Package 'leapR'

June 5, 2026

Title Layered enrichment analysis of pathways R

Version 1.1.0

Description leapR is a package that identifies pathways that are enriched across diverse 'omics experiments. It leverages any tabular expression data (proteomics, transcriptomics) using the `SummarizedExperiment` object. It works with any pathway in the .gct file format.

Depends R (>= 4.5.0)

Encoding UTF-8

LazyData false

RoxygenNote 7.3.3

biocViews GeneSetEnrichment, Proteomics, Pathways, GeneExpression, Transcriptomics

Imports stats, gplots, readr, tibble, gplots, methods, ggplot2, dplyr, stringr, tidyr, SummarizedExperiment, BiocStyle

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

License MIT + file LICENSE

Config/testthat/edition 3

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

git_branch devel

git_last_commit ea5f969

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Author Sara Gosline [aut, cre] (ORCID: <https://orcid.org/0000-0002-6534-4774>), Jason McDermott [aut], Jeremy Jacobson [aut], Vincent Danna [ctb], National Institutes of Health [fnd]

Maintainer Sara Gosline <sara.gosline@pnnl.gov>

Contents

`leapR-package` *leapR: Layered enrichment analysis of pathways R*

Description

leapR is a package that identifies pathways that are enriched across diverse 'omics experiments. It leverages any tabular expression data (proteomics, transcriptomics) using the 'SummarizedExperiment' object. It works with any pathway in the .gct file format.

Author(s)

Maintainer: Sara Gosline <sara.gosline@pnnl.gov> (ORCID)

Authors:

- Jason McDermott <jason.mcdermott@pnnl.gov>
- Jeremy Jacobson <jeremy.jacobson@pnnl.gov>

Other contributors:

- Vincent Danna <vincent.danna@pnnl.gov> [contributor]
- National Institutes of Health [funder]

calcTTest *calcTTest*

Description

calculates a t-test for two distributions of data on a per-gene basis append results to ExpressionSet with two extra columns: ‘pvalue‘ and ‘difference‘ for each feature

Usage

```
calcTTest(eset, assay_name, group1, group2)
```

Arguments

eset	SummarizedExperiment
assay_name	name of assay
group1	List of samples comprising group 1
group2	List of samples comprising group 2

Value

An Expression set with two columns added to the featureData slot: pvalue, and estimate

Examples

```
library(leapR)
url <- "https://api.figshare.com/v2/file/download/56536214"
tdata <- download.file(url,method='libcurl',destfile='transData.rda')
load('transData.rda')
p <- file.remove("transData.rda")

# read in the pathways
data("ncipid")

# read in the patient groups
data("shortlist")
data("longlist")
calcTTest(tset, 'transcriptomics', shortlist, longlist)
```

cluster_enrichment *cluster_enrichment*

Description

Cluster enrichment Run enrichment (Fisher’s exact) on clusters (lists of identifier groups)

Usage

```
cluster_enrichment(eset, assay_name, geneset, clusters, sigfilter = 0.05)
```

Arguments

eset	is an SummarizedExperiment containing data that is clustered
assay_name	is the name of the assay
geneset	is a GeneSet object for pathway annotation
clusters	is a list of clusters (gene lists) to calculate enrichment on, generally the result of the ‘cutree‘ function
sigfilter	minimum significance threshold default is .05

Details

This function will calculate enrichment (Fisher's exact test for membership overlap) on a series of lists of genes, such as from a set of clusters. The results are returned as a list of results matrices in the order of the input clusters.

Value

data frame with enrichment results

Examples

```
library(leapR)

# read in the example transcriptomic data
url <- "https://api.figshare.com/v2/file/download/56536214"
tdata <- download.file(url,method='libcurl',destfile='transData.rda')
load('transData.rda')
p <- file.remove("transData.rda")

# read in the pathways
data("ncipid")

# for the example we will limit the number of transcripts considered
#- arbitrarily in this case
transdata <- SummarizedExperiment::assay(tset,'transcriptomics')
transdata[which(is.na(transdata),arr.ind=TRUE)]<-0.0
# perform heirarchical clustering on the  data
transdata.hc <- hclust(dist(transdata), method="ward.D2")

transdata.hc.clusters <- cutree(transdata.hc, k=5)
clust.list <- lapply(seq_len(5), function(x) {
   return(names(which(transdata.hc.clusters==x)))})
#calculates enrichment for each of the clusters individually a
#and returns a list of enrichment results
transdata.hc.enrichment <- leapR::cluster_enrichment(eset=tset,
        assay_name='transcriptomics',
        geneset=ncipid,
        clusters=clust.list)
```

combine_omics *combine_omics Combine two or more omics matrices into one multi-omics matrix with 'tagged' ids.*

Description

combine_omics Combine two or more omics matrices into one multi-omics matrix with 'tagged' ids.

Usage

```
combine_omics(omics_list, id_list = rep(NA, length(omics_list)))
```

Arguments

omics_list	Is a list of SummarizedExperiment each with one assay
id_list	List of identifiers to use, in the same order as the omics_list elements. If an element is 'NA', then rownames are used.

Details

This combines matrices of different omics types together and adds prefix tags to the ids.

Value

SummarizedExperiment with an additional assay called 'combined'

Examples

```
library(leapR)
url <- 'https://api.figshare.com/v2/file/download/56536217'

pdata <- download.file(url,method='libcurl',destfile='protData.rda')
load('protData.rda')
p <- file.remove("protData.rda")

url <- "https://api.figshare.com/v2/file/download/56536214"
tdata <- download.file(url,method='libcurl',destfile='transData.rda')
load('transData.rda')
p <- file.remove("transData.rda")

url <- 'https://api.figshare.com/v2/file/download/56536211'
phdata<-download.file(url,method='libcurl',destfile = 'phosData.rda')
#phosphodata<-read.csv("phdata",check.names=FALSE,row.names=1)
load('phosData.rda')
p <- file.remove('phosData.rda')# read in the example protein data


# merge the three datasets by rows and add prefix tags for
# different omics types
multi_omics <- combine_omics(list(pset, tset, phset),
            list(NA,NA,'hgnc_id'))
```

`correlation_comparison_enrichment` *correlation_comparison_enrichment*

Description

internal function to calculate enrichment in differences in correlation # between two groups # access through the leapr wrapper

Usage

```
correlation_comparison_enrichment(
  geneset,
  eset,
  assay_name,
  set1,
  set2,
  mapping_column = NA
)
```

Arguments

`geneset`	pathway to use for enrichment
`eset`	SummarizedExperiment with abundance matrix
`assay_name`	name of assay
`set1`	first set to use
`set2`	second set to use
`mapping_column`	Column to use for id mapping within rowData

Value

data frame with enrichment results

`correlation_enrichment` *correlation_enrichment*

Description

calculate enrichment in correlation between pathway members # access through leapr wrapper

Usage

```
correlation_enrichment(geneset, eset, assay_name, mapping_column = NA)
```

Arguments

geneset	Geneset list
eset	a SummarizedExperiment object
assay_name	name of assay
mapping_column	Column to use to map identifiers, if not rownames

Value

list of enrichment statistic table and correlation matrix

---

enrichment_in_abundance

enrichment_in_abundance

Description

Enrichment in abundance calculates enrichment in pathways by the difference in abundance of the pathway members.

Usage

```
enrichment_in_abundance(
  geneset,
  eset,
  assay_name,
  mapping_column = NULL,
  abundance_column = NULL,
  fdr = 0,
  matchset = NULL,
  sample_comparison = NULL,
  min_p_threshold = NULL,
  sample_n = NULL,
  silence_try_errors = TRUE
)
```

Arguments

geneset	Gene set to calculate enrichment
eset	Molecular abundance data in 'SummarizedExperiment' format
assay_name	Name of assay to compare
mapping_column	Column to use to map identifiers
abundance_column	Columns to use to quantify abundance
fdr	number of times to sample for FDR value
matchset	Name of a set to use for enrichment
sample_comparison	list of samples to use as comparison. if missing background (eset) is used

`min_p_threshold`
: Only include p-values lower than this

`sample_n`
: size of sample to use

`silence_try_errors`
: set to true to silence try errors

Value

data frame of enrichment result

`enrichment_in_groups` *enrichment_in_groups*

Description

Calculate the enrichment in pathways using Fisher's exact or Kolmogorov-Smirnov test, using either the abundance column to identify feature or the targets list. access through leapr wrapper

Usage

```
enrichment_in_groups(
  geneset,
  targets = c(),
  background = NULL,
  assay_name = NULL,
  method = "fishers",
  minsize = 5,
  mapping_column = NULL,
  log_transformed = FALSE,
  abundance_column = NULL,
  randomize = FALSE,
  silence_try_errors = TRUE
)
```

Arguments

`geneset`
: geneset to use for enrichment

`targets`
: targets to use for enrichment

`background`
: 'SummarizedExperiment' describing background to use

`assay_name`
: is the name of the assay to use from the background

`method`
: method to use for statistical test, options are 'fishers', 'ks', 'ztest', or 'chisq'. Remember that KS test assumes normality, so it would be good to log your data before calling. NOTE: if you do not call 'suppressWarnings' then the KS test will warn you about ties.

`minsize`
: minimum size of set

`mapping_column`
: column name of mapping identifiers

`log_transformed`
: Set to TRUE if data are log transformed

abundance_column
: columns mapping abundance, either in the ‘assay‘ matrix or ‘rowData‘

randomize
: true/false whether to randomize

silence_try_errors
: true/false to silence errors

Value

data frame with enrichment results

`enrichment_in_relationships` *enrichment_in_relationships*

Description

enrichment_in_relationships function description is a general way to determine if a pathway is enriched in relationships (interactions, correlation) between its members # access through leapr wrapper

Usage

```
enrichment_in_relationships(
  geneset,
  relationships,
  idmap = NA,
  silence_try_errors = TRUE
)
```

Arguments

geneset
: List of pathways in gmt format

relationships
: table of relationship information, e.g. correlation

idmap
: list of identifiers to use for mapping, the names of the items should agree with names of features in matrix

silence_try_errors
: boolean to silence errors

Value

table of enrichment statistics

get_pathway_information *get_pathway_information*

Description

get_pathway_information extracts information about a pathway from a GeneSet object

Usage

```
get_pathway_information(geneset, path, remove.tags = FALSE)
```

Arguments

geneset	is a GeneSet object for pathway annotation
path	is the name of the gene set pathway to be return
remove.tags	boolean indicating whether to remove tags

Value

list of pathway information

Examples

```
library(leapR)

# load example gene set
data("ncipid")

tnfpathway = get_pathway_information(ncipid, "tnfpathway")
```

kinasesubstrates *Kinase substrate lists*

Description

Kinase substrate lists

Usage

```
kinasesubstrates
```

Format

A list with 4 items

names The names of the kinases

desc Short description of the kinase

sizes Length of the substrate list

matrix Substrate list for the kinase

Source

PhosphositePlus

`krbpaths` *KEGG, Reactome, BioCarta Pathways*

Description

KEGG, Reactome, BioCarta Pathways

Usage

```
krbpaths
```

Format

A list with 4 items

names The names of the pathways

desc Short description of the pathways

sizes Number of genes in the signaling pathways

matrix Matrix containing the genes in the pathways

Source

https://www.gsea-msigdb.org/gsea/msigdb_license_terms.jsp

`leapR` *leapR*

Description

leapR is a wrapper function that consolidates multiple enrichment methods.

Usage

```
leapR(geneset, enrichment_method, eset, assay_name, ...)
```

Arguments

`geneset`	is a list of four vectors, gene names, gene descriptions, gene sizes and a matrix of genes. It represents .gmt format pathway files.
`enrichment_method`	is a character string specifying the method of enrichment to be performed, one of: "enrichment_comparison", "enrichment_in_order", "enrichment_in_sets", "enrichment_in_pathway", "correlation_enrichment".
`eset`	is a 'SummarizedExperiment' object containing expression data, with features as rows and *n* sample/conditions as columns.
`assay_name`	is the assay to be analyzed within the 'eset'. Recommended to describe the data type (e.g. transcriptomics, proteomics) so that it can be integrated in 'combine_omics'
`...`	further arguments

Details

Further arguments and enrichment method optional argument information:

id_column	Is a character string, present in the `rowData` slot, that is used to specify a column for identifiers to n
primary_columns	Is a character vector composed of column names from `eset` (either in the 'assay' or in the 'rowData
secondary_columns	Is a character vector of column names for comparison, pulled from the 'assay' of the SummarizedE
threshold	Is a numeric value, an optional argument used with 'enrichment_in sets' method which filters out al
greaterthan	Is a logical value that defaults to TRUE, it's used with 'enrichment_in_sets' method. When set to T
minsize	Is a numeric value, an optional argument used with 'enrichment_in_sets' and 'enrichment_in_order
fdr	A numerical value which specifies how many times to randomly sample genes to calculate an empi
min_p_threshold	Is a numeric value, a lower p-value threshold and is an optional argument used with 'enrichment_co
sample_n	Is a way to subsample the number of components considered for each calculation randomly. This is

Enrichment Methods:

enrichment_comparison
Compares the distribution of abundances between two sets of conditions for each pathway using a t test. For each pathway in `geneset` uses a t test to compare the distribution of abundance values/numbers in `eset` `primary_columns` with those in `eset` `secondary_columns`. Lower p-values for pathways indicate that the expression of the pathway is significantly different between the set of conditions in primary_columns and the set of conditions in secondary_columns. Optionally, users can specify `fdr` which will calculate an empirical p-value by randomizing abundances `fdr` number of times. If the `min_p_threshold` is specified the method will only return pathways with an adjusted p-value lower than the specified threshold. If `sample_n` is specified the method will subsample the pathway members to the specified number of components.

enrichment_in_order
Calculates enrichment of pathways based on a ranked list using the Kolmogorov-Smirnov test. For each pathway in `geneset` uses a Kolmogorov-Smirnov test for rank order to test if the distribution of ranked abundance values in the `eset` `primary_columns` is significant relative to a random distribution. Note that currently `primary_columns` only accepts a single column for this method.

enrichment_in_sets
Calculates enrichment in pathway membership in a list (e.g. highly differential proteins) relative to background using Fisher's exact test. For each pathway in `geneset` uses a Fisher's exact test over- or under- representation of a list of components specified. If `targets` are specified this must be a vector of identifiers to serve as the target list for comparison. If `eset` and `primary_columns` are specified then `threshold` specifies a threshold value for determining the target list of components to test. Specifying `greaterthan` to be False will result in components with values lower than the specified `threshold`. If `eset` is a data frame or matrix, the background used for calculation will be taken as the rownames of `eset`

enrichment_in_pathway
Compares the distribution of abundances in a pathway with the background distribution of abundances using a t test. For each pathway in `geneset` calculates the significance of the difference between the abundances from pathway members versus abundance of non-pathway members in the set of conditions specified by `primary_columns`. Optionally, users can specify `fdr` which will calculate an empirical p-value by randomizing abundances `fdr` number of times. If the `min_p_threshold` is specified the method will only return pathways with an adjusted p-value lower than the specified threshold. If `sample_n` is specified the method will subsample the pathway members to the specified number of components.

correlation_enrichment
Calculates the enrichment of a pathway based on correlation between pathway members across conditions versus correlation between members not in the pathway. For each pathway in `geneset` calculates the pairwise correlation between all pathway members and non-pathway members across the specified `primary_columns` conditions in `eset`. Note that for large matrices this can take a long time. A p-value is calculated based on comparing the correlation within the members of a pathway with the correlation values between members of the pathway and non-members of the pathway.

Value

data frame with results

Examples

```
       library(leapR)

# read in the example abundance data
# read in the example transcriptomic data
tdata <- download.file("https://api.figshare.com/v2/file/download/56536214",
     method='libcurl',destfile='transData.rda')
load('transData.rda')
p <- file.remove("transData.rda")

# read in the pathways
data("ncipid")

# read in the patient groups
data("shortlist")
data("longlist")

# use enrichment_comparison to calculate enrichment in one set of
# conditions (shortlist) and another (longlist)
short_v_long = leapR(geneset=ncipid, assay_name='transcriptomics',
             enrichment_method='enrichment_comparison',
             eset=tset, primary_columns=shortlist,
              secondary_columns=longlist)

# use enrichment_in_sets to calculate the most enriched pathways
# from the highest abundance proteins
#     from one condition
onept_sets = leapR(geneset=ncipid, assay_name='transcriptomics',
              enrichment_method='enrichment_in_sets',
              eset=tset, primary_columns="TCGA-13-1484", threshold=1.5)
```

```
# use enrichment_in_order to calculate the most enriched pathways from the
# same condition
# Note: that this uses the entire set of abundance values and their order -
# whereas the previous example uses a hard threshold to get a short list of
# most abundant proteins and calculates enrichment based on set overlap.
# The results are likely to be similar - but with some notable differences.
onept_order = leapR(geneset=ncipid, assay_name='transcriptomics',
              enrichment_method='enrichment_in_order',
              eset=tset, primary_columns="TCGA-13-1484")

# use enrichment_in_pathway to calculate the most enriched pathways in a
# set of conditions based on abundance in the pathway members versus
# abundance in non-pathway members
short_pathways = leapR(geneset=ncipid, assay_name='transcriptomics',
              enrichment_method='enrichment_in_pathway',
              eset=tset, primary_columns=shortlist)

# use correlation_enrichment to calculate the most enriched pathways in
# correlation across the shortlist conditions
short_correlation_pathways = leapR(geneset=ncipid,
               assay_name='transcriptomics',
               enrichment_method='correlation_enrichment',
               eset=tset, primary_columns=shortlist)
```

`longlist` *Long list of patient samples*

Description

Long list of patient samples

Usage

```
longlist
```

Format

An object of class `character` of length 37.

`ncipid` *NCI Gene lists*

Description

A list of pathways and the genes that comprise these pathways

Usage

```
ncipid
```

Format

A list with 4 items

names The names of the signaling pathways

desc Short description of the pathways

sizes Number of genes in the signaling pathways

matrix Matrix containing the genes in the pathways

Source

NCIPID

---

plot_leapr_bar *Plot leapR pathway bars (single panel)*

Description

This plotting helper expects leapR generated results to plot. It will use BH_pvalue if present, otherwise pvalue.

Usage

```
plot_leapr_bar(
  res_df,
  title = NULL,
  top_n = 15,
  star_thresholds = c(0.05, 0.01, 0.001),
  wrap = 42,
  max_stars = 5L,
  fill_sig = "#2C7BB6",
  fill_ns = "#BFD7FF",
  outline = NA,
  axis_text_y_size = 8,
  axis_text_x_size = 9
)
```

Arguments

res_df	A leapR df containing BH_pvalue (or pvalue) and a pathway/term label column.
title	Plot title.
top_n	Number of top pathways/genes to display.
star_thresholds	list of numeric significance thresholds for star annotations.
wrap	Wrap width for pathway labels (helps formatting).
max_stars	Maximum number of stars to draw per bar (default 5).
fill_sig	Fill color for significant bars
fill_ns	Fill color for non-significant bars.
outline	Bar border color.

axis_text_y_size
: Font size for y-axis (category) labels.

axis_text_x_size
: Font size for x-axis (numeric) labels.

Value

A **ggplot2** object (or NULL if nothing to plot).

---

read_gene_sets *read_gene_sets*

Description

read_gene_sets is a function to import external pathway database files in .gmt format

Usage

```
read_gene_sets(
  gsfile,
  gene.labels = NA,
  gs.size.threshold.min = 5,
  gs.size.threshold.max = 15000
)
```

Arguments

gsfile
: is a gene set file, for example a .gmt file (gene matrix transposed file format)

gene.labels
: defaults to NA

gs.size.threshold.min
: defaults to 5

gs.size.threshold.max
: defaults to 15000

Value

gene set object

geneset list

Examples

```
gfile <- system.file('extdata','h.all.v2024.1.Hs.symbols.gmt',
          package='leapR')
glist <- read_gene_sets(gfile)
```

`shortlist` *A list of pathways and genes that comprise these pathways from msigdb*

Description

A list of pathways and genes that comprise these pathways from msigdb

Usage

```
shortlist
```

Format

a list with 4 items

Short list of patient samples

Index