020_N2_48 as a named vector
plot_gs2gene(subset, gsTopology, geneFC = ls$logFC[,"R5020_N2_48"], colorGS_By = "robustZ",
label_Gene = FALSE)

# There are still a large number of genes, making the plot cumbersome. There only fold-changes of
# genes with top 500 absolute fold-changes are provide so only pathway genes in that list of 500
# genes were plotted.
FC <- sort(abs(ls$logFC[,"R5020_N2_48"]), decreasing = TRUE)[1:500]
plot_gs2gene(subset, gsTopology, geneFC = FC, colorGS_By = "robustZ")

# To make the gene labels more informative, map genes' entrez id to chosen identifiers.
load(system.file("extdata", "entrez2name.rda", package = "sSNAPPY"))
plot_gs2gene(subset, gsTopology, geneFC = FC, mapEntrezID = entrez2name, colorGS_By = "robustZ")
```

plot_gs_network *Plot significantly perturbed gene-sets as a network*

Description

Plot significantly perturbed gene-sets as a network

Usage

```
plot_gs_network(
  normalisedScores,
  gsTopology,
  colorBy = c("robustZ", "pvalue"),
  foldGSname = TRUE,
  foldafter = 2,
  layout = "fr",
  edgeAlpha = 0.8,
  up_col = "brown3",
  down_col = "steelblue3",
  scale_edgeWidth = c(0.5, 3),
  edgeLegend = FALSE,
  scale_nodeSize = c(3, 6),
  nodeShape = 16,
  color_lg = TRUE,
  color_lg_title = NULL,
  lb_size = 3,
  lb_color = "black",
  plotIsolated = FALSE
)
```

Arguments

`normalisedScores`
: A `data.frame` of pathway perturbation scores derived from the `normalise_by_permu()` function

`gsTopology`
: List of pathway topology matrices generated using function `retrieve_topology()`

`colorBy`
: Choose to color nodes either by *robustZ* or *pvalue*. A column must exist in the `normalisedScores` `data.frame` for the chosen parameter

`foldGSname`
: `logical` Should long gene-set names be folded into two lines

`foldafter`
: The number of words after which gene-set names should be folded. Defaulted to 2

`layout`
: The layout algorithm to apply. Accept all layout supported by `igraph`

`edgeAlpha`
: `numerical` Transparency of edges. Default to 0.8

`up_col`
: The color used to label activated gene-sets. Only applicable if `colorBy` is set to be "robustZ"

`down_col` The color used to label inhibited gene-sets. Only applicable if `colorBy` is set to be "robustZ"

`scale_edgeWidth` A numerical vector of length 2 to be provided to `ggraph::scale_edge_width_continuous()` for specifying the minimum and maximum edge widths after transformation. Defaulted to `c(0.5, 3)`

`edgeLegend` logical‘ Should edge weight legend be shown

`scale_nodeSize` A numerical vector of length 2 to be provided to `ggplot2::scale_size()` for specifying the minimum and maximum node sizes after transformation. Defaulted to `c(3,6)`

`nodeShape` Shape of nodes

`color_lg` `logical` Should color legend be shown

`color_lg_title` Optional. Title for the color legend

`lb_size` Size of node text labels

`lb_color` Color of node text labels

`plotIsolated` `logical` Should nodes not connected to any other nodes be plotted. Default to FALSE

Value

A ggplot2 object

Examples

```
load(system.file("extdata", "gsTopology.rda", package = "sSNAPPY"))
load(system.file("extdata", "normalisedScores.rda", package = "sSNAPPY"))

#Subset pathways significantly perturbed in sample R5020_N2_48
subset <- dplyr::filter(normalisedScores, adjPvalue < 0.05, sample == "R5020_N2_48")

# Color network plot nodes by robust z-score
plot_gs_network(subset, gsTopology,
colorBy = "robustZ", layout = "dh",
color_lg_title = "Direction of pathway Perturbation")

# Color network plot nodes by p-values
plot_gs_network(subset, gsTopology, layout = "dh",
colorBy = "pvalue", color_lg_title = "P-value")
```

`rank_gene_pert` *Rank genes by perturbation scores within each sample*

Description

Rank genes by gene-wise raw perturbation scores within each sample to compare genes’ contributions to pathway perturbations.

Usage

```
rank_gene_pert(genePertScore, gsTopology)
```

Arguments

`genePertScore`	List of gene-wise raw perturbation score matrices generated using function `raw_gene_pert()`
`gsTopology`	List of pathway topology matrices generated using function `retrieve_topology()`

Details

Ranking is performed within each sample each pathway. If in a given pathway, both positive and negative gene-wise perturbation scores exist, positive and negative scores are ranked separately, where the larger a positive rank, the more the gene contributed to the pathway's activation, and the smaller a negative rank, the more the gene contributed to the pathways' inhibition. When there's a tie in two gene's perturbation score within a sample, the mean of the indices is used.

Value

A list where each element is a matrix corresponding to a pathway. Each column of an element corresponds to a sample, and each row corresponds to a pathway gene.

Examples

```
#compute weighted single sample logFCs
data(metadata_example)
data(logCPM_example)
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
factor = "patient", control = "Vehicle")
# extract all the KEGG pathways
gsTopology <- retrieve_topology(database = "kegg")
# compute raw gene-wise perturbation scores
genePertScore <- raw_gene_pert(ls$logFC, gsTopology)
# rank genes by gene-wise perturbation scores within each sample
# to compare their contributions to pathway perturbation
geneRank <- rank_gene_pert(genePertScore, gsTopology)
```

`raw_gene_pert` *Compute Gene-wise Perturbation Score*

Description

Propagate weighted single sample logFCs down the pathway topologies to compute gene-wise perturbation score per gene per sample per pathway

Usage

```
raw_gene_pert(weightedFC, gsTopology)
```

Arguments

`weightedFC`	A matrix of weighted single sample logFCs derived from function `weight_ss_fc()`
`gsTopology`	List of pathway topology matrices generated using function `retrieve_topology()`

Details

This function use the algorithm adopted from `SPIA` (see citation) to integrate genes' changes in expression and gene-gene interaction to compute gene-wise perturbation score per gene per sample per pathway. The rownames of the weighted single sample logFC matrix and the pathway topology matrices must use the same type of gene identifier (ie. entrez ID).

Value

A list where each element is a matrix corresponding to a pathway. Each column of an element corresponds to a sample, and each row corresponds to a pathway gene.

References

Tarca AL, Draghici S, Khatri P, Hassan SS, Mittal P, Kim JS, Kim CJ, Kusanovic JP, Romero R. A novel signaling pathway impact analysis. Bioinformatics. 2009 Jan 1;25(1):75-82.

Examples

```
#compute weighted single sample logFCs
data(metadata_example)
data(logCPM_example)
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
factor = "patient", control = "Vehicle")
# extract all the KEGG pathways
gsTopology <- retrieve_topology(database = "kegg")
# compute raw gene-wise perturbation scores
genePertScore <- raw_gene_pert(ls$logFC, gsTopology)
```

`retrieve_topology` *Retrieve pathway topology as weighted adjacent matrix*

Description

Retrieve pathway topology matrices and convert to normalized weighted directed adjacency matrices describing gene signaling networks.

Usage

```
retrieve_topology(database, pathwayName = NULL, beta = NULL)
```

Arguments

`database` See example for supported databases.

`pathwayName` Optional. Subset of pathway names as a vector.

`beta` Optional. A named numeric vector of weights to be assigned to each type of gene/protein relation type. See details for more information.

Details

This function takes the pathway topology information retrieved using `graphite` and convert them to normalized weighted directed adjacency matrices describing the gene signaling network, which can be used to compute gene-wise and pathway-level perturbation score through the scoring algorithm derived from the *SPIA* algorithm. See cited document for more details.

The beta parameter specifies weights to be assigned to each type of gene-gene interaction. It should be a named numeric vector of length 23, whose names must be: `c("activation","compound","binding/association","`
`"indirect","inhibition_phosphorylation","dephosphorylation_inhibition","dissociation","dephosphoryl`
`"state","activation_indirect","inhibition_ubiquination","ubiquination","expression_indirect","indir`
`"repression","binding/association_phosphorylation","dissociation_phosphorylation","indirect_phospho`

If unspecified, beta will be by default chosen as: `c(1,0,0,1,-1,1,0,0,-1,-1,0,0,1,0,1,-1,0,1,-1,-1,0,0,0)`.

The converted weighted adjacent matrices will be stored in a list. We recommend users to store the returned list as a file so this step only needs to be performed once for each database.

This function only supports and can only be used to retreive human databases as this stage.

Value

A list where each element is a matrix corresponding to a pathway

References

Tarca AL, Draghici S, Khatri P, Hassan SS, Mittal P, Kim JS, Kim CJ, Kusanovic JP, Romero R. A novel signaling pathway impact analysis. Bioinformatics. 2009 Jan 1;25(1):75-82. Sales, G., Calura, E., Cavalieri, D. et al. graphite - a Bioconductor package to convert pathway topology to gene network. BMC Bioinformatics 13, 20 (2012).

Examples

```
# explore all databases supported by graphite
dplyr::filter(graphite::pathwayDatabases(), species == "hsapiens")

# retrieve pathway topology matrices frmo the KEGG pathway
gsTopology <- retrieve_topology(database = "kegg")

# if only interested in selected pathways, specify the pathway names in the `pathwayName` parameter
gsTopology <- retrieve_topology(database = "kegg",
pathwayName = c("Glycolysis / Gluconeogenesis",
"Citrate cycle (TCA cycle)","Pentose phosphate pathway"))
```

sSNAPPY — *sSNAPPY: A package for testing directional single sample pathway perturbation*

Description

A package for testing directional single sample pathway perturbation

weight_ss_fc — *Compute weighted single sample LogFCs from normalised logCPM*

Description

Compute weighted single sample logFCs for each treated samples using normalised logCPM values. Fit a lowess curve on variances of single sample logFCs ~ mean of logCPM, and use it to predict a gene-wise weight. The weighted single sample logFCs are ready to be used for computing perturbation scores.

Usage

```
weight_ss_fc(expreMatrix, metadata = NULL, factor, control)

## S4 method for signature 'matrix'
weight_ss_fc(expreMatrix, metadata = NULL, factor, control)

## S4 method for signature 'data.frame'
weight_ss_fc(expreMatrix, metadata = NULL, factor, control)

## S4 method for signature 'DGEList'
weight_ss_fc(expreMatrix, metadata = NULL, factor, control)

## S4 method for signature 'SummarizedExperiment'
weight_ss_fc(expreMatrix, metadata = NULL, factor, control)
```

Arguments

`expreMatrix`	`matrix` and `data.frame` of logCPM, or `DGEList`/`SummarizedExperiment` storing gene expression counts and sample metadata. Feature names need to be in entrez IDs, and column names need to be sample names
`metadata`	Sample metadata `data.frame` as described in the details section.
`factor`	`character` Factor defines how samples can be put into matching pairs (eg. patient).
`control`	`character` Treatment level that is the control.

Details

This function computes weighted single-sample logFCs from normalised logCPM values, used for computing single-sample perturbation scores.

Since genes with smaller logCPM turn to have larger variances among single sample logFCs.A lowess curve will be fitted to estimate the relationship between variances of single-sample logFCs and mean of logCPM, and the relationship will be used to estimate the variance of each mean logCPM value. Gene-wise weights, which are inverse of variances, will then be multiplied to single-sample logFCs to down-weight genes with low counts.

It is assumed that the genes with extremely low counts have been removed and the count matrix has been normalised prior to the logCPM matrix was derived. Row names of the matrix must be genes' entrez IDs.

If a S4 object of `DGEList` or `SummarizedExperiment` is provided as input to `expreMatrix`, the gene expression matrix will be extracted from it and converted to a logCPM matrix. Sample metadata will also be extracted from the same S4 object unless otherwise specified.

Provided sample metadata should have the same number of rows as the number of columns in the logCPM matrix. Metadata also must have a column called "sample" storing sample names (column names of logCPM matrix), and a column called "treatment" storing treatment of each sample.The control treatment level specified by the `control` parameter must exist in the treatment column.

This analysis was designed for experimental designs involving matched pairs of samples, such as when tissues collected from the same patient were treated with different treatments to study different treatment effects. Parameter `factor` tells the function how samples can be put into matching pairs. It must also be included as a column in the metadata.

Value

A list with two elements: $weight gene-wise weights; $logFC weighted single sample logFC matrix

Examples

```
# Inspect metadata data frame to make sure it has treatment, sample and patient columns
data(metadata_example)
data(logCPM_example)
length(setdiff(colnames(logCPM_example), metadata_example$sample)) == 0
ls <- weight_ss_fc(logCPM_example, metadata = metadata_example,
 factor = "patient", control = "Vehicle")
```

Index