

Package ‘escape’

May 14, 2025

Title Easy single cell analysis platform for enrichment

Version 2.5.0

Description A bridging R package to facilitate gene set enrichment analysis (GSEA) in the context of single-cell RNA sequencing. Using raw count information, Seurat objects, or SingleCellExperiment format, users can perform and visualize ssGSEA, GSVA, AUCell, and UCell-based enrichment calculations across individual cells.

License MIT + file LICENSE

Encoding UTF-8

LazyData false

RoxygenNote 7.3.2

biocViews Software, SingleCell, Classification, Annotation,
GeneSetEnrichment, Sequencing, GeneSignaling, Pathways

Depends R (>= 4.1)

Imports AUCell, BiocParallel, grDevices, dplyr, ggdist, ggplot2,
ggpointdensity, GSEABase, GSVA, SingleCellExperiment, ggridges,
msigdb, stats, reshape2, patchwork, MatrixGenerics, utils,
SummarizedExperiment, UCell, stringr, methods, SeuratObject,
Matrix

Suggests Seurat, hexbin, scran, knitr, rmarkdown, markdown, BiocStyle,
RColorBrewer, rlang, spelling, testthat (>= 3.0.0), vdiffr

VignetteBuilder knitr

Language en-US

git_url <https://git.bioconductor.org/packages/escape>

git_branch devel

git_last_commit 9767695

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-05-14

Author Nick Borcharding [aut, cre],
Jared Andrews [aut],
Alexei Martsinkovskiy [ctb]

Maintainer Nick Borcharding <ncborch@gmail.com>

Contents

densityEnrichment	2
escape.gene.sets	3
escape.matrix	3
getGeneSets	4
geyserEnrichment	5
heatmapEnrichment	6
pcaEnrichment	7
performNormalization	9
performPCA	10
ridgeEnrichment	11
runEscape	12
scatterEnrichment	13
splitEnrichment	14
Index	16

densityEnrichment	<i>Visualize the mean density ranking of genes across gene set</i>
-------------------	--

Description

This function allows to the user to examine the mean ranking within the groups across the gene set. The visualization uses the density function to display the relative position and distribution of rank.

Usage

```
densityEnrichment(
  input.data,
  gene.set.use = NULL,
  gene.sets = NULL,
  group.by = NULL,
  palette = "inferno"
)
```

Arguments

input.data	The single-cell object to use.
gene.set.use	Selected individual gene set.
gene.sets	The gene set library to use to extract the individual gene set information from.
group.by	Categorical parameter to plot along the x.axis. If input is a single-cell object the default will be cluster.
palette	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object mean rank gene density across groups

Examples

```

GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small

densityEnrichment(pbmc_small,
                  gene.set.use = "Tcells",
                  gene.sets = GS)

```

escape.gene.sets	<i>Built-In Gene Sets for escape</i>
------------------	--------------------------------------

Description

A list of gene sets derived from Azizi, et al 2018 [PMID: 29961579](#)) relating to tumor immunity.

escape.matrix	<i>Calculate gene set enrichment scores</i>
---------------	---

Description

This function allows users to input both the single-cell RNA-sequencing counts and output the enrichment scores as a matrix.

Usage

```

escape.matrix(
  input.data,
  gene.sets = NULL,
  method = "ssGSEA",
  groups = 1000,
  min.size = 5,
  normalize = FALSE,
  make.positive = FALSE,
  BPPARAM = SerialParam(),
  ...
)

```

Arguments

input.data	The count matrix, Seurat, or Single-Cell Experiment object.
gene.sets	Gene sets can be a list, output from getGeneSets , or the built-in gene sets in the escape package escape.gene.sets .
method	Select the method to calculate enrichment, AUCell , GSVA , ssGSEA or UCell .
groups	The number of cells to separate the enrichment calculation.
min.size	Minimum number of gene necessary to perform the enrichment calculation

<code>normalize</code>	Whether to divide the enrichment score by the number of genes TRUE or report unnormalized FALSE .
<code>make.positive</code>	During normalization shift enrichment values to a positive range TRUE for downstream analysis or not TRUE (default). Will only be applied if normalize = TRUE .
<code>BPPARAM</code>	A <code>BiocParallel::bpparam()</code> object that for parallelization.
<code>...</code>	pass arguments to <code>AUCell GSVA</code> , <code>ssGSEA</code> , or <code>UCell</code> call

Value

matrix of enrichment scores

Author(s)

Nick Borcharding, Jared Andrews

See Also

[getGeneSets](#) to collect gene sets.

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
ES <- escape.matrix(pbmc_small,
                   gene.sets = GS,
                   min.size = NULL)
```

`getGeneSets`

Get a collection of gene sets from the msigdb

Description

This function retrieves gene sets from msigdb and caches the downloaded object for future calls. It allows subsetting by main collection (library), subcollection, or specific gene sets, and only supports human ("Homo sapiens") and mouse ("Mus musculus").

Usage

```
getGeneSets(
  species = "Homo sapiens",
  library = NULL,
  subcategory = NULL,
  gene.sets = NULL,
  version = "7.4",
  id = "SYM"
)
```

Arguments

species	The scientific name of the species of interest; only "Homo sapiens" or "Mus musculus" are supported.
library	A character vector of main collections (e.g. "H", "C5"). If provided, only gene sets in these collections are returned.
subcategory	A character vector specifying sub-collection abbreviations (e.g. "CGP", "CP:REACTOME") to further subset the gene sets.
gene.sets	A character vector of specific gene set names to select. This filter is applied after other subsetting.
version	The version of MSigDB to use (default "7.4").
id	The gene identifier type to use (default "SYM" for gene symbols).

Value

A named list of gene identifiers for each gene set.

Examples

```
## Not run:
# Get all hallmark gene sets from human.
gs <- getGeneSets(species = "Homo sapiens",
                  library = "H")

# Get a subset based on main collection and subcollection.
gs <- getGeneSets(species = "Homo sapiens",
                  library = c("C2", "C5"),
                  subcategory = "GO:BP")

## End(Not run)
```

geyserEnrichment

Generate a ridge plot to examine enrichment distributions

Description

This function allows to the user to examine the distribution of enrichment across groups by generating a ridge plot.

Usage

```
geyserEnrichment(
  input.data,
  assay = NULL,
  group.by = NULL,
  gene.set = NULL,
  color.by = "group",
  order.by = NULL,
  scale = FALSE,
  facet.by = NULL,
  palette = "inferno"
)
```

Arguments

<code>input.data</code>	Enrichment output from escape.matrix or runEscape .
<code>assay</code>	Name of the assay to plot if data is a single-cell object.
<code>group.by</code>	Categorical parameter to plot along the x.axis. If input is a single-cell object the default will be cluster.
<code>gene.set</code>	Gene set to plot (on y-axis).
<code>color.by</code>	How the color palette applies to the graph - can be "group" for a categorical color palette based on the group.by parameter or use the gene.set name if wanting to apply a gradient palette.
<code>order.by</code>	Method to organize the x-axis: "mean" will arrange the x-axis by the mean of the gene.set, while "group" will arrange the x-axis by in alphanumerical order. Using NULL will not reorder the x-axis.
<code>scale</code>	Visualize raw values FALSE or Z-transform enrichment values TRUE .
<code>facet.by</code>	Variable to facet the plot into n distinct graphs.
<code>palette</code>	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object with geyser-based distributions of selected gene.set

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                       gene.sets = GS,
                       min.size = NULL)

geyserEnrichment(pbmc_small,
                 assay = "escape",
                 gene.set = "Tcells")
```

heatmapEnrichment

Generate a heatmap to visualize enrichment values

Description

This function allows to the user to examine the heatmap with the mean enrichment values by group. The heatmap will have the gene sets as rows and columns will be the grouping variable.

Usage

```
heatmapEnrichment(
  input.data,
  assay = NULL,
  group.by = NULL,
  gene.set.use = "all",
```

```

cluster.rows = FALSE,
cluster.columns = FALSE,
facet.by = NULL,
scale = FALSE,
summary.stat = "mean",
palette = "inferno"
)

```

Arguments

input.data	Enrichment output from escape.matrix or runEscape .
assay	Name of the assay to plot if data is a single-cell object.
group.by	Categorical parameter to plot along the x.axis. If input is a single-cell object the default will be cluster.
gene.set.use	Selected gene sets to visualize. If "all" , the heatmap will be generated across all gene sets.
cluster.rows	Use Euclidean distance to order the row values.
cluster.columns	Use Euclidean distance to order the column values.
facet.by	Variable to facet the plot into n distinct graphs.
scale	Visualize raw values FALSE or Z-transform enrichment values TRUE .
summary.stat	Use 'median' or 'mean' values to display.
palette	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object with heatmap of mean enrichment values

Examples

```

GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                      gene.sets = GS,
                      min.size = NULL)

heatmapEnrichment(pbmc_small,
                  assay = "escape")

```

pcaEnrichment

Visualize the PCA of enrichment values

Description

This function allows to the user to examine the distribution of principal components run on the enrichment values.

Usage

```
pcaEnrichment(
  input.data,
  dimRed = NULL,
  x.axis = "PC1",
  y.axis = "PC2",
  facet.by = NULL,
  style = "point",
  add.percent.contribution = TRUE,
  display.factors = FALSE,
  number.of.factors = 10,
  palette = "inferno"
)
```

Arguments

<code>input.data</code>	PCA from performPCA .
<code>dimRed</code>	Name of the dimensional reduction to plot if data is a single-cell object.
<code>x.axis</code>	Component to plot on the x.axis.
<code>y.axis</code>	Component set to plot on the y.axis.
<code>facet.by</code>	Variable to facet the plot into n distinct graphs.
<code>style</code>	Return a "hex" bin plot or a "point" -based plot.
<code>add.percent.contribution</code>	Add the relative percent of contribution of the selected components to the axis labels.
<code>display.factors</code>	Add an arrow overlay to show the direction and magnitude of individual gene sets on the PCA dimensions.
<code>number.of.factors</code>	The number of gene.sets to display on the overlay.
<code>palette</code>	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object with PCA distribution

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                      gene.sets = GS,
                      min.size = NULL)

pbmc_small <- performPCA(pbmc_small,
                      assay = "escape")

pcaEnrichment(pbmc_small,
              x.axis = "PC1",
              y.axis = "PC2",
              dimRed = "escape.PCA")
```

performNormalization *Perform Normalization on Enrichment Data*

Description

This function allows users to normalize the enrichment calculations by accounting for single-cell dropout and producing positive values for downstream differential enrichment analyses. Default calculation uses will scale the enrichment values by the number of genes present from the gene set and then use a natural log transformation. A positive range values is useful for several downstream analyses, like differential evaluation for log2-fold change, but will alter the original enrichment values.

Usage

```
performNormalization(
  sc.data,
  enrichment.data = NULL,
  assay = "escape",
  gene.sets = NULL,
  make.positive = FALSE,
  scale.factor = NULL,
  groups = NULL
)
```

Arguments

sc.data	Single-cell object or matrix used in the gene set enrichment calculation in escape.matrix or runEscape .
enrichment.data	The enrichment results from escape.matrix or runEscape (optional)
assay	Name of the assay to normalize if using a single-cell object
gene.sets	The gene set library to use to extract the individual gene set information from
make.positive	Shift enrichment values to a positive range TRUE for downstream analysis or not TRUE (default).
scale.factor	A vector to use for normalizing enrichment scores per cell.
groups	the number of cells to calculate normalization on at once. chunks matrix into groups sized chunks. Useful in case of memory issues.

Value

Single-cell object or matrix of normalized enrichment scores

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
  gene.sets = GS,
  min.size = NULL)
```

```
pbmc_small <- performNormalization(pbmc_small,
                                   assay = "escape",
                                   gene.sets = GS)
```

performPCA
Perform Principal Component Analysis on Enrichment Data

Description

This function allows users to calculate the principal components for the gene set enrichment values. For single-cell data, the PCA will be stored with the dimensional reductions. If a matrix is used as input, the output is a list for further plotting. Alternatively, users can use functions for PCA calculations based on their desired workflow in lieu of using [performPCA](#), but will not be compatible with downstream [pcaEnrichment](#) visualization.

Usage

```
performPCA(
  input.data,
  assay = NULL,
  scale = TRUE,
  n.dim = 1:10,
  reduction.name = "escape.PCA",
  reduction.key = "PCA"
)
```

Arguments

<code>input.data</code>	Enrichment output from escape.matrix or runEscape .
<code>assay</code>	Name of the assay to plot if data is a single-cell object.
<code>scale</code>	Standardize the enrichment value (TRUE) or not (FALSE)
<code>n.dim</code>	The number of components to calculate.
<code>reduction.name</code>	Name of the reduced dimensions object to add if data is a single-cell object.
<code>reduction.key</code>	Name of the key to use with the components.

Value

single-cell object or list with PCA components to plot.

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                        gene.sets = GS,
                        min.size = NULL)

pbmc_small <- performPCA(pbmc_small,
                        assay = "escape")
```

ridgeEnrichment

Visualize enrichment results with a ridge plot

Description

This function allows to the user to examine the distribution of enrichment across groups by generating a ridge plot.

Usage

```
ridgeEnrichment(
  input.data,
  assay = NULL,
  group.by = NULL,
  gene.set = NULL,
  color.by = "group",
  order.by = NULL,
  scale = FALSE,
  facet.by = NULL,
  add.rug = FALSE,
  palette = "inferno"
)
```

Arguments

<code>input.data</code>	Enrichment output from escape.matrix or runEscape .
<code>assay</code>	Name of the assay to plot if data is a single-cell object.
<code>group.by</code>	Categorical parameter to plot along the x-axis. If input is a single-cell object the default will be cluster.
<code>gene.set</code>	Gene set to plot (on y-axis).
<code>color.by</code>	How the color palette applies to the graph - can be "group" for a categorical color palette based on the group.by parameter or use the gene.set name if wanting to apply a gradient palette.
<code>order.by</code>	Method to organize the x-axis: "mean" will arrange the x-axis by the mean of the gene.set, while "group" will arrange the x-axis by in alphanumerical order. Using NULL will not reorder the x-axis.
<code>scale</code>	Visualize raw values FALSE or Z-transform enrichment values TRUE .
<code>facet.by</code>	Variable to facet the plot into n distinct graphs.
<code>add.rug</code>	Add visualization of the discrete cells along the ridge plot (TRUE).
<code>palette</code>	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object with ridge-based distributions of selected gene.set

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                       gene.sets = GS,
                       min.size = NULL)

ridgeEnrichment(pbmc_small,
                 assay = "escape",
                 gene.set = "Tcells")

ridgeEnrichment(pbmc_small,
                 assay = "escape",
                 gene.set = "Tcells",
                 color.by = "Tcells")
```

runEscape

Enrichment calculation for single-cell workflows

Description

Run the escape-based gene-set enrichment calculation with Seurat or SingleCellExperiment pipelines

Usage

```
runEscape(
  input.data,
  gene.sets = NULL,
  method = "ssGSEA",
  groups = 1000,
  min.size = 5,
  normalize = FALSE,
  make.positive = FALSE,
  new.assay.name = "escape",
  BPPARAM = SerialParam(),
  ...
)
```

Arguments

<code>input.data</code>	The count matrix, Seurat, or Single-Cell Experiment object.
<code>gene.sets</code>	Gene sets can be a list, output from getGeneSets , or the built-in gene sets in the escape package escape.gene.sets .
<code>method</code>	Select the method to calculate enrichment, AUCell , GSVA , ssGSEA or UCell .
<code>groups</code>	The number of cells to separate the enrichment calculation.
<code>min.size</code>	Minimum number of gene necessary to perform the enrichment calculation
<code>normalize</code>	Whether to divide the enrichment score by the number of genes TRUE or report unnormalized FALSE .

make.positive	During normalization shift enrichment values to a positive range TRUE for downstream analysis or not TRUE (default). Will only be applied if normalize = TRUE .
new.assay.name	The new name of the assay to append to the single-cell object containing the enrichment scores.
BPPARAM	A BiocParallel::bpparam() object that for parallelization.
...	pass arguments to AUCell GSVA, ssGSEA or UCell call

Value

Seurat or Single-Cell Experiment object with escape enrichment scores in the assay slot.

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                       gene.sets = GS,
                       min.size = NULL)
```

scatterEnrichment	<i>Generate a density-based scatter plot</i>
-------------------	--

Description

This function allows to the user to examine the distribution of 2 gene sets along the x.axis and y.axis. The color gradient is generated using the a density estimate. See [ggpointdensity](#) for more information.

Usage

```
scatterEnrichment(
  input.data,
  assay = NULL,
  x.axis = NULL,
  y.axis = NULL,
  scale = FALSE,
  facet.by = NULL,
  style = "point",
  palette = "inferno"
)
```

Arguments

input.data	Enrichment output from escape.matrix or runEscape .
assay	Name of the assay to plot if data is a single-cell object.
x.axis	Gene set to plot on the x.axis.
y.axis	Gene set to plot on the y.axis. group.by parameter or use the gene.set name if wanting to apply a gradient palette.

scale	Visualize raw values FALSE or Z-transform enrichment values TRUE .
facet.by	Variable to facet the plot into n distinct graphs.
style	Return a "hex" bin plot or a "point" -based plot.
palette	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object with a scatter plot of selected gene.sets

Examples

```
GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                      gene.sets = GS,
                      min.size = NULL)

scatterEnrichment(pbmc_small,
                  assay = "escape",
                  x.axis = "Tcells",
                  y.axis = "Bcells")
```

splitEnrichment	<i>Visualize enrichment results with a split violin plot</i>
-----------------	--

Description

This function allows to the user to examine the distribution of enrichment across groups by generating a split violin plot.

Usage

```
splitEnrichment(
  input.data,
  assay = NULL,
  split.by = NULL,
  group.by = NULL,
  gene.set = NULL,
  order.by = NULL,
  facet.by = NULL,
  scale = TRUE,
  palette = "inferno"
)
```

Arguments

input.data	Enrichment output from escape.matrix or runEscape .
assay	Name of the assay to plot if data is a single-cell object.
split.by	Variable to form the split violin, must have 2 levels.

group.by	Categorical parameter to plot along the x.axis. If input is a single-cell object the default will be cluster.
gene.set	Gene set to plot (on y-axis).
order.by	Method to organize the x-axis - "mean" will arrange the x-axis by the mean of the gene.set, while "group" will arrange the x-axis by in alphanumerical order. Using NULL will not reorder the x-axis.
facet.by	Variable to facet the plot into n distinct graphs.
scale	Visualize raw values FALSE or Z-transform enrichment values TRUE .
palette	Colors to use in visualization - input any hcl.pals .

Value

ggplot2 object violin-based distributions of selected gene.set

Examples

```

GS <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
           Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))
pbmc_small <- SeuratObject::pbmc_small
pbmc_small <- runEscape(pbmc_small,
                        gene.sets = GS,
                        min.size = NULL)

splitEnrichment(pbmc_small,
                 assay = "escape",
                 split.by = "groups",
                 gene.set = "Tcells")

```

Index

densityEnrichment, [2](#)

escape.gene.sets, [3](#), [3](#), [12](#)
escape.matrix, [3](#), [6](#), [7](#), [9–11](#), [13](#), [14](#)

getGeneSets, [3](#), [4](#), [4](#), [12](#)
geyserEnrichment, [5](#)

hcl.pals, [2](#), [6–8](#), [11](#), [14](#), [15](#)
heatmapEnrichment, [6](#)

pcaEnrichment, [7](#), [10](#)
performNormalization, [9](#)
performPCA, [8](#), [10](#), [10](#)

ridgeEnrichment, [11](#)
runEscape, [6](#), [7](#), [9–11](#), [12](#), [13](#), [14](#)

scatterEnrichment, [13](#)
splitEnrichment, [14](#)