Package: geneSetSimplifyR (via r-universe)

November 23, 2026

Type Package

Title Simplify Gene-Set Enrichment Analysis

Version 0.99.0

Description Simplifies the results of gene-set enrichment analysis (GSEA) by clustering the enriched gene sets on the genes they share. Gene sets are embedded with latent semantic indexing (LSI) of their gene membership, and clustered by community detection on a shared nearest neighbor (SNN) graph. Each cluster is labeled with the two-word phrase that best describes its gene-set names, found by term frequency-inverse document frequency (TF-IDF) and gene-set centrality. Results from several conditions or analyses can be combined. The package contains several plotting functions so the user can quickly visualize the clustering output and draw biologically meaningful insights.

License MIT + file LICENSE

Copyright See file COPYRIGHTS for the code adapted from Signac.

Encoding UTF-8

Language en-US

URL https://github.com/IsoformAnalysisGroup/geneSetSimplifyR

BugReports https://github.com/IsoformAnalysisGroup/geneSetSimplifyR/issues

biocViews Software, GeneSetEnrichment, Pathways, Clustering, DimensionReduction, Visualization

Depends R (>= 4.4.0)

Imports cluster, clustree, dplyr (>= 1.1.0), fgsea, ggplot2, ggraph, grDevices, igraph, irlba, magrittr, Matrix, methods, pheatmap, Seurat (>= 5.0.0), SeuratObject (>= 5.0.0), stats, stringr (>= 1.5.0), tibble, tidyr, tidytext, utils, withr

Suggests BiocStyle, knitr, msigdbr (>= 10.0.0), rmarkdown, RSpectra, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/pak/sysreqs libfontconfig1-dev libfreetype6-dev libglpk-dev make libicu-dev libpng-dev libxml2-dev libssl-dev

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

Date/Publication 2026-11-23 14:10:02 UTC

RemoteUrl https://github.com/BiocStaging/geneSetSimplifyR

RemoteRef HEAD

RemoteSha 9c0ec0cf08286ce694da1186988142087e10d100

Contents

clusterGeneSets . . . 3
exampleEnrichment . . . 4
exampleGeneSets . . . 5
exampleGsList . . . 5
extractResolution . . . 6
extractTable . . . 6
geneSetSimplifyR . . . 7
getGeneSets . . . 9
getSNN . . . 10
gsList-class . . . 11
initializeList . . . 12
labelClusters . . . 13
parseFgsea . . . 14
parsePairedGSEA . . . 15
plotBarchart . . . 17
plotClusterLabels . . . 18
plotClustree . . . 19
plotDendrogram . . . 20
plotEnrichments . . . 21
plotHeatmap . . . 22
plotProportions . . . 23
plotUMAP . . . 24
plotViolin . . . 26
snnToIgraph . . . 27
snnToMatrix . . . 28
stopWords . . . 28
updateClusterLabels . . . 29
updateResolution . . . 30

Index **31**

clusterGeneSets *Cluster gene sets*

Description

Cluster gene sets

Usage

```
clusterGeneSets(
  geneSetsList,
  resolution = c(0.1, 0.3, 0.6, 1.1, 1.6, 2.1, 2.6, 3.1, 4.1, 5.1),
  K = NA,
  verbose = TRUE
)
```

Arguments

geneSetsList	'gsList' object created with the 'initializeList()' function.
resolution	Vector with the desired clustering resolutions to use. Higher number usually results in more clusters.
K	Integer or NA. Number of nearest neighbors used when building the KNN graph. If NA (default) it is set to round(sqrt(N)), where N is the number of gene sets. Must be smaller than N.
verbose	a logic indicating whether to print progress statements. Defaults to TRUE.

Details

More than 50 gene sets are needed. The LSI uses up to 50 dimensions (fewer when there are fewer genes); the first one, which mostly reflects gene-set size, is always left out. The gene sets are clustered with the Louvain algorithm (Seurat::FindClusters()) on the shared nearest neighbor (SNN) graph; see getSNN() to get the graph.

Clustering does not change the random number stream of the R session: the UMAP restores it afterwards. Re-clustering an object removes its existing labels, so run labelClusters() again afterwards.

Value

'gsList' object with clustering stored. The best resolution (highest silhouette score) is annotated and used throughout the rest of the workflow (unless specified in subsequent functions).

Examples

```
data(exampleGeneSets, exampleEnrichment)
geneSetsList <- initializeList(
    geneSetsList = exampleGeneSets,
    geneSetsDF = exampleEnrichment,
```

```
    verbose = FALSE
)
gsListObject <- clusterGeneSets(
    geneSetsList = geneSetsList,
    resolution = c(0.6, 1.1),
    verbose = FALSE
)
```

`exampleEnrichment` *Example Enrichment Results*

Description

fgsea results for the TCGA colon adenocarcinoma data analyzed in Vitting-Seerup & Sandelin (2017), where gene-set enrichment was done both for differentially expressed genes ("Expression") and differentially spliced genes ("Splicing"). Only gene sets with an adjusted p-value below 0.05 are included, a subset of 196 gene sets.

Usage

```
exampleEnrichment
```

Format

A tibble with 205 rows (one per gene set and source) and 5 columns:

source The analysis the gene set was found in: "Expression" or "Splicing".

pathway The gene set name.

enrichment_score Normalized enrichment score (NES) from fgsea.

pval p-value from fgsea.

padj Benjamini-Hochberg adjusted p-value from fgsea.

Source

Computed with `fgsea::fgseaMultilevel()` and MSigDB v7.5.1 gene sets on the TCGA results of Vitting-Seerup K, Sandelin A. The Landscape of Isoform Switches in Human Cancers. Mol Cancer Res. 2017;15(9):1206-1220. doi:10.1158/15417786.MCR160459. See `inst/scripts/make-exampleEnrichment.R`.

exampleGeneSets *Example Gene Sets*

Description

The 196 gene sets that are significant in `exampleEnrichment`, as a named list where the names are the gene-set names and each element holds the gene symbols of that gene set.

Usage

```
exampleGeneSets
```

Format

A named list of 196 character vectors of human gene symbols.

Source

Human gene sets from the Molecular Signatures Database (MSigDB) v7.5.1 (R package msigdbr 7.5.1): the Hallmark collection, the C2 curated collection (chemical and genetic perturbations, Reactome and WikiPathways) and the C5 Gene Ontology collection. Each gene set is restricted to the genes tested in the enrichment analysis of `exampleEnrichment`. KEGG gene sets are left out because of their license. See `inst/scripts/make-exampleGeneSets.R`. https://www.gsea-msigdb.org/gsea/msigdb

exampleGsList *Example clustered and labeled gene sets*

Description

A 'gsList' object made with `geneSetSimplifyR(exampleGeneSets, exampleEnrichment, removeFirstWord = TRUE)`: 196 gene sets from 2 sources ("Expression" and "Splicing") clustered at 10 resolutions. Use it to try the plotting and table functions without running the workflow. The script that creates it is in `inst/scripts/make-exampleGsList.R`.

Usage

```
exampleGsList
```

Format

A 'gsList' object (see gsList).

Source

Made from `exampleGeneSets` and `exampleEnrichment`.

extractResolution *Extract the resolution currently used by default*

Description

Returns the resolution that the plotting and table functions use when their `resolution` argument is `NULL`. This is the optimal resolution found by `clusterGeneSets()`, unless it was changed with `updateResolution()` or `updateClusterLabels(resolution = ...)`.

Usage

```
extractResolution(geneSetsList)
```

Arguments

`geneSetsList`	'gsList' object that has been clustered with the 'clusterGeneSets()' function.

Value

The default resolution, as a number.

Examples

```
data(exampleGsList)
extractResolution(exampleGsList)
```

extractTable *Extract summary table of clustering*

Description

Extract summary table of clustering

Usage

```
extractTable(
  geneSetsList,
  resolution = NULL,
  summarize = TRUE,
  removeClusterId = TRUE
)
```

Arguments

geneSetsList	'gsList' object with clusters and labels.
resolution	Numeric. Resolution to extract the table for. Default NULL uses the active resolution: the optimal resolution found by clusterGeneSets(), or the one last set with updateResolution() or updateClusterLabels().
summarize	Logical indicating whether to summarize the table by cluster (and source). Default is TRUE.
removeClusterId	Logical, if TRUE the cluster number will be removed from each cluster label. Default: TRUE.

Value

A data.frame. With summarize = TRUE, one row per cluster and source with the mean and standard deviation of the enrichment score (ignoring missing values) and the number of gene sets. With summarize = FALSE, one row per gene set and source with all columns of the 'enrichmentScore' slot plus its cluster; if a cluster label was changed with updateClusterLabels(), the original label is given in initial_cluster_label.

Examples

```
data(exampleGsList)
extractTable(exampleGsList)
head(extractTable(exampleGsList, resolution = 0.6, summarize = FALSE))
```

---

geneSetSimplifyR *Run the geneSetSimplifyR workflow*

Description

Wrapper function that runs the three core functions of the 'geneSetSimplifyR' workflow with default parameters.

To read more about the individual functions see their corresponding help pages: initializeList, clusterGeneSets, labelClusters

Usage

```
geneSetSimplifyR(
  geneSetsList,
  geneSetsDF = NULL,
  resolution = c(0.1, 0.3, 0.6, 1.1, 1.6, 2.1, 2.6, 3.1, 4.1, 5.1),
  verbose = TRUE,
  K = NA,
  removeFirstWord = FALSE,
  stopWords = geneSetSimplifyR::stopWords,
  nLabels = 10
)
```

Arguments

geneSetsList	A list containing the gene sets to be analyzed. The name of each element in the list should be a gene set and each element should contain the genes belonging to that gene set. Alternatively, the path to a GMT file with the gene sets.
geneSetsDF	Optional data.frame with gene sets and enrichment scores that can be useful for downstream visualization (both for effect size and direction and if multiple conditions are analyzed). Should have a column named 'pathway' containing the gene sets and a column named 'enrichment_score' (or 'NES') with the enrichment scores from GSEA analysis for each gene set. Optionally have a column named 'pval' (or 'pvalue') with the p-value of each gene set, and a column named 'source' indicating the source of each gene set. This allows for analysis of multi-conditional data throughout the entire workflow and is also supported by many plotting functions. See `initializeList` and the vignette for more info.
resolution	Vector with the desired clustering resolutions to use. Higher number usually results in more clusters.
verbose	a logic indicating whether to print progress statements. Defaults to TRUE.
K	Number of nearest neighbors for the KNN graph, passed to `clusterGeneSets()`. `NA` (default) uses `round(sqrt(N))` for `N` gene sets.
removeFirstWord	Logical. If TRUE, strip the first word (everything before the first underscore) from each gene-set name before generating cluster labels. Useful for MSigDB-style names where the first token is an author or database prefix (e.g. "GOBP", "GOMF", "HALLMARK"). Default: FALSE.
stopWords	Words ignored when labeling clusters, passed to `labelClusters()`. Defaults to the shipped `stopWords`.
nLabels	Number of top labels to keep per cluster, passed to `labelClusters()`. Default 10.

Value

'gsList' object with labeled clusters for all clustered resolutions.

Examples

```
data(exampleGeneSets, exampleEnrichment)
gsListObject <- geneSetSimplifyR(
    geneSetsList = exampleGeneSets,
    geneSetsDF = exampleEnrichment,
    verbose = FALSE
)
gsListObject
```

getGeneSets *Fetch MSigDB gene sets by pathway name*

Description

Fetches gene-set memberships from MSigDB via `msigdbr::msigdbr()` and returns them in the named-list format expected by `geneSetSimplifyR()` (pathway name -> character vector of gene IDs).

Useful in two situations:

- Inspecting the gene members of specific pathways (e.g. the top pathways from a cluster after running the workflow).
- Building a `geneSetsList` from scratch without going through `pairedGSEA::prepare_msigdb()`.

Usage

```
getGeneSets(
  pathway_names = NULL,
  species = "Homo sapiens",
  collection = "C5",
  subcollection = NULL,
  gene_id_type = c("ensembl_gene", "gene_symbol", "entrez_gene")
)
```

Arguments

`pathway_names`	Character vector of MSigDB pathway names (i.e. `gs_name` values). If `NULL` (default), all pathways in the requested collection/subcollection are returned.
`species`	Species, passed to `msigdbr::msigdbr()`. Default `"Homo sapiens"`.
`collection`	MSigDB collection (e.g. `"C5"`, `"H"`, `"C2"`). Default `"C5"`.
`subcollection`	Optional MSigDB sub-collection (e.g. `"GO:BP"`). `NULL` (default) returns the full collection.
`gene_id_type`	Which gene identifier to use for the returned vectors. One of `"ensembl_gene"` (default), `"gene_symbol"`, or `"entrez_gene"`.

Value

A named list where each element is a character vector of gene IDs for a pathway. Names are pathway names (`gs_name`).

Examples

```
if (requireNamespace("msigdbr", quietly = TRUE)) {
    # Hallmark gene sets (gene symbols)
    hallmark <- getGeneSets(collection = "H", gene_id_type = "gene_symbol")
```

```
    # Just the genes of two specific pathways
    twoSets <- getGeneSets(
        pathway_names = c("HALLMARK_APOPTOSIS", "HALLMARK_HYPOXIA"),
        collection = "H",
        gene_id_type = "gene_symbol"
    )
    lengths(twoSets)
}
```

---

getSNN *Get the SNN graph of a clustered gsList*

Description

Returns the shared nearest neighbor (SNN) graph that `clusterGeneSets()` built and clustered. Convert it with `snnToIgraph()` or `snnToMatrix()` to analyze it with other tools.

Usage

```
getSNN(geneSetsList)
```

Arguments

`geneSetsList`	A 'gsList' object clustered with `clusterGeneSets()`.

Value

The SNN graph: a sparse symmetric matrix (a Seurat `Graph`) with the gene sets as rows and columns.

See Also

`snnToIgraph()`, `snnToMatrix()`

Examples

```
data(exampleGsList)
snn <- getSNN(exampleGsList)
dim(snn)
```

gsList-class *The gsList class*

Description

A gsList holds the input and results of the geneSetSimplifyR workflow. It is created by initializeList(), filled by clusterGeneSets() and labelClusters(), or all at once by geneSetSimplifyR(). Use extractTable() to get the results as a table.

Usage

```
## S4 method for signature 'gsList'
show(object)
```

Arguments

object	A 'gsList' object.

Value

gsList objects are created by initializeList(), not by calling the class generator directly. show() prints a summary of the object and returns it invisibly.

Functions

- show(gsList): Print a short summary of the object.

Slots

genesetsDataframe A data.frame with genes as rows and gene sets as columns; 1 if the gene is in the gene set, else 0.

enrichmentScore A data.frame with one row per gene set and source: pathway, source, enrichment_score, pval, nlog10pval and any other column supplied to initializeList().

cluster A Seurat object with the LSI, UMAP, SNN graph and the cluster of every gene set for each resolution (GSEA_snn_res.<resolution> metadata columns).

labels A list with, for each resolution, the cluster labels (labelsDefault), the TF-IDF scores of the label candidates (labelsTFIDF) and the most central gene sets (labelsCentrality).

metadata A list with parameters and results of the workflow, e.g. the number of nearest neighbors, the chosen resolution and silhouette scores.

Examples

```
data(exampleGsList)
exampleGsList
```

`initializeList` *Initialize gene sets list*

Description

Initialize a 'gsList' object from a dataframe with gene sets and enrichment scores and a gene sets database.

Usage

```
initializeList(geneSetsList, geneSetsDF = NULL, verbose = TRUE)
```

Arguments

`geneSetsList`	A list containing the gene sets to be analyzed. The name of each element in the list should be a gene set and each element should contain the genes belonging to that gene set. Alternatively, the path to a GMT file with the gene sets (read with `fgsea::gmtPathways()`).
`geneSetsDF`	Optional data.frame with gene sets and enrichment scores that can be useful for downstream visualization (both for effect size and direction and if multiple conditions are analyzed). Should have a column named 'pathway' containing the gene sets and a column named 'enrichment_score' or 'NES' with the enrichment of each gene set. Optionally have a column named 'pval' or 'pvalue' with the p-value of each gene set; a 'nlog10pval' column with the -log10 transformed p-values is added automatically and is used by default by the plotting functions that summarize scores (e.g. 'plotHeatmap()'). p-values must lie between 0 and 1. A p-value of 0 is transformed as the smallest positive double (giving about 307.7) rather than infinity. Optionally have a column named 'source' indicating the source of each gene set. This allows for analysis of multi-conditional data throughout the entire workflow and is also supported by many plotting functions. If a gene-set was found in multiple conditions there should be multiple rows for the same 'pathway' indicating its respective source and corresponding enrichment_score. Each combination of 'pathway' and 'source' may appear only once. Any other numeric column is kept and can be plotted via the 'plotFrom' argument of the plotting functions. List columns (e.g. 'leadingEdge' from fgsea) are dropped. See the vignette for examples.
`verbose`	a logic indicating whether to print progress statements. Defaults to TRUE.

Details

Gene sets without any gene are removed with a warning. Missing gene ids (`NA` or empty strings) are ignored.

Value

'gsList' object with a gene sets dataframe and enrichment scores for each gene set.

Examples

```
data(exampleGeneSets, exampleEnrichment)
gsListObject <- initializeList(
    geneSetsList = exampleGeneSets,
    geneSetsDF = exampleEnrichment
)
gsListObject

# Gene sets can also be read from a GMT file
gmtFile <- tempfile(fileext = ".gmt")
writeLines(
    vapply(names(exampleGeneSets), function(name) {
        paste(c(name, "", exampleGeneSets[[name]]), collapse = "\t")
    }, character(1)),
    gmtFile
)
gsListObject <- initializeList(gmtFile, exampleEnrichment, verbose = FALSE)
```

---

labelClusters *Label clusters in clustered gene sets list object*

Description

Label clusters in clustered gene sets list object

Usage

```
labelClusters(
  geneSetsList,
  stopWords = geneSetSimplifyR::stopWords,
  nLabels = 10,
  removeFirstWord = FALSE,
  verbose = TRUE
)
```

Arguments

geneSetsList	'gsList' object that has been clustered with the 'clusterGeneSets()' function.
stopWords	Vector with stop words. The default vector can be viewed with data(stopWords). To add new stop words 'stopWords <- c(geneSetSimplifyR::stopWords, 'new word 1', 'new word 2', ...)
nLabels	Number of top labels to keep. Default to 10
removeFirstWord	Logical. If TRUE, strip the first word (everything before the first underscore) from each gene-set name before generating bigram tokens. Useful for MSigDB-style names where the first token is an author or database prefix (e.g. "GOBP", "GOMF", "HALLMARK"). Default: FALSE.
verbose	a logic indicating whether to print progress statements. Defaults to TRUE.

Value

'gsList' object with labeled clusters for all clustered resolutions.

Examples

```
data(exampleGsList)
# Relabel the shipped example, e.g. with extra stop words
gsListObject <- labelClusters(
    geneSetsList = exampleGsList,
    stopWords = c(stopWords, "regulation"),
    verbose = FALSE
)
```

`parseFgsea` *Parse fgsea results into geneSetSimplifyR input*

Description

Converts `fgsea::fgsea()` output into the two inputs `geneSetSimplifyR()` needs: `geneSetsList` and `geneSetsDF` (with `source`, `pathway`, `enrichment_score`). To combine several fgsea runs into one multi-source analysis, call this function once per run with a distinct `source` label and `rbind()` the resulting `geneSetsDF`s.

Usage

```
parseFgsea(fgseaResult, geneSets, cutoff = 0.05, source = "fgsea")
```

Arguments

`fgseaResult`	Output of `fgsea::fgsea()` or `fgsea::fora()`. Must contain `pathway` and `padj` columns, and either `NES` (GSEA) or `foldEnrichment` (over-representation analysis, used as `log2(foldEnrichment)`).
`geneSets`	Named list of gene-set memberships (the same one passed to `fgsea::fgsea()`).
`cutoff`	padj threshold for filtering pathways. Default 0.05.
`source`	Label written to the `source` column of `geneSetsDF`. Use a distinct value per run when combining multiple analyses. Default `"fgsea"`.

Value

A list with:

`geneSetsList` `geneSets` filtered to pathways passing the cutoff.

`geneSetsDF` Long-format data frame: `source`, `pathway`, `enrichment_score` (fgsea's NES), `pval` (when present) and `padj`. One row per pathway.

Examples

```
data(exampleGeneSets)
# A small table in the format of fgsea::fgsea() output
pathways <- names(exampleGeneSets)[1:90]
fgseaRes <- data.frame(
    pathway = pathways,
    pval = rep(c(0.001, 0.01, 0.3), 30),
    padj = rep(c(0.004, 0.03, 0.5), 30),
    NES = seq(-2.5, 2.5, length.out = 90)
)
parsed <- parseFgsea(
    fgseaResult = fgseaRes,
    geneSets    = exampleGeneSets,
    cutoff      = 0.05,
    source      = "treatmentA"
)
head(parsed$geneSetsDF)
gss <- geneSetSimplifyR(
    geneSetsList = parsed$geneSetsList,
    geneSetsDF   = parsed$geneSetsDF,
    resolution   = 1.1,
    verbose      = FALSE
)
```

---

parsePairedGSEA *Parse pairedGSEA results into geneSetSimplifyR input*

Description

Converts `pairedGSEA::paired_ora()` output into the two inputs `geneSetSimplifyR()` needs: `geneSetsList` and `geneSetsDF` (with `source`, `pathway`, `enrichment_score`). Each of the three blocks (expression, splicing, paired) is filtered by its own padj cutoff and added as a separate `source`. Pathways passing multiple cutoffs appear once per source.

Usage

```
parsePairedGSEA(
  oraResult,
  geneSets,
  cutoff_exp = 0.05,
  cutoff_splicing = 0.05,
  cutoff_paired = 0.05
)
```

Arguments

`oraResult`	Data frame from `pairedGSEA::paired_ora()`. Needs a `pathway` column plus `padj_<block>` and `enrichment_score_<block>` columns for each enabled block.

`geneSets` Named list of gene-set memberships (the same one passed to `pairedGSEA::paired_ora()`).

`cutoff_exp` padj cutoff for the expression block, or `NULL` to skip using expression. Default 0.05.

`cutoff_splicing` padj cutoff for the splicing block, or `NULL` to skip using splicing. Default 0.05.

`cutoff_paired` padj cutoff for the paired block, or `NULL` to skip it. Default 0.05.

Value

A list with:

`geneSetsList` `geneSets` filtered to pathways passing at least one cutoff.

`geneSetsDF` Long-format data frame: `source`, `pathway`, `enrichment_score`, `padj` and `pval` (when `oraResult` has `pval_<block>` columns). One row per (pathway, source).

Examples

```
data(exampleGeneSets)
# A small table in the format of pairedGSEA::paired_ora() output
pathways <- names(exampleGeneSets)[1:90]
oraResult <- data.frame(
    pathway = pathways,
    padj_expression = rep(c(0.01, 0.2), 45),
    enrichment_score_expression = seq(-2, 2, length.out = 90),
    padj_splicing = rep(c(0.2, 0.01, 0.01), 30),
    enrichment_score_splicing = seq(2, -2, length.out = 90),
    padj_paired = 1,
    enrichment_score_paired = 0
)
parsed <- parsePairedGSEA(
    oraResult       = oraResult,
    geneSets        = exampleGeneSets,
    cutoff_exp      = 0.05,
    cutoff_splicing = 0.05,
    cutoff_paired   = NULL
)
table(parsed$geneSetsDF$source)
gss <- geneSetSimplifyR(
    geneSetsList = parsed$geneSetsList,
    geneSetsDF   = parsed$geneSetsDF,
    resolution   = 1.1,
    verbose      = FALSE
)
```

plotBarchart *Plot clusters as a bar chart*

Description

Creates a bar chart showing the number of gene sets in each cluster, per source when there are several.

Usage

```
plotBarchart(
  geneSetsList,
  resolution = NULL,
  removeClusterId = FALSE,
  highlightClusterNo = NULL
)
```

Arguments

geneSetsList	'gsList' object with clusters and labels.
resolution	The cluster resolution to plot. Default NULL uses the active resolution: the optimal resolution found by clusterGeneSets(), or the one last set with updateResolution() or updateClusterLabels().
removeClusterId	Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.
highlightClusterNo	A vector with the cluster numbers of the clusters (at the resolution) to highlight (in bold) in the plot.

Value

A ggplot object.

Examples

```
data(exampleGsList)
plotBarchart(exampleGsList)
plotBarchart(exampleGsList, resolution = 0.6, highlightClusterNo = 1)
```

`plotClusterLabels` *Plot label information for a particular cluster*

Description

Shows why a cluster got its label: the top label candidates with their combined and TF-IDF scores, the most central gene sets, the most frequent words and genes, and the largest gene sets of the cluster. Labels are the original ones from `labelClusters()`, also after a rename with `updateClusterLabels()`.

Usage

```
plotClusterLabels(
  geneSetsList,
  clusterNumber,
  resolution = NULL,
  nLabels = 5,
  maxChar = 50
)
```

Arguments

`geneSetsList`	'gsList' object that has been clustered and labeled.
`clusterNumber`	Which cluster to plot, given by its cluster number.
`resolution`	The cluster resolution to plot. Default `NULL` uses the active resolution: the optimal resolution found by `clusterGeneSets()`, or the one last set with `updateResolution()` or `updateClusterLabels()`.
`nLabels`	The number of labels to plot. Default to 5
`maxChar`	The max number of characters of labels shown in plot. Longer labels will be truncated using "...". Defaults to 50

Value

a ggplot2 object.

Examples

```
data(exampleGsList)
plotClusterLabels(geneSetsList = exampleGsList, clusterNumber = 0)
```

plotClustree *Plot cluster tree*

Description

Graphs a clustering tree showing how the clusters of consecutive resolutions relate to each other: each node is a cluster (labeled with its top label), and edges show how many gene sets move from a cluster at one resolution to a cluster at the next.

Usage

```
plotClustree(
  geneSetsList,
  node_alpha = 1,
  node_text_size = 2.5,
  node_size_range = c(4, 20),
  edge_width = 0.7,
  node_text_angle = 30,
  resolutionsToPlot = NULL
)
```

Arguments

geneSetsList	'gsList' object with clusters and labels.
node_alpha	Numeric value indicating the alpha of nodes
node_text_size	Numeric value indicating node text size
node_size_range	Numeric vector with two values indicating the minimum and maximum point size of nodes, respectively
edge_width	Numeric value indicating the width of plotted edges
node_text_angle	Numeric value indicating the rotation of node labels
resolutionsToPlot	Resolutions to include in plot. Use NULL to plot all. Default is NULL

Value

A ggplot2 object

Examples

```
data(exampleGsList)
plotClustree(exampleGsList)
```

`plotDendrogram` *Plot dendrogram of cluster averages*

Description

A plot showing the hierarchical clustering of the cluster averages (dendrogram). A tool made for showing how clusters (not individual gene-sets) relate to each other. The distance between two clusters is the average SNN distance (1 - SNN similarity) between their gene sets, and clusters are joined by complete linkage.

Usage

```
plotDendrogram(
  geneSetsList,
  resolution = NULL,
  removeClusterId = FALSE,
  highlightClusterNo = NULL
)
```

Arguments

`geneSetsList`	'gsList' object with clusters and labels.
`resolution`	The cluster resolution to plot. Default `NULL` uses the active resolution: the optimal resolution found by `clusterGeneSets()`, or the one last set with `updateResolution()` or `updateClusterLabels()`.
`removeClusterId`	Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.
`highlightClusterNo`	A vector with the cluster numbers of the clusters (at the resolution) to highlight (in bold) in the plot.

Value

A ggplot object. The underlying `hclust` object is stored in its `hclust` attribute.

Examples

```
data(exampleGsList)
plotDendrogram(exampleGsList)
hc <- attr(plotDendrogram(exampleGsList), "hclust")
```

plotEnrichments *Plot enrichment as lollipop plot*

Description

Creates a lollipop plot showing the summarized enrichment and size of each gene-set cluster.

Usage

```
plotEnrichments(
  geneSetsList,
  resolution = NULL,
  plotFrom = c("nlog10pval", "enrichment_score"),
  summaryStat = c("median", "max", "mean"),
  removeClusterId = FALSE,
  highlightClusterNo = NULL
)
```

Arguments

geneSetsList	'gsList' object with clusters and labels.
resolution	The cluster resolution to plot. Default NULL uses the active resolution: the optimal resolution found by clusterGeneSets(), or the one last set with updateResolution() or updateClusterLabels().
plotFrom	Name of the column of the 'enrichmentScore' slot to visualize, e.g. 'nlog10pval' or 'enrichment_score'. Only numeric columns that are annotated (i.e. not all NA) can be plotted - which columns are available depends on what was supplied to 'geneSetsDF' in 'initializeList()'. Only one column is plotted; if several are supplied the first plottable one is used (the default therefore falls back to 'enrichment_score' when no p-values were supplied).
summaryStat	How the values of the gene sets within a cluster are summarized into the value shown for each cluster. One of 'median' (default), 'max' or 'mean'. Note that 'max' is the largest signed value, so for signed scores (e.g. NES) a strongly negative cluster will be summarized by its least negative gene set.
removeClusterId	Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.
highlightClusterNo	A vector with the cluster numbers of the clusters (at the resolution) to highlight (in bold) in the plot.

Value

A ggplot object.

Examples

```
data(exampleGsList)
plotEnrichments(exampleGsList)
plotEnrichments(exampleGsList, summaryStat = 'mean', highlightClusterNo = 0)
```

`plotHeatmap` *Plot enrichment as heatmap*

Description

Creates a heatmap summarizing the enrichment of each gene-set cluster in each source (useful for datasets with many data sources).

Usage

```
plotHeatmap(
  geneSetsList,
  resolution = NULL,
  plotFrom = c("nlog10pval", "enrichment_score"),
  summaryStat = c("median", "max", "mean"),
  removeClusterId = FALSE,
  highlightClusterNo = NULL,
  transpose = FALSE,
  cutree_rows = NA,
  cutree_cols = NA,
  drawPlot = TRUE,
  scaleCols = "auto"
)
```

Arguments

`geneSetsList`	'gsList' object with clusters and labels.
`resolution`	The cluster resolution to plot. Default `NULL` uses the active resolution: the optimal resolution found by `clusterGeneSets()`, or the one last set with `updateResolution()` or `updateClusterLabels()`.
`plotFrom`	Name of the column of the 'enrichmentScore' slot to visualize, e.g. 'nlog10pval' or 'enrichment_score'. Only numeric columns that are annotated (i.e. not all NA) can be plotted - which columns are available depends on what was supplied to 'geneSetsDF' in 'initializeList()'. Only one column is plotted; if several are supplied the first plottable one is used (the default therefore falls back to 'enrichment_score' when no p-values were supplied).
`summaryStat`	How the values of the gene sets within a cluster are summarized into the value shown in each heatmap cell. One of 'median' (default), 'max' or 'mean'. Note that 'max' is the largest signed value, so for signed scores (e.g. NES) a strongly negative cluster will be summarized by its least negative gene set.

removeClusterId
: Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.

highlightClusterNo
: A vector with the cluster numbers of the clusters (at the resolution) to highlight (in bold) in the plot.

transpose
: A logical indicating whether to transpose the heatmap (switch rows and cols)

cutree_rows
: pheatmap control parameter. Number of clusters the rows are divided into

cutree_cols
: pheatmap control parameter. Number of clusters the columns are divided into

drawPlot
: Whether to draw the plot. Default to TRUE.

scaleCols
: Whether to z-score each cluster across the sources before plotting. One of "auto" (default), TRUE or FALSE. "auto" scales only when there are at least three sources: with one source there is nothing to compare, and with two sources the z-scores can only be -0.71, 0 or 0.71. Scaling is always across sources, also when transpose = TRUE.

Details

Cells of clusters that have no gene set from a source are shown in gray. Rows and columns are ordered by hierarchical clustering when there are at least two sources (absent cells count as 0 for the clustering).

Value

The pheatmap object, invisibly. Use drawPlot = FALSE to get it without drawing, e.g. to arrange it with other plots.

Examples

```
data(exampleGsList)
plotHeatmap(exampleGsList)
plotHeatmap(
    exampleGsList,
    plotFrom = 'enrichment_score',
    summaryStat = 'mean'
)
```

---

plotProportions *Plot proportions of gene sets in each cluster for each source*

Description

Graphs a bar plot showing the proportion of gene sets within each cluster that come from each source in the analyzed data. A gene set found in several sources is counted once, as 'Both' (two sources) or 'Multiple' (three or more). Needs at least two sources.

Usage

```
plotProportions(
  geneSetsList,
  resolution = NULL,
  removeClusterId = FALSE,
  highlightClusterNo = NULL
)
```

Arguments

geneSetsList	'gsList' object with clusters and labels.
resolution	The cluster resolution to plot. Default NULL uses the active resolution: the optimal resolution found by clusterGeneSets(), or the one last set with updateResolution() or updateClusterLabels().
removeClusterId	Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.
highlightClusterNo	A vector with the cluster numbers of the clusters (at the resolution) to highlight (in bold) in the plot.

Value

A ggplot object.

Examples

```
data(exampleGsList)
plotProportions(exampleGsList)
```

---

plotUMAP *Scatter plot of UMAP dimensionality reduction*

Description

Graphs the output of the UMAP dimensionality reduction as a scatterplot using the 'DimPlot()' function from the Seurat package. Gene sets are colored and labeled by their cluster. When there is only one source in the data (default), each point will be shaped by the direction of the enrichment score of the gene set. When there are two sources in the data, each point will be shaped by the source indicating whether the gene set was found significant in either or both sources. With three or more sources, points are not shaped.

Usage

```
plotUMAP(
  geneSetsList,
  resolution = NULL,
  removeClusterId = FALSE,
  interactive = FALSE,
  highlightClusterNo = NULL,
  pointSize = 2,
  pointAlpha = 1/3,
  labelSize = 3,
  labelAlpha = 1,
  highlightPointAlpha = 2/3,
  highlightPointSize = 4,
  highlightLabelAlpha = 1,
  highlightLabelSize = 5
)
```

Arguments

geneSetsList	'gsList' object with clusters and labels.
resolution	The cluster resolution to plot. Default NULL uses the active resolution: the optimal resolution found by clusterGeneSets(), or the one last set with updateResolution() or updateClusterLabels().
removeClusterId	Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.
interactive	Logical, indicating whether to plot an interactive UMAP plot. Default: FALSE. If TRUE, an interactive plot is made with 'HoverLocator()' from the Seurat package; hovering over a point shows its gene set and cluster.
highlightClusterNo	A vector with the cluster numbers of the clusters (at the resolution) to highlight in the plot.
pointSize	Size of points. Default is 2.
pointAlpha	Transparency of points. Default is 1/3.
labelSize	Size of the cluster labels. Default is 3.
labelAlpha	Transparency of the cluster labels. Default is 1.
highlightPointAlpha	Transparency of the points of highlighted clusters. Default is 2/3.
highlightPointSize	Size of the points of highlighted clusters. Default is 4.
highlightLabelAlpha	Transparency of the labels of highlighted clusters. Default is 1.
highlightLabelSize	Size of the labels of highlighted clusters. Default is 5.

Value

A ggplot object (a patchwork from 'Seurat::DimPlot()'), or a plotly object when `interactive = TRUE`.

Examples

```
data(exampleGsList)
plotUMAP(exampleGsList)
plotUMAP(exampleGsList, highlightClusterNo = 0, removeClusterId = TRUE)
```

---

plotViolin *Violin plot of clustering*

Description

Creates a violin plot of the values of the gene sets in each cluster, by default their -log10 p-values (or enrichment scores when no p-values were supplied). Clusters are ordered by their median value.

Usage

```
plotViolin(
  geneSetsList,
  resolution = NULL,
  plotFrom = c("nlog10pval", "enrichment_score"),
  removeClusterId = FALSE,
  highlightClusterNo = NULL
)
```

Arguments

`geneSetsList`	'gsList' object with clusters and labels.
`resolution`	The cluster resolution to plot. Default `NULL` uses the active resolution: the optimal resolution found by `clusterGeneSets()`, or the one last set with `updateResolution()` or `updateClusterLabels()`.
`plotFrom`	Name of the column of the 'enrichmentScore' slot to visualize, e.g. 'nlog10pval' or 'enrichment_score'. Only numeric columns that are annotated (i.e. not all NA) can be plotted - which columns are available depends on what was supplied to 'geneSetsDF' in 'initializeList()'. Only one column is plotted; if several are supplied the first plottable one is used (the default therefore falls back to 'enrichment_score' when no p-values were supplied).
`removeClusterId`	Logical, if TRUE the cluster number will be removed from each cluster label. Default: FALSE.
`highlightClusterNo`	A vector with the cluster numbers of the clusters (at the resolution) to highlight (in bold) in the plot.

Value

A ggplot object.

Examples

```
data(exampleGsList)
plotViolin(exampleGsList, highlightClusterNo = c(0, 1))
plotViolin(
    exampleGsList,
    plotFrom = 'enrichment_score',
    removeClusterId = TRUE
)
```

---

snnToIgraph *Convert an SNN graph to an igraph object*

Description

Turns the SNN graph from `getSNN()` into a weighted undirected `igraph` graph whose vertex names are the gene sets.

Usage

```
snnToIgraph(snn)
```

Arguments

snn	The SNN graph from `getSNN()` (a sparse symmetric matrix / Seurat `Graph`).

Value

A weighted undirected `igraph` object.

See Also

`getSNN()`, `snnToMatrix()`

Examples

```
data(exampleGsList)
g <- snnToIgraph(getSNN(exampleGsList))
g
```

snnToMatrix *Convert an SNN graph to a dense similarity matrix*

Description

Turns the SNN graph from `getSNN()`, whose values are similarities in `[0, 1]`, into a dense matrix with the gene-set names as dimnames, for methods that need a full similarity matrix.

Usage

```
snnToMatrix(snn, warnAbove = 5000)
```

Arguments

`snn`	The SNN graph from `getSNN()`.
`warnAbove`	Number of gene sets above which a warning about the memory cost of densifying is emitted. Set to `Inf` to silence. The dense matrix needs roughly `8 * N^2` bytes, so 10,000 gene sets is about 800 MB.

Value

A dense symmetric numeric matrix with gene sets as rows/columns.

See Also

`getSNN()`, `snnToIgraph()`

Examples

```
data(exampleGsList)
simMat <- snnToMatrix(getSNN(exampleGsList))
simMat[1:3, 1:3]
```

stopWords *Stop Words*

Description

Words ignored when labeling clusters, e.g. "of" and "and". Bigrams (two-word label candidates) containing one of these words are not used as labels.

Usage

```
stopWords
```

Format

A character vector of 48 lower-case words: common English words (e.g. "of", "the"), gene-set collection prefixes (e.g. "gobp", "reactome"), the direction suffixes "up" and "dn" and the word "vs" from MSigDB gene-set names, author names that start MSigDB gene-set names, and other short fragments of gene-set names (e.g. "a5p4", "2").

Source

Curated by the package authors.

Examples

```
head(stopWords)
# Add your own stop words
myStopWords <- c(stopWords, "regulation", "response")
```

---

updateClusterLabels *Rename a cluster*

Description

Replaces the label of one cluster with a name of your choice. The original label is kept and reported as `initial_cluster_label` by `extractTable(summarize = FALSE)`, also after several renames.

Usage

```
updateClusterLabels(
  geneSetsList,
  resolution = NULL,
  clusterNumber,
  newClusterName
)
```

Arguments

`geneSetsList`	'gsList' object that has been clustered and labeled.
`resolution`	Numeric. The clustering resolution of the cluster to rename. It must have been used as a resolution in `clusterGeneSets()`. When given, this resolution also becomes the active resolution: plotting and table functions called with `resolution = NULL` afterwards use it. Default `NULL` renames a cluster of the active resolution.
`clusterNumber`	The number of the cluster to rename, as shown in front of its label (e.g. `0` for "0: Immune Response").
`newClusterName`	The new name of the cluster.

Value

'gsList' object with the cluster label updated.

Examples

```
data(exampleGsList)
gsListObject <- updateClusterLabels(
    geneSetsList = exampleGsList,
    clusterNumber = 0,
    newClusterName = 'cancer response'
)
head(extractTable(gsListObject, summarize = FALSE))
```

---

updateResolution *Update default resolution of gsList*

Description

Sets the resolution used by all plotting and table functions when their `resolution` argument is `NULL`. It replaces the optimal resolution found by `clusterGeneSets()` as the default.

Usage

```
updateResolution(geneSetsList, resolution)
```

Arguments

`geneSetsList`	'gsList' object that has been clustered and labeled.
`resolution`	Numeric. The clustering resolution to use as the new default. It must have been used as a resolution in `clusterGeneSets()`.

Value

'gsList' object updated with the new default resolution.

Examples

```
data(exampleGsList)
gsListObject <- updateResolution(exampleGsList, 0.6)
extractResolution(gsListObject)
```

Index

∗ **datasets**
    exampleEnrichment, 4
    exampleGeneSets, 5
    exampleGsList, 5
    stopWords, 28

clusterGeneSets, 3, *7*
clusterGeneSets(), *11*

exampleEnrichment, 4
exampleGeneSets, 5
exampleGsList, 5
extractResolution, 6
extractTable, 6
extractTable(), *11*

fgsea::fgsea(), *14*
fgsea::gmtPathways(), *12*

geneSetSimplifyR, 7
geneSetSimplifyR(), *11*
getGeneSets, 9
getSNN, 10
getSNN(), *3*, *27*, *28*
gsList, *5*
gsList *(*gsList-class*)*, 11
gsList-class, 11

initializeList, *7*, *8*, 12
initializeList(), *11*

labelClusters, *7*, 13
labelClusters(), *11*

msigdbr::msigdbr(), *9*

parseFgsea, 14
parsePairedGSEA, 15
plotBarchart, 17
plotClusterLabels, 18
plotClustree, 19
plotDendrogram, 20
plotEnrichments, 21
plotHeatmap, 22
plotProportions, 23
plotUMAP, 24
plotViolin, 26

show,gsList-method *(*gsList-class*)*, 11
snnToIgraph, 27
snnToIgraph(), *10*, *28*
snnToMatrix, 28
snnToMatrix(), *10*, *27*
stopWords, *8*, 28

updateClusterLabels, 29
updateResolution, 30