Package: hipathia (via r-universe)

October 17, 2026

Title HiPathia: High-throughput Pathway Analysis

Version 3.13.0

Description Hipathia is a method for the computation of signal transduction along signaling pathways from transcriptomic data. The method is based on an iterative algorithm which is able to compute the signal intensity passing through the nodes of a network by taking into account the level of expression of each gene and the intensity of the signal arriving to it. It also provides a new approach to functional analysis allowing to compute the signal arriving to the functions annotated to each pathway.

Depends R (>= 4.1), igraph (>= 1.0.1), zen4R(>= 0.10.4), MultiAssayExperiment(>= 1.4.9), SummarizedExperiment(>= 1.8.1)

License GPL-2

Encoding UTF-8

Imports coin, stats, limma, grDevices, utils, graphics, preprocessCore, servr, DelayedArray, matrixStats, methods, S4Vectors, ggplot2, ggpubr, dplyr, tibble, visNetwork, reshape2, MetBrewer

RoxygenNote 7.3.3

Suggests BiocStyle, knitr, rmarkdown, testthat

VignetteBuilder knitr

biocViews Pathways, GraphAndNetwork, GeneExpression, GeneSignaling, GO

LazyData true

Config/pak/sysreqs cmake libglpk-dev make libicu-dev libsecret-1-dev libuv1-dev libxml2-dev libssl-dev zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-04-28 12:47:49 UTC

RemoteUrl https://github.com/bioc/hipathia

RemoteRef HEAD

RemoteSha 88814b548e2b16ce6cb4af96cc9a173e2e1eb257

Contents

annotate_paths . . . 3
brca . . . 4
brca_data . . . 4
brca_design . . . 5
comp . . . 6
create_report . . . 6
DAcomp . . . 7
DAdata . . . 9
DAoverview . . . 10
DAreport . . . 10
DAsummary . . . 12
DAtop . . . 13
define_colors . . . 13
do_pca . . . 14
do_wilcoxon . . . 15
exp_data . . . 16
get_file . . . 16
get_go_names . . . 17
get_highest_sig_ancestor . . . 18
get_node_names . . . 18
get_nodes_data . . . 19
get_path_names . . . 20
get_paths_data . . . 21
get_pathway_functions . . . 22
get_pathways_annotations . . . 22
get_pathways_list . . . 23
get_pathways_summary . . . 24
go_vals . . . 25
heatmap_plot . . . 25
hhead . . . 27
hidata . . . 28
hipathia . . . 28
igraphs_upgrade . . . 29
is_accepted_species . . . 30
load_annofuns . . . 30
load_annots . . . 31
load_entrez_hgnc . . . 31
load_gobp_frame . . . 32
load_gobp_net . . . 32
load_mgi . . . 32
load_pathways . . . 33
load_pseudo_mgi . . . 34
load_xref . . . 34
mgi_from_sif . . . 35
multiple_pca_plot . . . 35
node_color . . . 36

node_color_per_de . . . 38
normalize_data . . . 39
normalize_paths . . . 41
path_vals . . . 41
paths_to_go_ancestor . . . 42
pathway_comparison_plot . . . 43
pathways . . . 44
pca_plot . . . 45
plotVG . . . 46
quantify_terms . . . 47
results . . . 48
save_results . . . 49
top_pathways . . . 49
translate_data . . . 50
translate_matrix . . . 51
visualize_report . . . 51

Index **53**

`annotate_paths` *Annotates functions to pathways*

Description

Annotates functions from a database to each pathway

Usage

```
annotate_paths(metaginfo, dbannot)
```

Arguments

`metaginfo`	Pathways object
`dbannot`	Either a string indicating which precomputed annotation to use ("uniprot" for Uniprot Keywords or "GO" for Gene Ontology terms), or a dataframe with the annotation of the genes to the functions. First column are gene symbols, second column the functions.

Value

Object of annotations from pathways to functions

#@examples #pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320", #"hsa04012")) #annotate_paths(pathways, "GO")

#@export

brca *BRCA gene expression dataset as SummarizedExperiment*

Description

A dataset containing a matrix with the Gene expression of 40 samples from the BRCA-US project from The Cancer Genome Atlas (TCGA), and their experimental design, containing 20 "Tumor" samples 20 "Normal" samples.

Usage

```
data(brca)
```

Format

SummarizedExperiment. The assay is a matrix with 40 columns and 18638 rows. Row names are Entrez IDs and column names are the TCGA identifyers of the samples. The colData() is a data.frame with 1 column and 40 rows, including the experimental design of the 40 samples from the BRCA-US project from TCGA. Field `group` is the type of sample, either "Tumor" or "Normal".

Details

The gene expression matrix includes 40 samples. The data has been log-transformed and normalized with TMM.

Value

SummarizedExperiment including a matrix with 40 columns and 18638 rows. Row names are Entrez IDs and column names are the TCGA identifyers of the samples.

Source

https://cancergenome.nih.gov/

brca_data *BRCA gene expression dataset*

Description

Gene expression of 40 samples from the BRCA-US project from The Cancer Genome Atlas (TCGA).

Usage

```
data(brca_data)
```

Format

Matrix with 40 columns and 18638 rows. Row names are Entrez IDs and column names are the TCGA identifyers of the samples.

Details

Gene expression matrix with 40 samples taken from the BRCA-US project from The Cancer Genome Atlas (TCGA). The data has been log-transformed and normalized with TMM.

Value

Matrix with 40 columns and 18638 rows. Row names are Entrez IDs and column names are the TCGA identifyers of the samples.

Source

https://cancergenome.nih.gov/

---

brca_design *BRCA experimental design*

Description

Experimental design of the gene expression matrix `brca_data` with 40 samples taken from the BRCA-US project from The Cancer Genome Atlas (TCGA). 20 samples are "Tumor" samples and 20 samples are "Normal" samples.

Usage

```
data(brca_design)
```

Format

Dataframe with 1 column and 40 rows, including the experimental design of the 40 samples from the BRCA-US project from TCGA. Field `group` is the type of sample, either "Tumor" or "Normal".

Value

Dataframe with 1 column and 40 rows, including the experimental design of the 40 samples from the BRCA-US project from TCGA. Field `group` is the type of sample, either "Tumor" or "Normal".

Source

https://cancergenome.nih.gov/

comp *Wilcoxon comparison of pathways object*

Description

Comparison object returned by `hipathia::do_wilcoxon` function, after calling `comp <- do_wilcoxon(path_vals, sample_group, g1 = "Tumor", g2 = "Normal") path_names <- get_path_names(pathways, rownames(comp)) comp <- cbind(path_names, comp)`

Usage

```
data(comp)
```

Format

Table with 1868 rows and 5 columns

Value

Pathway comparison result

create_report *Create visualization HTML*

Description

Saves the results of a Wilcoxon comparison for the Hipathia pathway values into a folder, and creates a HTML from which to visualize the results on top of the pathways. The results are stored into the specified folder. If this folder does not exist, it will be created. The parent folder must exist.

Usage

```
create_report(
  comp,
  metaginfo,
  output_folder = NULL,
  path = NULL,
  node_colors = NULL,
  group_by = "pathway",
  conf = 0.05,
  verbose = FALSE
)
```

Arguments

comp	Comparison object as given by the do_wilcoxon function
metaginfo	Pathways object as returned by the load_pathways function
output_folder	Name of the folder in which the report will be stored.
path	Absolute path to the parent directory in which ‘output_folder‘ will be saved. If it is not provided, it will be created in a temp folder.
node_colors	List of colors with which to paint the nodes of the pathways, as returned by the node_color_per_de function. Default is white.
group_by	How to group the subpathways to be visualized. By default they are grouped by the pathway to which they belong. Available groupings include "uniprot", to group subpathways by their annotated Uniprot functions, "GO", to group subpathways by their annotated GO terms, and "genes", to group subpathways by the genes they include. Default is set to "pathway".
conf	Level of significance. By default 0.05.
verbose	Boolean, whether to show details about the results of the execution

Value

Saves the results and creates a report to visualize them through a server in the specified output_folder. Returns the folder where the report has been stored.

Examples

```
data(comp)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
report <- create_report(comp, pathways, "save_results")

## Not run:
data(results)
data(brca)
sample_group <- colData(brca)[,1]
colors_de <- node_color_per_de(results, pathways,
sample_group, "Tumor", "Normal")
report_colors <- create_report(comp, pathways, "save_results",
node_colors = colors_de)

## End(Not run)
```

DAcomp *Compares the gene expression, pathway activation level and the function activation level of the*

Description

Compares the gene expression, pathway activation level and the function activation level of the

Usage

```
DAcomp(
  hidata,
  groups,
  expdes,
  g2 = NULL,
  path.method = "wilcoxon",
  node.method = "limma",
  fun.method = "wilcoxon",
  order = FALSE,
  paired = FALSE,
  adjust = TRUE,
  conf.level = 0.05,
  sel_assay = 1
)
```

Arguments

hidata	Either a SummarizedExperiment object or a matrix, returned by function hipathia.
groups	Either a character indicating the name of the column in colData including the classes to compare, or a character vector with the class to which each sample belongs. Samples must be ordered as in hidata.
expdes	String, either an equation expression to pas to limma, or the label of the first group to be compared
g2	String, label of the second group to be compared, if not specified in expdes.
path.method	String, method to be used when comparing pathways. Options include wilcoxon (default, performs a Wilcoxon test comparing conditions expdes and g2 - in this case, mandatory parameter) and limma (performs a limma DE analysis using functions lmFit, contrasts.fit and eBayes using the formula in expdes or comparing conditions expdes and g2.
node.method	String, method to be used when comparing nodes. Options include wilcoxon (performs a Wilcoxon test comparing conditions expdes and g2 - in this case, mandatory parameter) and limma (default, performs a limma DE analysis using functions lmFit, contrasts.fit and eBayes using the formula in expdes or comparing conditions expdes and g2.
fun.method	String, method to be used when comparing functions. Options include wilcoxon (default, performs a Wilcoxon test comparing conditions expdes and g2 - in this case, mandatory parameter) and limma (performs a limma DE analysis using functions lmFit, contrasts.fit and eBayes using the formula in expdes or comparing conditions expdes and g2.
order	Boolean, whether to order the results table by the FDRp.value column. Default is FALSE.
paired	Boolean, whether the samples to be compared are paired. If TRUE, function wilcoxsign_test from package coin is used. If FALSE, function wilcox.test from package stats is used.

adjust	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method. Default is TRUE.
conf.level	Numeric, cut off for significance. Default is 0.05.
sel_assay	Character or integer, indicating the assay to be normalized in the SummarizedExperiment. Default is 1.

Value

List including comparison results for nodes, pathways and functions, if present.

Examples

```
data(hidata)
comp <- DAcomp(hidata, groups = "group", expdes = "Tumor", g2 = "Normal")
```

DAdata *Wilcoxon and limma comparison object for nodes, pathways and functional annotations*

Description

Comparison object returned by `hipathia::DAcomp` function, after calling `DAdata <- DAcomp(hidata, "group", g1 = "Tumor", g2 = "Normal")`

Usage

```
data(DAdata)
```

Format

List object with 4 entries: Nodes includes a matrix with 6826 rows and 8 columns Paths includes a matrix with 1876 rows and 13 columns Uni.terms includes a matrix with 142 rows and 6 columns GO.terms includes a matrix with 1654 rows and 6 columns

Value

List of tibbles with the comparison results

DAoverview *Table and plot of total number of altered and not altered nodes, paths and functions (Uniprot keywords and/or GO terms, if present).*

Description

Table and plot of total number of altered and not altered nodes, paths and functions (Uniprot keywords and/or GO terms, if present).

Usage

```
DAoverview(DAdata, conf.level = 0.05, adjust = TRUE, colors = "hiro")
```

Arguments

DAdata	List of comparison results, returned by function DAcomp.
conf.level	Numeric, cut off for significance. Default is 0.05.
adjust	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method. Default is TRUE.
colors	String with the color scheme or vector of colors to be used. See define_colors for available options. Default is "hiro".

Value

Plot and tibble including the number of total, altered, UP- and DOWN-regulated features for nodes, paths and functions if present.

Examples

```
data(DAdata)
DAoverview(DAdata)
```

DAreport *Create visualization HTML*

Description

Saves the results of a DAdata comparison for the Hipathia pathway values into a folder, and creates a HTML from which to visualize the results on top of the pathways. The results are stored into the specified folder. If this folder does not exist, it will be created. The parent folder must exist.

Usage

```
DAreport(
  DAdata,
  pathways,
  conf.level = 0.05,
  adjust = TRUE,
  group_by = "pathway",
  colors = "classic",
  output_folder = NULL,
  path = NULL,
  verbose = TRUE
)
```

Arguments

DAdata	List of comparison results, returned by function DAcomp.
pathways	Pathways object as returned by the load_pathways function
conf.level	Level of significance. By default 0.05.
adjust	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method. Default is TRUE.
group_by	How to group the subpathways to be visualized. By default they are grouped by the pathway to which they belong. Available groupings include "uniprot", to group subpathways by their annotated Uniprot functions, "GO", to group subpathways by their annotated GO terms, and "genes", to group subpathways by the genes they include. Default is set to "pathway".
colors	String with the color scheme or vector of colors to be used. See define_colors for available options. Default is "hiro".
output_folder	Name of the folder in which the report will be stored.
path	Absolute path to the parent directory in which ‘output_folder‘ will be saved. If it is not provided, it will be created in a temp folder.
verbose	Boolean, whether to show details about the results of the execution

Value

Saves the results and creates a report to visualize them through a server in the specified output_folder. Returns the folder where the report has been stored.

Examples

```
data(DAdata)
data(pathways)
DAreport(DAdata, pathways)
```

DAsummary *Lists and plots the top* n *altered pathways, taking into account the number of altered .*

Description

Lists and plots the top n altered pathways, taking into account the number of altered .

Usage

```
DAsummary(
  DAdata,
  n = 10,
  conf.level = 0.05,
  adjust = TRUE,
  ratio = FALSE,
  colors = "hiro",
  order.by = "number"
)
```

Arguments

DAdata	List of comparison results, returned by function DAcomp.
n	Number of top features to show.
conf.level	Numeric, cut off for significance. Default is 0.05.
adjust	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method. Default is TRUE.
ratio	Boolean, whether to plot the ratio of significant paths with respect to the total paths in the pathway. Default is FALSE.
colors	String with the color scheme or vector of colors to be used. See define_colors for available options. Default is "hiro".
order.by	String, how to order table of results. Available options include ratio (default, uses the ratio of significant paths with respect to the total paths in the pathway) and number (uses the number of significant paths in the pathway).

Value

Plot and tibble including top n altered pathways.

Examples

```
data(DAdata)
DAsummary(DAdata)
```

DAtop *Lists and plots the top* n *altered nodes, paths and functions (Uniprot keywords and/or GO terms, if present).*

Description

Lists and plots the top n altered nodes, paths and functions (Uniprot keywords and/or GO terms, if present).

Usage

```
DAtop(DAdata, n = 10, conf.level = 0.05, adjust = TRUE, colors = "hiro")
```

Arguments

DAdata	List of comparison results, returned by function DAcomp.
n	Number of top features to show.
conf.level	Numeric, cut off for significance. Default is 0.05.
adjust	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method. Default is TRUE.
colors	String with the color scheme or vector of colors to be used. See define_colors for available options. Default is "hiro".

Value

Plot and list of tables including top n altered features for nodes, paths and functions if present.

Examples

```
data(DAdata)
DAtop(DAdata)
```

define_colors *Color palettes to be used in plots.*

Description

Color palettes to be used in plots.

Usage

```
define_colors(colors, no.col = NULL)
```

Arguments

colors	String with the color scheme or vector of colors to be used. Available predefined options include: hipathia, classic, soft, okee, hiro, new, vg, orchid.
no.col	String with the color given to non-significant nodes, if not given in parameter colors.

Value

Plot and list of tables including top n altered features for nodes, paths and functions if present.

Examples

```
define_colors("hiro")
```

---

do_pca *Performs a Principal Components Analysis*

Description

Performs a Principal Components Analysis

Usage

```
do_pca(data, sel_assay = 1, cor = FALSE)
```

Arguments

data	SummarizedExperiment or matrix of values to be analyzed. Samples must be represented in the columns.
sel_assay	Character or integer, indicating the assay to be normalized in the SummarizedExperiment. Default is 1.
cor	A logical value indicating whether the calculation should use the correlation matrix or the covariance matrix. (The correlation matrix can only be used if there are no constant variables.)

Value

do_pca returns a list with class princomp.

Examples

```
data(path_vals)
pca_model <- do_pca(path_vals[seq_len(ncol(path_vals)),])
```

`do_wilcoxon` *Apply Wilcoxon test*

Description

Performs a Wilcoxon test for the values in `sel_vals` comparing conditions `g1` and `g2`

Usage

```
do_wilcoxon(
  data,
  group,
  g1,
  g2,
  paired = FALSE,
  adjust = TRUE,
  sel_assay = 1,
  order = FALSE
)
```

Arguments

`data`	Either a SummarizedExperiment object or a matrix, containing the values. Columns represent samples.
`group`	Either a character indicating the name of the column in colData including the classes to compare, or a character vector with the class to which each sample belongs. Samples must be ordered as in `data`
`g1`	String, label of the first group to be compared
`g2`	String, label of the second group to be compared
`paired`	Boolean, whether the samples to be compared are paired. If TRUE, function `wilcoxsign_test` from package `coin` is used. If FALSE, function `wilcox.test` from package `stats` is used.
`adjust`	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method
`sel_assay`	Character or integer, indicating the assay to be normalized in the SummarizedExperiment. Default is 1.
`order`	Boolean, whether to order the results table by the `FDRp.value` column. Default is FALSE.

Value

Dataframe with the result of the comparison

Examples

```
data(path_vals)
data(brca_design)
sample_group <- brca_design[colnames(path_vals),"group"]
comp <- do_wilcoxon(path_vals, sample_group, g1 = "Tumor", g2 = "Normal")
```

---

exp_data *Normalized BRCA gene expression dataset*

Description

Experimental design matrix once expression matrix `brca_data` has been translated to Entrez geens with `translate_matrix` and normalized using `normalize_data`.

Usage

```
data(exp_data)
```

Format

Matrix with 40 columns and 3184 rows. Row names are Entrez IDs and column names are the TCGA identifyers of the samples.

Details

To create the data, the following functions have been called: `trans_data <- translate_matrix(brca_data, "hsa") exp_data <- normalize_data(trans_data)`

Value

Matrix with 40 columns and 3184 rows. Row names are Entrez IDs and column names are the TCGA identifyers of the samples.

---

get_file *Loads a RData object and downloads it from Zenodo if necessary*

Description

Loads a RData object and downloads it from Zenodo if necessary

Usage

```
get_file(file)
```

Arguments

file	File name of the object to load #@examples #get_file("xref_hsa_v3.rda") #get_file("meta_graph_info_hsa_v3.rda")

Value

Object in the file

---

get_go_names *Tranlates GO IDs to GO names*

Description

Translates the GO IDs to readable and comprensible names.

Usage

```
get_go_names(names, species, maxchar = NULL, disambiguate = FALSE)
```

Arguments

names	Character vector with the GO IDs to be translated.
species	Species of the samples.
maxchar	Integer, describes the number of maximum characters to be shown. By default no filter is applied.
disambiguate	Boolean, whether to return unique strings by disambiguating with numbers.

Value

A character vector including the readable names of the GO IDs, in the same order as provided.

Examples

```
data(go_vals)
get_go_names(rownames(go_vals), "hsa")
```

get_highest_sig_ancestor *Get highest common GO ancestor of GO annotations*

Description

Get highest common GO ancestor of GO annotations

Usage

```
get_highest_sig_ancestor(
  go_terms,
  go_comp,
  metaginfo,
  unique = TRUE,
  pval = 0.05
)
```

Arguments

go_terms	GO terms for which the highest common ancestors are to be looked for.
go_comp	Wilcoxon comparison of the matrix of GO values as returned by do_wilcoxon.
metaginfo	Pathways object
unique	Boolean, whether to return only one highest significant GO ancestor or all of them. By default, TRUE.
pval	P-value cut-off. Default values is set to 0.05.

Value

highest common ancestors

#@export

get_node_names *Tranlates node IDs to node names*

Description

Translates the node IDs to readable and comprensible names.

The names of the nodes are encoded as "pathway: name", where "pathway" is the pathway to which the node belongs and "node" is the name of the node. Nodes may include more genes than the one depicted in the name.

Usage

```
get_node_names(metaginfo, names, maxchar = NULL)
```

Arguments

metaginfo	Pathways object
names	Character vector with the subpathway IDs to be translated
maxchar	Integer, describes the number of maximum characters to be shown. By default no filter is applied.

Value

A character vector including the readable names of the subpathways IDs, in the same order as provided.

Examples

```
data(results)
pathways_list <- c("hsa03320", "hsa04012")
pathways <- load_pathways(species = "hsa", pathways_list)
node_vals <- get_nodes_data(results)
translated_names <- get_node_names(pathways, rownames(node_vals))
```

---

get_nodes_data *Gets the object of node activation values*

Description

This function returns the object with the levels of activation of each node for each sample. Rows represent the nodes and columns represent the samples. Each cell is the value of activation of a node in a sample.

Rownames are the IDs of the nodes In order to transform IDs into readable names, use `get_node_names`.

Effector subpathways are subgraphs of a pathway including all the paths leading to an effector protein. Effector proteins are defined as final nodes in the graph. Each effector protein (final node) in a pathway defines its own effector subpathway as the nodes and edges in a path leading to it.

Decomposed subpathways are subgraphs of a pathway including all the paths starting in a receptor protein and ending in an effector protein. Receptor proteins are defined as initial nodes and effector proteins are defined as final nodes in the graph. Each effector subpathway can be decomposed in as many decomposed subpathways as initial nodes it includes.

Usage

```
get_nodes_data(results, matrix = FALSE)
```

Arguments

results	Results object as returned by hipathia.
matrix	Boolean, if TRUE the function returns a matrix object, if FALSE (as default) returns a SummarizedExperiment object.

Value

Object, either a SummarizedExperiment or a matrix, with the levels of activation of each decomposed subpathway for each sample.

Examples

```
data(results)
path_vals <- get_paths_data(results)
```

---

get_path_names *Tranlates path IDs to path names*

Description

Translates the subpathway IDs to readable and comprensible names.

For effector subpathways, the names of the subpathways are encoded as "pathway: effector_protein", where "pathway" is the pathway to which the subpathway belongs and "effector_protein" is the name of the last node in the subpathway.

For decomposed subpathways, the names of the subpathways are encoded as "pathway: receptor_protein - effector_protein", where "pathway" is the pathway to which the subpathway belongs, "receptor_protein" is the name of the initial node of the subpathway and "effector_protein" is the name of the last node in the subpathway.

Usage

```
get_path_names(metaginfo, names, maxchar = NULL)
```

Arguments

metaginfo	Pathways object
names	Character vector with the subpathway IDs to be translated
maxchar	Integer, describes the number of maximum characters to be shown. By default no filter is applied.

Value

A character vector including the readable names of the subpathways IDs, in the same order as provided.

Examples

```
data(path_vals)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
translated_names <- get_path_names(pathways, rownames(path_vals))
```

`get_paths_data` *Gets the object of subpathway activation values*

Description

This function returns the object with the levels of activation of each subpathway for each sample. Rows represent the subpathways and columns represent the samples. Each cell is the value of activation of a subpathway in a sample.

Rownames are the IDs of the subpathways. In order to transform IDs into readable names, use `get_path_names`.

Effector subpathways are subgraphs of a pathway including all the paths leading to an effector protein. Effector proteins are defined as final nodes in the graph. Each effector protein (final node) in a pathway defines its own effector subpathway as the nodes and edges in a path leading to it.

Decomposed subpathways are subgraphs of a pathway including all the paths starting in a receptor protein and ending in an effector protein. Receptor proteins are defined as initial nodes and effector proteins are defined as final nodes in the graph. Each effector subpathway can be decomposed in as many decomposed subpathways as initial nodes it includes.

Usage

```
get_paths_data(results, matrix = FALSE)
```

Arguments

`results`	Results object as returned by `hipathia`.
`matrix`	Boolean, if TRUE the function returns a matrix object, if FALSE (as default) returns a SummarizedExperiment object.

Value

Object, either a SummarizedExperiment or a matrix, with the levels of activation of each decomposed subpathway for each sample.

Examples

```
data(results)
path_vals <- get_paths_data(results)
```

get_pathway_functions *Returns functions related to a pathway*

Description

Returns functions related to a pathway

Usage

```
get_pathway_functions(
  pathigraph,
  dbannot,
  entrez2hgnc,
  use_last_nodes = TRUE,
  unique = TRUE
)
```

Arguments

pathigraph	Pathway object
dbannot	Dataframe with the annotation of the genes to the functions. First column are gene symbols, second column the functions.
entrez2hgnc	Relation between Entrez and HGNC genes.
use_last_nodes	Boolean, whether to annotate functions to the last nodes of the pathways or not. If FALSE, functions will refer to all the nodes of the pathway.
unique	Boolean, whether to return the first function for each path.

Value

List of annotations from pathways to functions

get_pathways_annotations *Get Pathways functional annotations*

Description

Get functional annotation of the pathways, either for a particular annotation or a stored one.

Usage

```
get_pathways_annotations(pathway_names, metaginfo, dbannot, collapse = FALSE)
```

Arguments

pathway_names	Character vector of the names of the pathways
metaginfo	Pathways object
dbannot	Either a string indicating which precomputed annotation to use ("uniprot" for Uniprot Keywords or "GO" for Gene Ontology terms), or a dataframe with the annotation of the genes to the functions. First column are gene symbols, second column the functions.
collapse	Boolean, whether to collapse all functions of the same path in a single character string.

Value

2-columns matrix with the annotations of each pathway ID in the annotation dbannot.

Examples

```
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
pathway_names <- c("P-hsa03320-37", "P-hsa03320-61", "P-hsa03320-46",
"P-hsa03320-57", "P-hsa03320-64", "P-hsa03320-47", "P-hsa03320-65")
## Not run: get_pathways_annotations(pathway_names, pathways, "GO")
get_pathways_annotations(pathway_names, pathways, "uniprot")
```

---

get_pathways_list *Lists the IDs of the pathways in a pathways object*

Description

Lists the IDs of the pathways included in the pathways object metaginfo

Usage

```
get_pathways_list(metaginfo)
```

Arguments

metaginfo	Pathways object

Value

List of the pathway IDs included in the pathways object

Examples

```
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
pathways_list <- get_pathways_list(pathways)
```

`get_pathways_summary` *Compute pathway summary*

Description

Computes a summary of the results, summarizing the number and proportion of up- and down-regulated subpathways in each pathway.

Usage

```
get_pathways_summary(comp, metaginfo, conf = 0.05)
```

Arguments

`comp`	Comparison data frame as returned by the `do_wilcoxon` function.
`metaginfo`	Pathways object
`conf`	Level of significance of the comparison for the adjusted p-value. Default is 0.05.

Value

Table with the summarized information for each of the pathways. Rows are the analized pathways. Columns are: * `num_total_paths` Number of total subpathways in which each pathway is decomposed. * `num_significant_paths` Number of significant subpathways in the provided comparison. * `percent_significant_paths` Percentage of significant subpathways from the total number of subpathways in a pathway. * `num_up_paths` Number of significant up-regulated subpathways in the provided comparison. * `percent_up_paths` Percentage of significant up-regulated subpathways from the total number of subpathways in a pathway. * `num_down_paths` Number of significant down-regulated subpathways in the provided comparison. * `percent_down_paths` Percentage of significant down-regulated subpathways from the total number of subpathways in a pathway.

Examples

```
data(comp)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
get_pathways_summary(comp, pathways)
```

go_vals *Gene Ontology matrix of the BRCA gene expression dataset*

Description

Matrix of Gene Ontology terms activation values for the BRCA dataset. This matrix is computed from the Results object returned by the `hipathia` function by means of the `quantify_terms` function.

Usage

```
data(go_vals)
```

Format

Matrix with 40 columns and 1654 rows. Row names are Gene Ontology terms and column names are the TCGA identifyers of the samples.

Details

```
go_vals <- quantify_terms(results, pathways, "GO")
```

Value

Matrix with 40 columns and 1654 rows. Row names are Gene Ontology terms and column names are the TCGA identifyers of the samples.

heatmap_plot *Plots subpathways heatmap*

Description

Plots a heatmap with the values of the subpathways.

Usage

```
heatmap_plot(
  data,
  group = NULL,
  sel_assay = 1,
  colors = "classic",
  sample_clust = TRUE,
  variable_clust = FALSE,
  labRow = NULL,
  labCol = NULL,
  sample_colors = NULL,
```

```
  scale = TRUE,
  save_png = NULL,
  legend = TRUE,
  legend_xy = "topright",
  pch = 15,
  main = NULL
)
```

Arguments

data	Either a SummarizedExperiment or a matrix with the values to be plotted. Rows are features and columns are samples.
group	Either a character indicating the name of the column in colData including the classes to plot, or a character vector with the class to which each sample belongs. Samples must be ordered as in data. By default, all samples will be assigned to the same class.
sel_assay	Character or integer, indicating the assay to be normalized in the SummarizedExperiment. Default is 1.
colors	Either a character vector with colors or a key name indicating the color scheme to be used in the heatmap. If a character vector is provided, it is recommended to provide at least 3 colors. Three different predefined color schemes may be selected by providing a key name. Options are: * classic Blue for lower values, white for medium values, red for higher values. * hipathia Hipathia predefined color scheme: Green for lower values, white for medium values, orange for higher values. * redgreen Green for lower values, black for medium values, red for higher values. By default classic color scheme is applied.
sample_clust	Boolean, whether to cluster samples (columns). By default TRUE.
variable_clust	Boolean, whether to cluster variables (rows). By default FALSE. If TRUE, rows with 0 variance are removed.
labRow, labCol	Character vectors with row and column labels to be used. By default rownames(data) or colnames(data) are used, respectively.
sample_colors	Named character vector of colors. The names of the colors must be the classes in group. Each sample will be assigned the color corresponding to its class, taken from the group vector. By default a color will be assigned automatically to each class.
scale	Boolean, whether to scale each row to the interval [0,1]. Default is TRUE.
save_png	Path to the file where the image as PNG will be saved. By default, the image is not saved.
legend	Boolean, whether to display a legend.
legend_xy	Position for the legend, in case legend is TRUE.
pch	Graphical parameter from par() function.
main	Main title of the image

Value

Heatmap of the values of the subpathways

Examples

```
data(brca_design)
data(path_vals)
sample_group <- brca_design[colnames(path_vals),"group"]
heatmap_plot(path_vals, group = sample_group)
heatmap_plot(path_vals, group = "group", colors = "hipathia",
variable_clust = TRUE)
```

---

hhead *Head function for SummarizedExperiment, data.frames and matrix objects*

Description

Shows the first `n` rows and the first `n` columns of a matrix, in case the matrix has more than `n+5` rows or columns. Otherwise, it shows all the rows or columns, respectively.

Usage

```
hhead(mat, n = 5, sel_assay = 1)
```

Arguments

`mat`	Object to be shown
`n`	Number of rows and columns
`sel_assay`	Character or integer, indicating the assay to be translated in the SummarizedExperiment. Default is 1.

Value

Matrix with as much as `n` rows and `n` columns.

Examples

```
mat <- matrix(rnorm(100), ncol = 10)
hhead(mat)
hhead(mat, 3)
hhead(mat, 7)
```

hidata *Results object*

Description

Results object returned by `hipathia::hipathia` function, after calling `hidata <- hipathia(brca, pathways, verbose=TRUE, uni.terms = TRUE,GO.terms = TRUE)`

Usage

```
data(hidata)
```

Format

MultiAssayExperiment object of 4 listed experiments, with the activity values of nodes, paths and functional annotations for each sample: Nodes includes a matrix with 6826 rows Paths includes a matrix with 1876 rows Uni.terms includes a matrix with 142 rows GO.terms includes a matrix with 1654 rows

Value

Object of results, including nodes, pathways and functional information.

hipathia *Computes the level of activation of the subpathways for each of the samples*

Description

#@importFrom igraph

Usage

```
hipathia(
  genes_vals,
  metaginfo,
  uni.terms = FALSE,
  GO.terms = FALSE,
  sel_assay = 1,
  decompose = FALSE,
  scale = TRUE,
  maxnum = 100,
  verbose = TRUE,
  tol = 1e-06,
  test = TRUE
)
```

Arguments

genes_vals	A SummarizedExperiment or matrix with the normalized expression values of the genes. Rows represent genes and columns represent samples. Rownames() must be accepted gene IDs.
metaginfo	Pathways object
uni.terms	Boolean, whether to compute functional analysis with Uniprot keywords.
GO.terms	Boolean, whether to compute functional analysis with Gene Ontology terms.
sel_assay	Character or integer, indicating the assay to be processed in the SummarizedExperiment. Only applied if genes_vals is a SummarizedExperiment.Default is 1.
decompose	Boolean, whether to compute the values for the decomposed subpathways. By default, effector subpathways are computed.
scale	Boolean, whether to scale the values matrix to [0,1]. Default is TRUE.
maxnum	Number of maximum iterations when iterating the signal through the loops into the pathways
verbose	Boolean, whether to show details about the results of the execution of hipathia
tol	Tolerance for the difference between two iterations when iterating the signal through the loops into the pathways
test	Boolean, whether to test the input objects. Default is TRUE.

Value

A MultiAssayExperiment object with the level of activation of the subpathways from the pathways in pathigraphs for the experiment with expression values in genes_vals.

Examples

```
data(exp_data)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
results <- hipathia(exp_data, pathways, verbose = TRUE)
## Not run: results <- hipathia(exp_data, pathways, decompose = TRUE,
verbose = FALSE)
## End(Not run)
```

---

igraphs_upgrade *Upgrade igraphs to current version*

Description

Upgrades the igraph objects in metaginfo object to the corresponding version of the igraph package.

Usage

```
igraphs_upgrade(metaginfo)
```

Arguments

metaginfo	Pathways object

Value

The pathways object with the upgraded igraph objects

---

is_accepted_species *Checks whether a species is accepted*

Description

Checks whether a species is accepted

Usage

```
is_accepted_species(species)
```

Arguments

species	Species of the samples. #@examples #is_accepted_species("hsa") #is_accepted_species("fca")

Value

Boolean, whether species is accepted or not.

---

load_annofuns *Loads annotations object*

Description

Loads annotations object

Usage

```
load_annofuns(db, species)
```

Arguments

db	Database to be used. Either "GO" or "uniprot".
species	Species of the samples. #@examples #load_annofuns("GO", "hsa") #load_annofuns("uniprot", "hsa")

Value

Annotations object

`load_annots` *Loads functional annotations to genes*

Description

Loads functional annotations from HGNC to the selected database.

Usage

```
load_annots(db, species)
```

Arguments

`db`	Database to be used. Either "GO" or "uniprot".
`species`	Species of the samples. #@examples #load_annots("GO", "hsa")

Value

Functional annotations from HGNC to the selected database.

`load_entrez_hgnc` *Loads table of translation from HGNC to Entrez*

Description

Loads table of translation from HGNC to Entrez

Usage

```
load_entrez_hgnc(species)
```

Arguments

`species`	Species of the samples. #@examples #load_entrez_hgnc("hsa")

Value

Table of translation from HGNC to Entrez

load_gobp_frame *Loads GO graph information*

Description

#@examples #load_gobp_frame()

Usage

```
load_gobp_frame()
```

Value

GO graph information

load_gobp_net *Loads GO graph*

Description

#@examples #load_gobp_net()

Usage

```
load_gobp_net()
```

Value

GO graph

load_mgi *Loads object with graph information*

Description

Loads object with graph information

Usage

```
load_mgi(species)
```

Arguments

species	Species of the samples. #@examples #load_mgi("hsa")

Value

Graph information object

`load_pathways` *Loads the pathways object.*

Description

Loads the pathways object, which includes information about the pathways to be analyzed.

Usage

```
load_pathways(species, pathways_list = NULL)
```

Arguments

`species`	Species of the samples.
`pathways_list`	Vector of the IDs of the pathways to load. By default all available pathways are load.

Details

The object of pathways includes information about the pathways and the subpathways which will be analyzed. This object must be provided to some of the functions (like `hipathia` or `quantify_terms`) in the package. These functions will analyze all the pathways included in this object. By default, all available pathways are load. In order to restrict the analysis to a predefined set of pathways, specify the set of pathways to load with the parameter `pathways_list`.

Value

An pathways object including * `species` Species to which the pathways are related. * `pathigraphs` List of Pathigraph objects. Each Pathigraph contains the necessary information of a pathway for it to be analyzed with `Hipathia`. * `all_genes` List of all the genes included in the selection of pathways stored in `pathigraphs`. * `eff_norm` Vector of normalization values for effector subpathways. * `path_norm` Vector of normalization values for decomposed subpathways.

Examples

```
## Not run: pathways <- load_pathways("hsa")   # Loads all pathways for human
pathways <- load_pathways("mmu", c("mmu03320", "mmu04024", "mmu05200"))
   # Loads pathways 03320, 04024 and 05200 for mouse
```

load_pseudo_mgi *Loads object with pseudo graph information*

Description

Loads object with pseudo graph information

Usage

```
load_pseudo_mgi(species, group_by)
```

Arguments

species	Species of the samples.
group_by	How to group the subpathways to be visualized. By default they are grouped by the pathway to which they belong. Available groupings include "uniprot", to group subpathways by their annotated Uniprot functions, "GO", to group subpathways by their annotated GO terms, and "genes", to group subpathways by the genes they include.
	#@examples #load_pseudo_mgi("hsa", "uniprot")

Value

Pseudo graph information object

load_xref *Loads table of references*

Description

Loads table of references

Usage

```
load_xref(species)
```

Arguments

species	Species of the samples.
	#@examples #load_xref("hsa")

Value

Table of references

mgi_from_sif *Create a Pathways object from SIF files*

Description

Creates a Pathways object from the information of a pathway stored in a SIF file with some attributes. This pathways object can be used by function `hipathia` to analyze data.

Usage

```
mgi_from_sif(sif.folder, spe, entrez_symbol = NULL, dbannot = NULL)
```

Arguments

`sif.folder`	Path to the folder in which SIF and ATT files are stored.
`spe`	Species
`entrez_symbol`	Relation between Entrez (NCBI) genes and gene symbols. Data.frame with 2 columns: First column is the EntrezGene ID, second column is the gene Symbol. The genes in the nodes of the pathways should be defined by Entrez IDs in the SIF and ATT files of the pathways. In order to be more readable, gene names are used when plotting the pathways.
`dbannot`	Functional annotation of the genes in the pathways to create function nodes.

Value

A pathways object with the same structure of that returned by function `load_pathways`.

multiple_pca_plot *Plots multiple components of a PCA*

Description

Plots multiple components of a PCA analysis computed with `do_pca`

Usage

```
multiple_pca_plot(
  fit,
  group = NULL,
  sample_colors = NULL,
  comps = seq_len(3),
  plot_variance = FALSE,
  legend = TRUE,
  cex = 2,
  pch = 20,
```

```
  main = "Multiple PCA plot",
  save_png = NULL
)
```

Arguments

fit	princomp object as returned by do_pca
group	Vector with the group to which each sample belongs. The samples must be ordered as in path_vals. By default, all samples will be assigned to the same class.
sample_colors	Named character vector of colors. The names of the colors must be the classes in group. Each sample will be assigned the color corresponding to its class, taken from the group vector. By default a color will be assigned automatically to each class.
comps	Vector with the components to be plot
plot_variance	Logical, whether to plot the cumulative variance.
legend	Boolean, whether to plot a legend in the plot. Default is TRUE.
cex	Graphical parameter from par() function.
pch	Graphical parameter from par() function.
main	Main title of the image
save_png	Path to the file where the image as PNG will be saved. By default, the image is not saved.

Value

Plots multiple components of a PCA

Examples

```
data(path_vals)
sample_group <- brca_design[colnames(path_vals),"group"]
pca_model <- do_pca(path_vals[seq_len(ncol(path_vals)),])
multiple_pca_plot(pca_model, sample_group, cex = 3, plot_variance = TRUE)
```

---

node_color *Get colors of the nodes from a comparison file*

Description

Computes the colors of the nodes depending on the sign and p.value from the provided file. Significant up- and down-regulated nodes are depicted with the selected color, with a gradient towards the non-significant color depending on the value of the p-value. Smaller p-values give rise to purer colors than higher p-values.

Usage

```
node_color(
  comp,
  metaginfo,
  group_by = "pathway",
  colors = "classic",
  conf = 0.05,
  adjust = TRUE
)
```

Arguments

comp	Comparison file as returned by do_wilcoxon. Must include a column named "UP/DOWN" with the sign of the comparison coded as UP or DOWN, a column named "p.value" of raw p.values and a column named "FDRp.value" of adjusted p.values.
metaginfo	Object of pathways.
group_by	How to group the subpathways to be visualized. By default they are grouped by the pathway to which they belong. Available groupings include "uniprot", to group subpathways by their annotated Uniprot functions, "GO", to group subpathways by their annotated GO terms, and "genes", to group subpathways by the genes they include. Default is set to "pathway".
colors	Either a character vector with 3 colors (indicating, in this order, down-regulation, non-significance and up-regulation colors) or a key name indicating the color scheme to be used. Options are:
conf	Level of significance of the comparison for the adjusted p-value.
adjust	Boolean, whether to adjust the p.value from the comparison. Default is TRUE.

Value

List of color vectors, named by the pathways to which they belong. The color vectors represent the differential expression of the nodes in each pathway.

Slots

`classic` ColorBrewer blue, white and colorBrewer red.

`hipathia` Hipathia predefined color scheme: Green, white and orange. By default `classic` color scheme is applied.

Examples

```
data(results)
data(brca)
pathways_list <- c("hsa03320", "hsa04012")
pathways <- load_pathways(species = "hsa", pathways_list)
comp <- do_wilcoxon(results[["nodes"]], "group", "Tumor", "Normal")
colors_de <- node_color(comp, pathways)
```

node_color_per_de *Colors of the nodes by its differential expression*

Description

Performs a Limma differential expression on the nodes and computes the colors of the nodes depending on it_ Significant up- and down-regulated nodes are depicted with the selected color, with a gradient towards the non-significant color depending on the value of the p-value. Smaller p-values give rise to purer colors than higher p-values.

Usage

```
node_color_per_de(
  results,
  metaginfo,
  group,
  expdes,
  g2 = NULL,
  group_by = "pathway",
  colors = "classic",
  conf = 0.05,
  adjust = TRUE
)
```

Arguments

results	Object of results as provided by the hipathia function_
metaginfo	Object of pathways_
group	Character indicating the column in which the group variable is stored, in case the object provided to hipathia was a SummarizedExperiment, or a vector with the class to which each sample belongs. Samples must be ordered as in results.
expdes	String, either the comparison to be performed or the label of the first group to be compared.
g2	String, label of the second group to be compared. Only necessary in case expdes is the name of the first group, not the comparison.
group_by	How to group the subpathways to be visualized. By default they are grouped by the pathway to which they belong. Available groupings include "uniprot", to group subpathways by their annotated Uniprot functions, "GO", to group subpathways by their annotated GO terms, and "genes", to group subpathways by the genes they include. Default is set to "pathway".
colors	Either a character vector with 3 colors (indicating, in this order, down-regulation, non-significance and up-regulation colors) or a key name indicating the color scheme to be used. Options are:
conf	Level of significance of the comparison for the adjusted p-value.
adjust	Boolean, whether to adjust the p.value from the comparison. Default is TRUE.

Value

List of color vectors, named by the pathways to which they belong. The color vectors represent the differential expression of the nodes in each pathway.

Slots

`classic` ColorBrewer blue, white and colorBrewer red.

`hipathia` Hipathia predefined color scheme: Green, white and orange. By default `classic` color scheme is applied.

Examples

```
data(results)
data(brca)
pathways_list <- c("hsa03320", "hsa04012")
pathways <- load_pathways(species = "hsa", pathways_list)
colors_de <- node_color_per_de(results, pathways, "group", "Tumor - Normal")
colors_de <- node_color_per_de(results, pathways, "group", "Tumor", "Normal")
```

---

normalize_data *Normalize expression data from a SummarizedExperiment or matrix to be used in* `hipathia`

Description

Transforms the rank of the SummarizedExperiment or matrix of gene expression to [0,1] in order to be processed by `hipathia`. The transformation may be performed in two different ways. If `percentil = FALSE`, the transformation is a re-scaling of the rank of the matrix. If `percentil = TRUE`, the transformation is performed assigning to each cell its percentil in the corresponding distribution. This option is recommended for distributions with very long tails.

Usage

```
normalize_data(
  data,
  sel_assay = 1,
  by_quantiles = FALSE,
  by_gene = FALSE,
  percentil = FALSE,
  truncation_percentil = NULL
)
```

Arguments

data	Either a SummarizedExperiment or a matrix of gene expression.
sel_assay	Character or integer, indicating the assay to be normalized in the SummarizedExperiment. Default is 1.
by_quantiles	Boolean, whether to normalize the data by quantiles. Default is FALSE.
by_gene	Boolean, whether to transform the rank of each row of the matrix to [0,1]. Default is FALSE.
percentil	Boolean, whether to take as value the percentil of each sample in the corresponding distribution.
truncation_percentil	Real number p in [0,1]. When provided, values beyond percentil p are truncated to the value of percentil p, and values beyond 1-p are truncated to percentil 1-p. By default no truncation is performed.

Details

This transformation may be applied either to the whole matrix (by setting `by_gene = FALSE`), which we strongly recommend, or to each of the rows (by setting `by_gene = TRUE`), allowing each gene to have its own scale.

A previous quantiles normalization may be applied by setting `by_quantiles = TRUE`. This is recommended for noisy data.

For distributions with extreme outlayer values, a percentil p may be given to the parameter `truncation_percentil`. When provided, values beyond percentil p are truncated to the value of percentil p, and values beyond 1-p are truncated to percentil 1-p. This step is performed before any other tranformation. By default no truncation is performed.

Value

Matrix of gene expression whose values are in [0,1].

Examples

```
data("brca_data")
trans_data <- translate_data(brca_data, "hsa")
exp_data <- normalize_data(trans_data)
exp_data <- normalize_data(trans_data, by_quantiles = TRUE,
truncation_percentil=0.95)
```

normalize_paths *Normalize the pathway matrix by rows*

Description

Due to the nature of the Hipathia method, the length of a pathway may influence its signal rank. In order to compare signal values among subpathways, we strongly recommend to normalize the matrix with this normalization.

Usage

```
normalize_paths(path_vals, metaginfo)
```

Arguments

path_vals	SummarizedExperiment or matrix of the pathway values
metaginfo	Pathways object

Details

This function removes the bias caused by the length of the subpathways by dividing by the value obtained from running the method with a basal value of 0.5 at each node.

Value

SummarizedExperiment or matrix of normalized pathway values, depending on the class of path_vals.

Examples

```
data(path_vals)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
path_normalized <- normalize_paths(path_vals, pathways)
```

path_vals *Pathways matrix of the BRCA gene expression dataset*

Description

Matrix of pathway activation values for the BRCA dataset. This matrix is extracted from the Results object returned by the hipathia function by means of the get_paths_matrix function.

Usage

```
data(path_vals)
```

Format

Matrix with 40 columns and 1868 rows. Row names are Pathway IDs and column names are the TCGA identifyers of the samples.

Details

```
path_vals <- get_paths_matrix(results)
```

Value

Matrix with 40 columns and 1868 rows. Row names are Pathway IDs and column names are the TCGA identifyers of the samples.

---

paths_to_go_ancestor *Create path results table with highest significant GO ancestors*

Description

Create table of results with the comparison of the paths together with the GO functional annotation and the highest significant GO ancestor (HSGOA).

Usage

```
paths_to_go_ancestor(pathways, comp_paths, comp_go, pval = 0.05)
```

Arguments

pathways	Pathways object
comp_paths	Wilcoxon comparison of the matrix of pathways values as returned by do_wilcoxon.
comp_go	Wilcoxon comparison of the matrix of GO values as returned by do_wilcoxon.
pval	P-value cut-off. Default values is set to 0.05.

Details

The table returns in each row: the name of a pathway and its Wilcoxon comparison information (direction, adjusted p-value), the GO term to which the path is related (not necessarily unique), the Wilcoxon comparison informationfor this GO (direction, adjusted p-value), the HSGOA of this GO and its Wilcoxon comparison information (direction, adjusted p-value).

The HSGOA is computed as the GO term with minimum level from all the significant (with respect to value pval) ancestors of a GO. The level of a GO term is computed as the number of nodes in the shortest path from this GO term to the term "GO:0008150". The ancestors of a node are defined as all the nodes from which a path can be defined from the ancestor to the node.

Value

Table of comparisons with Highest common ancestors

Examples

```
data(comp)
data(go_vals)
data(brca_design)
data(path_vals)
sample_group <- brca_design[colnames(path_vals),"group"]
comp_go <- do_wilcoxon(go_vals, sample_group, g1 = "Tumor", g2 = "Normal")
## Not run: pathways <- load_pathways(species = "hsa", pathways_list =
c("hsa03320", "hsa04012"))
table <- paths_to_go_ancestor(pathways, comp, comp_go)
## End(Not run)
```

`pathway_comparison_plot` *Plots pathway with colored significant paths*

Description

Plots the layout of a pathway, coloring the significant subpathways in different colors depending on whether they are significantly up- or down-regulated. Nodes may be also colored providing a suitable list of colors for each node. Function `node_color_per_de` assigns colors to the nodes depending on their differential expression.

Usage

```
pathway_comparison_plot(
  comp,
  metaginfo,
  pathway,
  conf = 0.05,
  node_colors = NULL,
  colors = "classic"
)
```

Arguments

`comp`	Comparison data frame as returned by the `do_wilcox` function.
`metaginfo`	Pathways object.
`pathway`	Name of the pathway to be plotted.
`conf`	Level of significance of the comparison for the adjusted p-value. Default is 0.05.
`node_colors`	List, named by the pathway name, including the color of each node for each pathway.
`colors`	Either a character vector with 3 colors (indicating, in this order, down-regulation, non-significance and up-regulation colors) or a key name indicating the color scheme to be used. Options are:

Value

Image in which a pathway is ploted. Edges are colored so that the UP- and DOWN-activated subpathways are identified.

Slots

`classic` ColorBrewer blue, white and colorBrewer red.

`hipathia` Hipathia predefined color scheme: Green, white and orange. By default `classic` color scheme is applied.

Examples

```
data(comp)
pathways_list <- c("hsa03320", "hsa04012")
pathways <- load_pathways(species = "hsa", pathways_list)
pathway_comparison_plot(comp, metaginfo = pathways, pathway = "hsa03320")

## Not run:
data(results)
data(brca)
colors_de <- node_color_per_de(results, pathways, group, "Tumor", "Normal")
pathway_comparison_plot(comp, metaginfo = pathways, pathway = "hsa04012",
node_colors = colors_de)

## End(Not run)
```

---

pathways *Pathways object including pathways has03320 and hsa04012.*

Description

Pathways object returned by `hipathia::load_pathways` function, after calling `pathways <- load_pathways(species = "hsa",pathways_list = c("hsa03320", "hsa04012"))`

Usage

```
data(pathways)
```

Format

Pathways object

Value

Pathways object including pathways has03320 and hsa04012.

pca_plot *Plots two components of a PCA*

Description

Plots two components of a PCA computed with do_pca

Usage

```
pca_plot(
  fit,
  group = NULL,
  sample_colors = NULL,
  cp1 = 1,
  cp2 = 2,
  legend = TRUE,
  legend_xy = "bottomleft",
  cex = 2,
  pch = 20,
  mgp = c(3, 1, 0),
  main = "PCA plot",
  save_png = NULL
)
```

Arguments

fit	princomp object as returned by do_pca
group	Vector with the group to which each sample belongs. The samples must be ordered as in rownames(fit$scores). By default, all samples will be assigned to the same class.
sample_colors	Named character vector of colors. The names of the colors must be the classes in group. Each sample will be assigned the color corresponding to its class, taken from the group vector. By default a color will be assigned automatically to each class.
cp1	Integer, number of the component in the X-axis. Default is 1, the first component.
cp2	Integer, number of the component in the Y-axis. Default is 2, the second component.
legend	Boolean, whether to plot a legend in the plot. Default is TRUE.
legend_xy	Situation of the legend in the plot. Available options are: "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center".
cex	Graphical parameter from par() function.
pch	Graphical parameter from par() function.
mgp	Graphical parameter from par() function.

main	Title of the graphics
save_png	Path to the file where the image as PNG will be saved. By default, the image is not saved.

Value

Plots two components of a PCA

Examples

```
data(path_vals)
sample_group <- brca_design[colnames(path_vals),"group"]
pca_model <- do_pca(path_vals[seq_len(ncol(path_vals)),])
pca_plot(pca_model, sample_group)
```

---

plotVG *Plots a pathway with or without the comparison information, using the visNetwork library.*

Description

Plots a pathway with or without the comparison information, using the visNetwork library.

Usage

```
plotVG(
  name,
  pathways,
  DAdata = NULL,
  colors = "hiro",
  conf = 0.05,
  adjust = TRUE,
  main = "Pathway",
  submain = "",
  no.col = "BlanchedAlmond",
  height = "800px"
)
```

Arguments

name	KEGG ID of the pathway to plot.
pathways	Pathways object.
DAdata	List of comparison results, returned by function DAcomp.
colors	String with the color scheme or vector of colors to be used. See define_colors for available options. Default is "hiro".

conf	Numeric, cut off for significance. Default is 0.05.
adjust	Boolean, whether to adjust the p.value with Benjamini-Hochberg FDR method. Default is TRUE.
main	Title of the plot.
submain	Subtitle of the plot.
no.col	String with the color given to non-significant nodes.
height	Height of the plot. Default is "800px".

Value

Plot of the pathway.

Examples

```
data(pathways)
plotVG("hsa03320", pathways)

data(DAdata)
plotVG("hsa04012", pathways, DAdata)
```

---

quantify_terms *Computes the level of activation of the functions related to the previously computed subpathways*

Description

Computes the level of activation of the functions related to the previously computed subpathways

Usage

```
quantify_terms(
  results,
  metaginfo,
  dbannot,
  out_matrix = FALSE,
  normalize = TRUE
)
```

Arguments

results	List of results as returned by the hipathia function
metaginfo	Pathways object
dbannot	Either a string indicating which precomputed annotation to use ("uniprot" for Uniprot Keywords or "GO" for Gene Ontology terms), or a dataframe with the annotation of the genes to the functions. First column are gene symbols, second column the functions.

out_matrix	Boolean, whther the output object should be a matrix object. Default is FALSE, returning a SummarizedExperiment object.
normalize	Boolean, whether to normalize the matrix of pathway values with normalize_paths before quantifying the signal. Due to the nature of the Hipathia method, in which the length of each pathway may alter its signal rank, we strongly recommend to perform this normalization. This normalization removes the bias. Default is set to TRUE.

Value

Matrix with the level of activation of the functions in dbannot

Examples

```
data(results)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
go_values <- quantify_terms(results, pathways, "GO")
uniprot_values <- quantify_terms(results, pathways, "uniprot")
```

---

results *Results object*

Description

Results object returned by hipathia::hipathia function, after calling results <- hipathia(exp_data, pathways, verbose=TRUE)

Usage

```
data(results)
```

Format

Object of results, including pathways information.

Value

Object of results, including pathways information.

save_results *Save results to folder*

Description

Saves results to a folder. In particular, it saves the matrix of subpathway values, a table with the results of the provided comparison, the accuracy of the results and the .SIF and attributes of the pathways.

Usage

```
save_results(results, comp, metaginfo, output_folder = NULL, path = NULL)
```

Arguments

results	Results object as returned by the hipathia function.
comp	Comparison as returned by the do_wilcoxon function.
metaginfo	Pathways object
output_folder	Name of the folder in which the results will be stored.
path	Absolute path to the parent directory in which 'output_folder' will be saved. If it is not provided, it will be created in a temp folder.

Value

Creates a folder in disk in which all the information to browse the pathway results is stored.

Examples

```
data(results)
data(comp)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
save_results(results, comp, pathways, "output_results")
```

top_pathways *Computes pathway significance*

Description

Performs a test for each pathway checking if the number of significant paths is significant, compared to not having any of the paths as significant.

Usage

```
top_pathways(comp)
```

Arguments

comp	Comparison data frame as returned by the do_wilcoxon function.

Value

Table with the names of the pathways and their p-value for the Fisher test comparing the proportion of significant subpaths vs. 0.

Examples

```
data(comp)
top_pathways(comp)
```

---

translate_data *Translation of the rownames IDs of a SummarizedExperiment to Entrez IDs.*

Description

Translates the IDs in the rownames of a SummarizedExperiment to Entrez IDs. For accepted IDs to be transformed see the DOCUMENTATION.

Usage

```
translate_data(data, species, sel_assay = 1, verbose = TRUE)
```

Arguments

data	Either a SummarizedExperiment object or a matrix of gene expression.
species	Species of the samples.
sel_assay	Character or integer, indicating the assay to be translated in the SummarizedExperiment. Default is 1.
verbose	Boolean, whether to show details about the results of the execution.

Value

Either a SummarizedExperiment or a matrix (depending on the input type) of gene expression with Entrez IDs as rownames.

Examples

```
data("brca_data")
trans_data <- translate_data(brca_data, "hsa")
```

translate_matrix *Translation of the rownames IDs of a matrix to Entrez IDs.*

Description

Translates the IDs in the rownames of a matrix to Entrez IDs. For accepted IDs to be transformed see the DOCUMENTATION.

Usage

```
translate_matrix(exp, species, verbose = TRUE)
```

Arguments

exp	Matrix of gene expression.
species	Species of the samples.
verbose	Boolean, whether to show details about the results of the execution.

Value

Matrix of gene expression with Entrez IDs as rownames.

visualize_report *Visualize a HiPathia report*

Description

Visualize a HiPathia report

Usage

```
visualize_report(output_folder, port = 4000)
```

Arguments

output_folder	Folder in which results to visualize are stored
port	Port to use

Value

The instructions to visualize a HiPathia report in a web browser

Examples

```
data(comp)
pathways <- load_pathways(species = "hsa", pathways_list = c("hsa03320",
"hsa04012"))
report <- create_report(comp, pathways, "save_results")
visualize_report(report)

## Not run:
data(results)
data(brca)
sample_group <- colData(brca)[,1]
colors_de <- node_color_per_de(results, pathways,
sample_group, "Tumor", "Normal")
report <- create_report(comp, pathways, "save_results",
node_colors = colors_de)
visualize_report(report)
visualize_report(report, port = 5000)

## End(Not run)
```

Index

∗ **datasets**
  brca, 4
  brca_data, 4
  brca_design, 5
  comp, 6
  DAdata, 9
  exp_data, 16
  go_vals, 25
  hidata, 28
  path_vals, 41
  pathways, 44
  results, 48

annotate_paths, 3

brca, 4
brca_data, 4
brca_design, 5

comp, 6
create_report, 6

DAcomp, 7
DAdata, 9
DAoverview, 10
DAreport, 10
DAsummary, 12
DAtop, 13
define_colors, 13
do_pca, 14
do_wilcoxon, 15

exp_data, 16

get_file, 16
get_go_names, 17
get_highest_sig_ancestor, 18
get_node_names, 18
get_nodes_data, 19
get_path_names, 20
get_paths_data, 21
get_pathway_functions, 22
get_pathways_annotations, 22
get_pathways_list, 23
get_pathways_summary, 24
go_vals, 25

heatmap_plot, 25
hhead, 27
hidata, 28
hipathia, 28

igraphs_upgrade, 29
is_accepted_species, 30

load_annofuns, 30
load_annots, 31
load_entrez_hgnc, 31
load_gobp_frame, 32
load_gobp_net, 32
load_mgi, 32
load_pathways, 33
load_pseudo_mgi, 34
load_xref, 34

mgi_from_sif, 35
multiple_pca_plot, 35

node_color, 36
node_color_per_de, 38
normalize_data, 39
normalize_paths, 41

path_vals, 41
paths_to_go_ancestor, 42
pathway_comparison_plot, 43
pathways, 44
pca_plot, 45
plotVG, 46

quantify_terms, 47

results, 48

save_results, 49

top_pathways, 49
translate_data, 50
translate_matrix, 51

visualize_report, 51