Package 'multiGSEA'

November 12, 2026

Type Package

Title Combining GSEA-based pathway enrichment with multi omics data integration

Version 1.22.0

Date 2020-03-05

Description Extracted features from pathways derived from 8 different databases (KEGG, Reactome, Biocarta, etc.) can be used on transcriptomic, proteomic, and/or metabolomic level to calculate a combined GSEA-based enrichment score.

License GPL-3

Depends R (>= 4.0.0)

Imports magrittr, graphite, AnnotationDbi, metaboliteIDmapping, dplyr, fgsea, metap, rappdirs, rlang, methods

VignetteBuilder knitr

biocViews GeneSetEnrichment, Pathways, Reactome, BioCarta

URL https://github.com/yigbt/multiGSEA

BugReports https://github.com/yigbt/multiGSEA/issues

Encoding UTF-8

NeedsCompilation no

RoxygenNote 7.2.0

Suggests org.Hs.eg.db, org.Mm.eg.db, org.Rn.eg.db, org.Ss.eg.db, org.Bt.eg.db, org.Ce.eg.db, org.Dm.eg.db, org.Dr.eg.db, org.Gg.eg.db, org.Xl.eg.db, org.Cf.eg.db, knitr, rmarkdown, BiocStyle, testthat (>= 2.1.0)

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

git_branch RELEASE_3_23

git_last_commit 722e5d5

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Sebastian Canzler [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7935-9582>),
Jörg Hackermüller [aut] (ORCID:
<https://orcid.org/0000-0003-4920-7072>)

Maintainer Sebastian Canzler <sebastian.canzler@ufz.de>

Contents

archiveDir 2
archivePath 3
combinePvalues 3
extractPvalues 4
getFeatures 5
getGeneMapping 6
getIDMappingDatabase 7
getMappedFeatures 7
getMetaboliteIDformats 8
getMetaboliteMapping 9
getMultiOmicsFeatures 9
getOrganisms 11
initOmicsDataStructure 11
loadLocal 12
mapIDType 12
metabolome 13
multiGSEA 14
proteome 15
rankFeatures 15
rename_duplicates 16
transcriptome 17

Index **18**

`archiveDir` *Retrieve the path to the cache directory.*

Description

Retrieve the path to the cache directory for the multiGSEA package. Create the cache directory if need be.

Usage

```
archiveDir()
```

Value

String containing the path to the cache directory.

archivePath *Retrieve path to a cached file.*

Description

The function retrieves the path to a file that is cached in the archive directory.

Usage

```
archivePath(filename)
```

Arguments

filename	Name of the file.

Value

String containing the path to the file.

combinePvalues *Calculate a combined p-value for multiple omics layer.*

Description

This function applies the Stouffer method, the Edgington method or the Fisher\'s combined probability test to combine p-values of independent tests that are based on the same null hypothesis. The Stouffer method can also be applied in a weighted fashion.

Usage

```
combinePvalues(df, method = "stouffer", col_pattern = "pval", weights = NULL)
```

Arguments

df	Data frame where rows represent a certain pathway or gene set and columns represent p-values derived from independent tests, e.g., different omics layer.
method	String that specifies the method to combine multiple p-values. Default: "stouffer" Options: "stouffer", "fisher", "edgington"
col_pattern	String of the pattern that specifies the columns to be combined. Default: "pval", Options: "pval", "padj" (legacy)
weights	List of weights that will be used in a weighted Stouffer method.

Value

Vector of length `nrow(df)` with combined p-values.

Examples

```
df <- cbind(runif(5), runif(5), runif(5))
colnames(df) <- c("trans.pval", "prot.pval", "meta.pval")

# run the unweighted summation of z values
combinePvalues(df)

# run the weighted variant
combinePvalues(df, weights = c(10, 5, 1))

# run the Fisher's combined probability test
combinePvalues(df, method = "fisher")

# run the Edgington's method
combinePvalues(df, method = "edgington")
```

`extractPvalues` *Create a reshaped data frame from multiGSEA output.*

Description

This function reshapes the output from multiGSEA to get a single data frame with columns for p-values and adjusted p-values for each omics layer. Each row of the data frame represents one pathway.

Usage

```
extractPvalues(enrichmentScores, pathwayNames)
```

Arguments

`enrichmentScores`	Nested List of enrichment scores, calculated by multiGSEA function.
`pathwayNames`	List containing Pathway names.

Value

Data frame where rows are pathways and columns are (adjusted) p-values for each omics layer.

Examples

```
# Download pathway definition and extract features
pathways <- getMultiOmicsFeatures(dbs = c("kegg"), layer = c("transcriptome", "proteome"))

# load omics data and calculate ranks
data(transcriptome)
data(proteome)
ranks <- initOmicsDataStructure(c("transcriptome", "proteome"))
ranks$transcriptome <- rankFeatures(transcriptome$logFC, transcriptome$pValue)
```

```
names(ranks$transcriptome) <- transcriptome$Symbol
ranks$proteome <- rankFeatures(proteome$logFC, proteome$pValue)
names(ranks$proteome) <- proteome$Symbol

# run the enrichment
es <- multiGSEA(pathways, ranks)

extractPvalues(
  enrichmentScores = es,
  pathwayNames = names(pathways[[1]])
)
```

`getFeatures` *Wrapper to extract features (nodes) from given pathways.*

Description

Function to extract the features (nodes) from a given list of pathways. These pathways have to be compiled with the `pathways` function. Features can only be extracted for \'proteins\' or \'metabolites\'. Features will by default be mapped to gene symbols.

Usage

```
getFeatures(
  pathway,
  which = "proteins",
  org = "hsapiens",
  returntype = "SYMBOL"
)
```

Arguments

`pathway`	A pathway created with `pathways` command.
`which`	Mode to extract the features, either \'proteins\' or \'metabolites\'.
`org`	String specifying the organism, which is necessary for featureID mapping. Default: human
`returntype`	String that specifies the returning ID type. Default: SYMBOL Options (genes/proteins): SYMBOL, ENTREZID, UNIPROT, ENSEMBL, REFSEQ Options (metabolites): HMDB, CAS, DTXCID, DTXSID, SID, CID, ChEBI, KEGG, Drugbank

Value

Feature list with gene symbols (genes/proteins) or CHEBI IDs (metabolites)

Examples

```
pathways <- graphite::pathways("hsapiens", "kegg")[[1]]
getFeatures(pathways)

pathways <- graphite::pathways("mmusculus", "kegg")[[1]]
getFeatures(pathways, which = "metabolites", org = "mmusculus", returntype = "HMDB")

pathways <- graphite::pathways("mmusculus", "kegg")[[1]]
getFeatures(pathways, which = "proteins", org = "mmusculus", returntype = "SYMBOL")
```

`getGeneMapping` *Mapping between pathway encoded genes/proteins and different ID formats.*

Description

Function to retrieve the gene/protein identifier mapping. Ongoing from genes/proteins retrieved from pathway definitions, which often include two or more ID formats or a format that is not present in your omics measurement, this function maps those IDs to a given format. Depending on the organism, additional packages have to be installed.

Usage

```
getGeneMapping(features, keytype, org = "hsapiens", returntype = "SYMBOL")
```

Arguments

`features`	List of identifiers to be mapped.
`keytype`	String specifying the ID type, e.g., "ENTREZID" or "UNIPROT".
`org`	String that defines the organism. Default: hsapiens Options: see `getOrganisms`
`returntype`	String that specifies the returning ID type. Default: SYMBOL, Options: SYMBOL, ENTREZID, UNIPROT, ENSEMBL, REFSEQ

Value

List containing mapped gene/protein IDs.

Examples

```
features <- graphite::nodes(graphite::pathways("hsapiens", "kegg")[[1]])
features <- gsub("ENTREZID:", "", features)
keytype <- "ENTREZID"
getGeneMapping(features, keytype)

getGeneMapping(features, keytype, returntype = "UNIPROT")
```

```
features <- graphite::nodes(graphite::pathways("rnorvegicus", "reactome")[[1]])
features <- gsub("UNIPROT:", "", features)
getGeneMapping(features, keytype = "UNIPROT", org = "rnorvegicus")

getGeneMapping(features,
  keytype = "UNIPROT",
  org = "rnorvegicus",
  returntype = "ENSEMBL"
)
```

`getIDMappingDatabase` *Get the correct ID mapping database*

Description

Check by means of the given organism name if the required 'AnnotationDbi' package is installed. Select the ID mapping table based on the organism name and return it.

Usage

```
getIDMappingDatabase(organism)
```

Arguments

`organism`	String that defines the organism.

Value

AnnotationDbi database for ID mapping.

`getMappedFeatures` *Wrapper to get feature mappings.*

Description

Feature mappings will be used from hard disk in case they have been mapped before and 'useLocal' is not set to be FALSE. In other cases, a feature extraction will be done and the results are stored for a following occasion.

Usage

```
getMappedFeatures(
  pathways,
  returnID = "SYMBOL",
  organism = "hsapiens",
  which = "proteins",
  useLocal = TRUE
)
```

Arguments

pathways	List of pathway definitions.
returnID	String specifying the returned ID format.
organism	String defining the organism of analysis.
which	Mode to extract the features, either \'proteins\' or \'metabolites\'.
useLocal	Boolean specifying whether or not to use the local preprocessed mapping.

Value

List of mapped features for an omics layer.

---

getMetaboliteIDformats

Helper function to get all different metabolite ID formats

Description

This helper function extracts all used ID formats in all pathways and returns a nested list for each pathway database.

Usage

```
getMetaboliteIDformats(pathways)
```

Arguments

pathways	List of pathway databases and their pathway definition.

Value

List of metabolite ID formats.

getMetaboliteMapping *Mapping between pathway encoded metabolites and different metabolite ID formats.*

Description

Function to retrieve the metabolite identifier mapping. Ongoing from metabolites retrieved from pathway definitions, which often include two or more ID formats, this function maps those IDs to a given format. The complete mapping table based on Comptox Dashboard, PubChem, HMDB, and ChEBI is provided in the AnnotationHub package metaboliteIDmapping.

Usage

```
getMetaboliteMapping(features, keytype, returntype = "HMDB")
```

Arguments

features	List of identifiers to be mapped.
keytype	String specifying the ID type, e.g., "ChEBI" or "KEGGCOMP".
returntype	String that specifies the returning ID type. Default: HMDB Options: HMDB, CAS, DTXCID, DTXSID, SID, CID, ChEBI, KEGG, Drugbank

Value

List containing mapped gene/protein IDs.

Examples

```
features <- graphite::nodes(graphite::pathways("hsapiens", "kegg")[[1]], which = "metabolites")
features <- gsub("KEGGCOMP:", "", features)
keytype <- "KEGG"

getMetaboliteMapping(features, keytype)

getMetaboliteMapping(features, keytype = "KEGG", returntype = "CID")
```

getMultiOmicsFeatures *Collect feature mapping for user given databases and omics layer.*

Description

The functions makes use of the graphite R package to collect pathways from user specified databases. Depending on the omics layer specified, the function extracts either annotated genes/proteins (for transcriptome, proteome layer) or metabolites (for metabolite layer). The data structure that is returned is mandatory to calculate the multi-omics pathway enrichment.

Usage

```
getMultiOmicsFeatures(
  dbs = c("all"),
  layer = c("all"),
  returnTranscriptome = "SYMBOL",
  returnProteome = "SYMBOL",
  returnMetabolome = "HMDB",
  organism = "hsapiens",
  useLocal = TRUE
)
```

Arguments

dbs	List of databases that should be queried for pathways. Default: all available databases
layer	List of omics layer that should be addressed. Default: all three layer (transcriptome, proteome, metabolome)
returnTranscriptome	String specifying the returned gene ID format. Default: SYMBOL Options: SYMBOL, ENTREZID, UNIPROT, ENSEMBL, REFSEQ
returnProteome	String specifying the returned protein ID format. Default: SYMBOL Options: SYMBOL, ENTREZID, UNIPROT, ENSEMBL, REFSEQ
returnMetabolome	String specifying the returned metabolite ID format. Default: HMDB Options: HMDB, CAS, DTXCID, DTXSID, SID, CID, ChEBI, KEGG, Drugbank
organism	String specifying the organism of interest. This has direct influence on the available pathway databases. Default: "hsapiens" Options: see getOrganisms
useLocal	Boolean to use local pathway/feature descriptions. In case useLocal is set to FALSE, pathway definitions and feature extraction will be recalculated. This could take several minutes depending on the database used. Pathbank, for example, contains nearly 50000 pathway definition that have to be re-mapped. useLocal has no effect when pathway definitions are retrieved for the first time. Default: TRUE

Value

Nested list with extracted and mapped pathway features.

Examples

```
getMultiOmicsFeatures(
  dbs = c("kegg"),
  layer = c("transcriptome", "proteome"),
  organism = "hsapiens"
)

getMultiOmicsFeatures(
  dbs = c("kegg", "reactome"),
```

```
  layer = c("transcriptome", "metabolome"),
  organism = "mmusculus"
)

getMultiOmicsFeatures(
  dbs = c("reactome"),
  layer = c("proteome"),
  organism = "rnorvegicus",
  returnProteome = "ENTREZID"
)
```

`getOrganisms` *Get list of supported organisms*

Description

Get a list of organisms that are covered in our workflow through a supporting 'AnnotationDBi' package. Without such a package we would not be able to map transcript and protein identifier between different formats. All the organisms that are listed here have at lest kegg and or reactome pathway annotation that can be queried by 'graphite'.

Usage

```
getOrganisms()
```

Value

List of supported organisms

Examples

```
getOrganisms()
```

`initOmicsDataStructure` *Create an empty data structure for measured omics features*

Description

This function creates a data structure of nested but empty lists. One list for each omics layer. By default all three supported omics layer are used to create a data structures with three empty sublists: transcriptome, proteome, and metabolome.

Usage

```
initOmicsDataStructure(layer = c("transcriptome", "proteome", "metabolome"))
```

Arguments

layer	List specifying the omics layer which should be created

Value

List with length(layer) empty sublists

Examples

```
initOmicsDataStructure()
initOmicsDataStructure(c("transcriptome", "proteome"))
initOmicsDataStructure(c("Transcriptome", "Metabolome"))
```

---

loadLocal *Read a local RDS file.*

Description

Use the readRDS function to load the given file which should be in RDS format.

Usage

```
loadLocal(filename)
```

Arguments

filename	Path to the file to be read.

Value

Content of file.

---

mapIDType *Helper function to map only a subset of metabolite IDs*

Description

This helper function becomes necessary since there are sometimes multiple ID formats used in a single pathway definition.

Usage

```
mapIDType(features, keytype = "CHEBI", maptype = "ChEBI", returntype = "HMDB")
```

Arguments

features	List of metabolite feature IDs of the pathway.
keytype	String specifying the ID format in pathway definition.
maptype	String specifying the corresponding ID format in multiGSEA.
returntype	String identifying the ID type that should be mapped.

Value

List of mapped metabolite IDs.

---

metabolome *Metabolomic data set that is used in the toy example provided by the 'multiGSEA' package.*

Description

Processed metabolomics data set that will be used throughout the vignette provided by the 'multiGSEA' package. The raw data was originally published by [Quiros _et al._](http://doi.org/10.1083/jcb.201702058) and can be accessed within the online supplementary material.

Usage

```
data(metabolome)
```

Format

A tibble with 4 variables and 4881 measured proteome features:

HMDB HMDB identifier of measured metabolites.

logFC Log2-transformed fold change between treatment and control.

pValue P-value associated with the fold change.

adj.pValue Adjusted p-value associated with the fold change.

Examples

```
data(metabolome)
```

multiGSEA *Calculate pathway enrichment for multiple omics layer.*

Description

This function calculates GSEA-based enrichments scores for multiple omics layer at once. Input pathways or gene sets have to be prepared in advance by means of the function initOmicsDataStructure. The function uses pre- ranked lists for each omics layer to calculate the enrichment score. The ranking can be calculated by means of the function rankFeatures.

Usage

```
multiGSEA(pathways, ranks, eps = 0)
```

Arguments

pathways	Nested list containing all pathway features for the respective omics layer.
ranks	Nested list containing the measured and pre-ranked features for each omics layer.
eps	This parameter sets the boundary for calculating the p value.

Value

Nested list containing the enrichment scores for each given pathway and omics layer.

Examples

```
# Download pathway definition and extract features
pathways <- getMultiOmicsFeatures(dbs = c("kegg"), layer = c("transcriptome", "proteome"))

# load omics data and calculate ranks
data(transcriptome)
data(proteome)
ranks <- initOmicsDataStructure(c("transcriptome", "proteome"))
ranks$transcriptome <- rankFeatures(transcriptome$logFC, transcriptome$pValue)
names(ranks$transcriptome) <- transcriptome$Symbol
ranks$proteome <- rankFeatures(proteome$logFC, proteome$pValue)
names(ranks$proteome) <- proteome$Symbol

## run the enrichment
multiGSEA(pathways, ranks)
```

proteome — *Proteomic data set that is used in the toy example provided by the 'multiGSEA' package.*

Description

Processed proteomics data set that will be used throughout the vignette provided by the 'multiGSEA' package. The raw data was originally published by [Quiros _et al._](http://doi.org/10.1083/jcb.201702058) and deposited at [ProteomeXchange](http://proteomecentral.proteomexchange.org/cgi/GetDataset?ID=PXD006293).

Usage

```
data(proteome)
```

Format

A tibble with 4 variables and 8275 measured proteome features:

Symbol HGNC symbol of measured proteins.

logFC Log2-transformed fold change between treatment and control.

pValue P-value associated with the fold change.

adj.pValue Adjusted p-value associated with the fold change.

Examples

```
data(proteome)
```

rankFeatures — *Pre-rank features prior to calculating enrichment scores.*

Description

Rank features based on the direction of their fold change and their magnitude implicated through their assigned p-value.

Usage

```
rankFeatures(logFC, pvalues, base = 10)
```

Arguments

logFC	Vector containing the log-transformed fold changes of features.
pvalues	Vector containing the p-values associated with those logFCs.
base	Integer specifying the base of the logarithm. Default: 10

Value

Vector of pre-ranked features, still unsorted

Examples

```
logFC <- rnorm(10)
pvalues <- runif(10)
rankFeatures(logFC, pvalues)
```

---

rename_duplicates *Make a list of strings unique*

Description

It might happen that there are duplicated strings in a list. With this function we will rename those duplicated entries in a way that we simply add the number of occurrences to the string. I.e., when the string foo occurs three times in a list, it will be renamed to foo_1, foo_2, and foo_3, respectively.

Usage

```
rename_duplicates(names)
```

Arguments

names	List of strings where duplicates should be renamed

Value

List where duplicates are renamed.

Examples

```
l <- c("foo", "bar", "foo", "bars")
rename_duplicates(l)
```

`transcriptome` *Transcriptomic data set that is used in the toy example provided by the 'multiGSEA' package.*

Description

Processed transcriptomics data set that will be used throughout the vignette provided by the 'multiGSEA' package. The raw data was originally published by [Quiros _et al._](http://doi.org/10.1083/jcb.201702058) and deposited at [NCBI Geo](https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE84631).

Usage

```
data(transcriptome)
```

Format

A tibble with 4 variables and 15174 measured transcriptome features:

Symbol HGNC symbol of measured transcripts.

logFC Log2-transformed fold change between treatment and control.

pValue P-value associated with the fold change.

adj.pValue Adjusted p-value associated with the fold change.

Examples

```
data(transcriptome)
```

Index

∗ **datasets**
metabolome, 13
proteome, 15
transcriptome, 17

archiveDir, 2
archivePath, 3

combinePvalues, 3

extractPvalues, 4

getFeatures, 5
getGeneMapping, 6
getIDMappingDatabase, 7
getMappedFeatures, 7
getMetaboliteIDformats, 8
getMetaboliteMapping, 9
getMultiOmicsFeatures, 9
getOrganisms, *6*, *10*, 11

initOmicsDataStructure, 11, *14*

loadLocal, 12

mapIDType, 12
metabolome, 13
multiGSEA, 14

pathways, *5*
proteome, 15

rankFeatures, *14*, 15
rename_duplicates, 16

transcriptome, 17