

# Package ‘escape’

June 5, 2026

**Title** Easy single cell analysis platform for enrichment

**Version** 2.9.0

**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. Alternatively, escape supports use of rank-based GSEA, such as the use of differential gene expression via fgsea.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.3.3

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

**Depends** R (>= 4.1)

**Imports** ggdist, ggplot2 (>= 3.5.0), grDevices, Matrix, MatrixGenerics,  
methods, stats, SummarizedExperiment, utils

**Suggests** AUCell, BiocParallel, BiocStyle, DelayedMatrixStats, dplyr,  
fgsea, GSEABase, ggraph, ggridges, ggpointdensity, GSVA,  
hexbin, igraph, irlba, knitr, msigdb, patchwork, rmarkdown,  
rlang, scran, SeuratObject, Seurat, SingleCellExperiment,  
spelling, stringr, testthat (>= 3.0.0), UCell

**VignetteBuilder** knitr

**Language** en-US

**BugReports** <https://github.com/BorchLab/escape/issues>

**git\_url** <https://git.bioconductor.org/packages/escape>

**git\_branch** devel

**git\_last\_commit** 15bf933

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-06-04

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

**Maintainer** Nick Borcharding <[ncborch@gmail.com](mailto:ncborch@gmail.com)>

## Contents

densityEnrichment . . . . .	2
enrichIt . . . . .	3
enrichItPlot . . . . .	5
escape.gene.sets . . . . .	6
escape.matrix . . . . .	6
getGeneSets . . . . .	8
geyserEnrichment . . . . .	9
gseaEnrichment . . . . .	11
heatmapEnrichment . . . . .	12
pcaEnrichment . . . . .	13
performNormalization . . . . .	15
performPCA . . . . .	16
ridgeEnrichment . . . . .	17
runEscape . . . . .	18
scatterEnrichment . . . . .	20
splitEnrichment . . . . .	22
<b>Index</b>	<b>24</b>

---

densityEnrichment	<i>Visualize Mean Density Ranking of Genes Across Gene Sets</i>
-------------------	---

---

### Description

This function allows the user to examine the mean ranking within 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,
  gene.sets,
  group.by = NULL,
  rug.height = 0.02,
  palette = "inferno"
)
```

### Arguments

input.data	A <a href="#">Seurat</a> object or a <a href="#">SingleCellExperiment</a> .
gene.set.use	Character. Name of the gene set to display.
gene.sets	A named list of character vectors, the result of <a href="#">getGeneSets</a> , or the built-in data object <a href="#">escape.gene.sets</a> .
group.by	Character. Metadata column used for grouping. Defaults to the Seurat/SCE ident slot when NULL.
rug.height	Numeric. Vertical spacing of the hit rug as a fraction of the y-axis. Default is 0.02.
palette	Character. Color palette name from <a href="#">hcl.pals</a> . Default is "inferno".

**Value**

A ‘patchwork’/‘ggplot2’ object.

**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)
```

---

enrichIt

*Flexible GSEA for Precomputed Gene Lists*


---

**Description**

A convenience front-end to **fgsea** that lets you point at the `avg_log2FC` and `p_val_adj` columns coming out of Seurat / DESeq2 / edgeR etc. It converts them to a signed  $-\log_{10}(p)$  ranking, filters on significance / effect size, and then runs fgsea.

**Usage**

```
enrichIt(
  input.data,
  gene.sets,
  gene_col = NULL,
  logFC_col = "avg_log2FC",
  pval_col = c("p_val_adj", "p_val"),
  ranking_fun = c("signed_log10_p", "logFC"),
  pval_cutoff = 1,
  logFC_cutoff = 0,
  minSize = 5,
  maxSize = 500,
  padjust_method = "BH",
  nproc = 0
)
```

**Arguments**

<code>input.data</code>	<p>Either:</p> <ul style="list-style-type: none"> <li>• A named numeric vector <b>already ranked</b>, or</li> <li>• A <code>data.frame</code>/<code>tibble</code> with one row per gene and columns containing log-fold-change and p-value. If the gene ID is not in <code>rownames(data)</code>, supply <code>gene_col</code>.</li> </ul>
<code>gene.sets</code>	A named list of character vectors, the result of <code>getGeneSets</code> , or the built-in data object <code>escape.gene.sets</code> .

gene_col	Character or NULL. Name of the column holding gene identifiers (ignored when they are row-names). Default is NULL.
logFC_col	Character. Column name for log-fold-change values. Default is "avg_log2FC" (matches Seurat's FindMarkers()).
pval_col	Character. Column name for p-values (or adjusted p-values). Default is c("p_val_adj", "p_val") (first match is used).
ranking_fun	Character. How to build the ranking: <ul style="list-style-type: none"> <li>"signed_log10_p" (default): <math>\text{sign}(\text{logFC}) * -\log_{10}(p)</math>.</li> <li>"logFC": Use log-fold-change values directly.</li> </ul>
pval_cutoff	Numeric. Filter genes with p-value above this threshold <b>before</b> ranking. Default is 1 (no filtering).
logFC_cutoff	Numeric. Filter genes with absolute log-fold-change below this threshold <b>before</b> ranking. Default is 0 (no filtering).
minSize	Integer. Minimum pathway size passed to <b>fgsea</b> . Default is 5.
maxSize	Integer. Maximum pathway size passed to <b>fgsea</b> . Default is 500.
padjust_method	Character. Multiple-testing correction method; any method accepted by <a href="#">p.adjust</a> . Default is "BH".
nproc	Integer. Passed to <b>fgsea</b> . Use 0 for multithread if OpenMP is available. Default is 0.

### Value

'data.frame' with the usual fgsea columns plus a convenient 'leadingEdge' character column collapsed with '\;\'.

### See Also

[fgsea](#), [getGeneSets](#), [gseaEnrichment](#)

### Examples

```
pbmc_small <- SeuratObject::pbmc_small

Seurat::Idents(pbmc_small) <- "groups"
markers <- Seurat::FindMarkers(pbmc_small,
                              ident.1 = "g1",
                              ident.2 = "g2")

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

gsea <- enrichIt(markers,
                 gene.sets = gs)
```

enrichItPlot

*Adaptive Visualisation of enrichIt Results***Description**

Create bar, dot, or network plots from [enrichIt](#) results.

**Usage**

```
enrichItPlot(
  res,
  plot.type = c("bar", "dot", "cnet"),
  top = 20,
  x.measure = "-log10(padj)",
  color.measure = x.measure,
  show.counts = TRUE,
  palette = "inferno",
  ...
)
```

**Arguments**

res	Data frame. Output from <a href="#">enrichIt</a> .
plot.type	Character. Visualization type. Options: <ul style="list-style-type: none"> <li>• "bar" (default): Horizontal bar plot.</li> <li>• "dot": Dot plot with size and color encoding.</li> <li>• "cnet": Concept network plot showing gene-pathway relationships.</li> </ul>
top	Integer. Keep the top <i>n</i> terms <b>per database</b> (ranked by adjusted p-value). Set to Inf to keep all. Default is 20.
x.measure	Character. Column in res mapped to the x-axis (ignored for "cnet"). Default is "-log10(padj)".
color.measure	Character. Column mapped to color (dot plot only). Default is same as x.measure.
show.counts	Logical. If TRUE, annotate bar plot with the Count (number of genes). Default is TRUE.
palette	Character. Color palette name from <a href="#">hcl.pals</a> . Default is "inferno".
...	Further arguments passed to <b>ggplot2</b> geoms (e.g., alpha, linewidth).

**Value**

A **ggplot2** object (bar/dot) or **ggraph** object (cnet).

**Examples**

```
## Not run:
ranks <- setNames(markers$avg_log2FC, rownames(markers))
gs <- getGeneSets("Homo sapiens", library = c("H", "C2"))
res <- enrichIt(ranks, gs)

enrichItPlot(res)
```

```
enrichItPlot(res, "dot", top=10)
enrichItPlot(res, "cnet", top=5)

## End(Not run)
```

---

escape.gene.sets      *Built-In Gene Sets for escape*

---

## Description

‘escape.gene.sets’ ships with **escape** and provides a convenient set of cell-type and pathway signatures from the scRNA-seq tumor micro-environment study by Azizi *et al.* (2018, Cell doi:10.1016/j.cell.2018.06.021). These signatures capture major immune and stromal populations observed across breast-cancer samples and serve as a lightweight default for quick testing or exploratory analyses.

## Usage

```
data("escape.gene.sets")
```

## Details

The original paper defined cell-type signatures as the top differentially expressed genes per cluster (Azizi *et al.*, Supplementary Table S3).

## Source

Supplementary Table S3 in Azizi *et al.* (2018) <<https://pubmed.ncbi.nlm.nih.gov/29961579/>>

## References

Azizi E, *et al.* **Single-cell map of diverse immune phenotypes in the breast tumor microenvironment.** *Cell* 173(5):1293-1308 (2018).

## See Also

```
[runEscape()], [escape.matrix()], [getGeneSets()]
```

---

escape.matrix      *Calculate Single-Cell Gene-Set Enrichment Scores*

---

## Description

escape.matrix() computes per-cell enrichment for arbitrary gene-set collections using one of four scoring back-ends and returns a dense numeric matrix (cells x gene-sets). The expression matrix is processed in user-defined chunks (groups) so that memory use remains predictable; each chunk is dispatched in parallel via a **BiocParallel** BPPARAM backend. Heavy engines (**GSVA**, **UCell**, **AUCell**) are loaded lazily, keeping them in the package’s **Suggests** field.

**Usage**

```
escape.matrix(
  input.data,
  gene.sets = NULL,
  method = "ssGSEA",
  groups = 1000,
  min.size = 5,
  normalize = FALSE,
  make.positive = FALSE,
  min.expr.cells = 0,
  min.filter.by = NULL,
  BPPARAM = NULL,
  ...
)
```

**Arguments**

<code>input.data</code>	A raw-counts matrix (genes x cells), a <a href="#">Seurat</a> object, or a <a href="#">SingleCellExperiment</a> . Gene identifiers must match those in <code>gene.sets</code> .
<code>gene.sets</code>	A named list of character vectors, the result of <a href="#">getGeneSets</a> , or the built-in data object <a href="#">escape.gene.sets</a> . List names become column names in the result.
<code>method</code>	Character. Scoring algorithm (case-insensitive). One of "GSVA", "ssGSEA", "UCell", or "AUCell". Default is "ssGSEA".
<code>groups</code>	Integer. Number of cells per processing chunk. Larger values reduce overhead but increase memory usage. Default is 1000.
<code>min.size</code>	Integer or NULL. Minimum number of genes from a set that must be detected in the expression matrix for that set to be scored. Default is 5. Use NULL to disable filtering.
<code>normalize</code>	Logical. If TRUE, the score matrix is passed to <a href="#">performNormalization</a> (dropout scaling and optional log transform). Default is FALSE.
<code>make.positive</code>	Logical. If TRUE <i>and</i> <code>normalize = TRUE</code> , shifts every gene-set column so its global minimum is zero, facilitating downstream log-ratio analyses. Default is FALSE.
<code>min.expr.cells</code>	Numeric. Gene-expression filter threshold. Default is 0 (no gene filtering).
<code>min.filter.by</code>	Character or NULL. Column name in <code>meta.data</code> (Seurat) or <code>colData</code> (SCE) defining groups within which the <code>min.expr.cells</code> rule is applied. Default is NULL.
<code>BPPARAM</code>	A <b>BiocParallel</b> parameter object describing the parallel backend. Default is NULL (serial execution).
<code>...</code>	Extra arguments passed verbatim to the chosen back-end scoring function ( <a href="#">gsva()</a> , <a href="#">ScoreSignatures_UCell()</a> , or <a href="#">AUCell_calcAUC()</a> ).

**Value**

A numeric matrix with one row per cell and one column per gene set, ordered as in `gene.sets`.

**Supported methods**

"GSVA" Gene-set variation analysis (Poisson kernel).

"ssGSEA" Single-sample GSEA.  
"UCell" Rank-based UCell scoring.  
"AUCell" Area-under-the-curve ranking score.

**Author(s)**

Nick Borcherding, Jared Andrews

**See Also**

[runEscape](#) to attach scores to a single-cell object; [getGeneSets](#) for MSigDB retrieval; [performNormalization](#) for the optional normalization workflow.

**Examples**

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small
es <- escape.matrix(pbmc,
                  gene.sets = gs,
                  method = "ssGSEA",
                  groups = 500,
                  min.size = 3)
```

---

getGeneSets

*Get a Collection of Gene Sets from 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 = c("Homo sapiens", "Mus musculus"),
  library = NULL,
  subcategory = NULL,
  gene.sets = NULL,
  version = "7.4",
  id = "SYM"
)
```

**Arguments**

species	Character. Species name. Either "Homo sapiens" (default) or "Mus musculus".
library	Character or NULL. Vector of main collection codes (e.g., "H", "C5"). Default is NULL (all collections).
subcategory	Character or NULL. Vector of sub-collection codes (e.g., "G0:BP"). Default is NULL (all subcategories).
gene.sets	Character or NULL. Vector of specific gene-set names. Default is NULL (all gene sets).
version	Character. MSigDB version. Default is "7.4".
id	Character. Identifier type. Default is "SYM" (gene symbols).

**Value**

A named list of character vectors (gene IDs).

**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 = "G0:BP")

## End(Not run)
```

---

geyserEnrichment

*Visualize Enrichment Distributions Using Geyser Plots*

---

**Description**

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

**Usage**

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

```
summary.stat = "mean",
palette = "inferno"
)
```

### Arguments

input.data	Output of <code>escape.matrix</code> or a single-cell object previously processed by <code>runEscape</code> .
assay	Character. Name of the assay holding enrichment scores when <code>input.data</code> is a single-cell object. Ignored otherwise.
group.by	Character. Metadata column plotted on the x-axis. Defaults to the Seurat/SCE <code>ident</code> slot when NULL.
gene.set.use	Character. Name of the gene set to display.
color.by	Character. Aesthetic mapped to point color. Options: <ul style="list-style-type: none"> <li>• "group" (default): Uses <code>group.by</code> for categorical coloring.</li> <li>• <i>gene-set name</i>: Use the same value as <code>gene.set.use</code> to obtain a numeric gradient.</li> <li>• Any other metadata column present in the data.</li> </ul>
order.by	Character or NULL. How to arrange the x-axis: <ul style="list-style-type: none"> <li>• "mean": Groups ordered by decreasing group mean.</li> <li>• "group": Natural (alphanumeric) sort of group labels.</li> <li>• NULL (default): Keep original ordering.</li> </ul>
scale	Logical. If TRUE, scores are centered and scaled (Z-score) prior to plotting. Default is FALSE.
facet.by	Character or NULL. Metadata column used to facet the plot.
summarise.by	Character or NULL. Metadata column used to summarise data before plotting.
summary.stat	Character. Method used to summarize expression within each group defined by <code>summarise.by</code> . One of: "mean" (default), "median", "max", "sum", or "geometric".
palette	Character. Color palette name from <code>hcl.pals</code> . Default is "inferno".

### Value

A `ggplot2` object.

### Examples

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small |>
  runEscape(gene.sets = gs,
            min.size = NULL)

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

gseaEnrichment

*Classical GSEA-style Running-Enrichment Plot***Description**

Produces the familiar two-panel GSEA graphic: running enrichment score (RES) plus a "hit" rug for a **single gene-set** evaluated across multiple biological groups (clusters, conditions, samples, etc.).

**Usage**

```
gseaEnrichment(
  input.data,
  gene.set.use,
  gene.sets,
  group.by = NULL,
  summary.fun = "mean",
  p = 1,
  nperm = 1000,
  rug.height = 0.02,
  digits = 2,
  BPPARAM = NULL,
  palette = "inferno"
)
```

**Arguments**

input.data	A <a href="#">Seurat</a> object or a <a href="#">SingleCellExperiment</a> .
gene.set.use	Character. Name of the gene set to display.
gene.sets	A named list of character vectors, the result of <a href="#">getGeneSets</a> , or the built-in data object <a href="#">escape.gene.sets</a> .
group.by	Character. Metadata column used for grouping. Defaults to the Seurat/SCE ident slot when NULL.
summary.fun	Character. Method used to collapse expression within each group <b>before</b> ranking. One of: "mean" (default), "median", "max", "sum", or "geometric".
p	Numeric. Weighting exponent in the KS statistic. Classical GSEA uses $p = 1$ . Default is 1.
nperm	Integer. Number of gene-label permutations per group. Default is 1000. Set to 0 to skip NES/p-value calculation.
rug.height	Numeric. Vertical spacing of the hit rug as a fraction of the y-axis. Default is 0.02.
digits	Integer. Number of decimal places displayed for ES in the legend. Default is 2.
BPPARAM	A <b>BiocParallel</b> parameter object describing the parallel backend. Default is NULL (serial execution).
palette	Character. Color palette name from <a href="#">hcl.pals</a> . Default is "inferno".

**Value**

A single 'patchwork'/'ggplot2' object

**Algorithm**

Based on Subramanian *et al.*, PNAS 2005:

1. Within every group, library-size-normalize counts to CPM.
2. Collapse gene expression with `summary.fun` (mean/median/etc.).
3. Rank genes (descending) to obtain one ordered list per group.
4. Compute the weighted Kolmogorov-Smirnov running score ( $\text{weight} = \text{lstat}^p$ ).
5. ES = maximum signed deviation of the curve.

**See Also**

[escape.matrix](#), [densityEnrichment](#)

**Examples**

```
pbmc_small <- SeuratObject::pbmc_small

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

gseaEnrichment(pbmc_small,
               gene.set.use = "Bcells",
               gene.sets    = gs,
               group.by     = "groups",
               summary.fun  = "mean",
               digits       = 3)
```

---

heatmapEnrichment

*Visualize Enrichment Value Summaries Using Heatmaps*

---

**Description**

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

**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**

<code>input.data</code>	Output of <code>escape.matrix</code> or a single-cell object previously processed by <code>runEscape</code> .
<code>assay</code>	Character. Name of the assay holding enrichment scores when <code>input.data</code> is a single-cell object. Ignored otherwise.
<code>group.by</code>	Character. Metadata column plotted on the x-axis. Defaults to the Seurat/SCE <code>ident</code> slot when NULL.
<code>gene.set.use</code>	Character vector or "all". Gene-set names to plot. Use "all" (default) to show every available gene set.
<code>cluster.rows</code>	Logical. If TRUE, rows are ordered by Ward-linkage hierarchical clustering (Euclidean distance). Default is FALSE.
<code>cluster.columns</code>	Logical. If TRUE, columns are ordered by Ward-linkage hierarchical clustering (Euclidean distance). Default is FALSE.
<code>facet.by</code>	Character or NULL. Metadata column used to facet the plot.
<code>scale</code>	Logical. If TRUE, Z-transforms each gene-set column <b>after</b> summarization. Default is FALSE.
<code>summary.stat</code>	Character. Method used to summarize expression within each group. One of: "mean" (default), "median", "max", "sum", or "geometric".
<code>palette</code>	Character. Color palette name from <code>hcl.pals</code> . Default is "inferno".

**Value**

A `ggplot2` object.

**Examples**

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small |>
  runEscape(gene.sets = gs, min.size = NULL)

heatmapEnrichment(pbmc, assay = "escape", palette = "viridis")
```

---

pcaEnrichment

*Visualize the PCA of Enrichment Values*

---

**Description**

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

**Usage**

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

**Arguments**

<code>input.data</code>	Single-cell object (Seurat / SCE) <b>or</b> the raw list returned by <a href="#">performPCA</a> .
<code>dimRed</code>	Character. Name of the dimensional-reduction slot to pull from a single-cell object. Ignored when <code>input.data</code> is the list output.
<code>x.axis</code>	Character. Name of the principal component to display on the x-axis (e.g., "PC1"). Default is "PC1".
<code>y.axis</code>	Character. Name of the principal component to display on the y-axis (e.g., "PC2"). Default is "PC2".
<code>facet.by</code>	Character or NULL. Metadata column used to facet the plot.
<code>style</code>	Character. Plot style. Options: <ul style="list-style-type: none"> <li>"point" (default): Density-aware scatter plot.</li> <li>"hex": Hexagonal binning.</li> </ul>
<code>add.percent.contribution</code>	Logical. If TRUE, include percent variance explained in axis labels. Default is TRUE.
<code>display.factors</code>	Logical. If TRUE, draw arrows for the top gene-set loadings. Default is FALSE.
<code>number.of.factors</code>	Integer. Number of loadings to display when <code>display.factors</code> = TRUE. Default is 10.
<code>palette</code>	Character. Color palette name from <a href="#">hcl.pals</a> . Default is "inferno".

**Value**

A **ggplot2** object.

**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

Scales each enrichment value by the **number of genes from the set that are expressed** in that cell (non-zero counts). Optionally shifts results into a positive range and/or applies a natural-log transform for compatibility with log-based differential tests.

## Usage

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

## Arguments

input.data	A raw-counts matrix (genes x cells), a <a href="#">Seurat</a> object, or a <a href="#">SingleCellExperiment</a> . Gene identifiers must match those in gene.sets.
enrichment.data	Matrix. Output of <a href="#">escape.matrix</a> or NULL if enrichment scores are already stored in input.data.
assay	Character. Name of the assay holding enrichment scores when input.data is a single-cell object. Default is "escape". Ignored otherwise.
gene.sets	A named list of character vectors, the result of <a href="#">getGeneSets</a> , or the built-in data object <a href="#">escape.gene.sets</a> . List names must match column names in the enrichment matrix.
make.positive	Logical. If TRUE, shifts each column so its minimum is zero. Default is FALSE.
scale.factor	Numeric vector or NULL. Optional per-cell scaling factors (length = number of cells). Use when you want external per-cell normalization factors. Default is NULL (compute from gene counts).
groups	Integer or NULL. Number of cells per processing chunk. Larger values reduce overhead but increase memory usage. Default is NULL (process all cells at once).

**Value**

If 'input.data' is an object, the same object with a new assay "<assay>\_normalized". Otherwise a matrix of normalized scores.

**Examples**

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small |>
  runEscape(gene.sets = gs,
           min.size = NULL)

pbmc <- performNormalization(pbmc,
                             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 = "escape",
  scale = TRUE,
  n.dim = 10,
  reduction.name = "escape.PCA",
  reduction.key = "escPC_"
)
```

**Arguments**

input.data	Output of <code>escape.matrix</code> or a single-cell object previously processed by <code>runEscape</code> .
assay	Character. Name of the assay holding enrichment scores when input.data is a single-cell object. Default is "escape". Ignored otherwise.
scale	Logical. If TRUE, standardizes each gene-set column before PCA. Default is TRUE.
n.dim	Integer. The number of principal components to compute and keep. Default is 10.
reduction.name	Character. Name used for the dimensional reduction slot when writing back to a Seurat/SCE object. Default is "escape.PCA".

reduction.key Character. Key prefix for the dimensional reduction when writing back to a Seurat/SCE object. Default is "escPC\_".

### Value

\*If\* 'input.data' is a single-cell object, the same object with a new dimensional-reduction slot.  
 \*Otherwise\* a list with 'PCA', 'eigen\_values', 'contribution', and 'rotation'.

### Examples

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small |>
  runEscape(gene.sets = gs,
           min.size = NULL)

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

---

 ridgeEnrichment

*Visualize Enrichment Distributions Using Ridge Plots*


---

### Description

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

### Usage

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

### Arguments

input.data	Output of <code>escape.matrix</code> or a single-cell object previously processed by <code>runEscape</code> .
gene.set.use	Character. Name of the gene set to display.
assay	Character. Name of the assay holding enrichment scores when <code>input.data</code> is a single-cell object. Ignored otherwise.
group.by	Character. Metadata column plotted on the y-axis. Defaults to the Seurat/SCE <code>ident</code> slot when NULL.

color.by	Character. Aesthetic mapped to fill color. Options: <ul style="list-style-type: none"> <li>• "group" (default): Uses group.by for categorical coloring.</li> <li>• <i>gene-set name</i>: Use the same value as gene.set.use to obtain a numeric gradient.</li> <li>• Any other metadata column present in the data.</li> </ul>
order.by	Character or NULL. How to arrange the y-axis: <ul style="list-style-type: none"> <li>• "mean": Groups ordered by decreasing group mean.</li> <li>• "group": Natural (alphanumeric) sort of group labels.</li> <li>• NULL (default): Keep original ordering.</li> </ul>
scale	Logical. If TRUE, scores are centered and scaled (Z-score) prior to plotting. Default is FALSE.
facet.by	Character or NULL. Metadata column used to facet the plot.
add.rug	Logical. If TRUE, draw per-cell tick marks underneath each ridge. Default is FALSE.
palette	Character. Color palette name from <a href="#">hcl.pals</a> . Default is "inferno".

**Value**

A [ggplot2] object.

**Examples**

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small |>
  runEscape(gene.sets = gs, min.size = NULL)

ridgeEnrichment(pbmc, assay = "escape",
                gene.set.use = "Tcells",
                group.by = "groups")
```

---

runEscape

*Calculate Enrichment Scores Using Seurat or SingleCellExperiment Objects*

---

**Description**

runEscape() is a convenience wrapper around [escape.matrix](#) that computes enrichment scores and inserts them as a new assay (default "escape") in a **Seurat** or **SingleCellExperiment** object. All arguments (except new.assay.name) map directly to their counterparts in `escape.matrix()`.

**Usage**

```
runEscape(
  input.data,
  gene.sets,
  method = c("ssGSEA", "GSVA", "UCell", "AUCell"),
  groups = 1000,
  min.size = 5,
  normalize = FALSE,
  make.positive = FALSE,
  new.assay.name = "escape",
  min.expr.cells = 0,
  min.filter.by = NULL,
  BPPARAM = NULL,
  ...
)
```

**Arguments**

<code>input.data</code>	A raw-counts matrix (genes x cells), a <a href="#">Seurat</a> object, or a <a href="#">SingleCellExperiment</a> . Gene identifiers must match those in <code>gene.sets</code> .
<code>gene.sets</code>	A named list of character vectors, the result of <a href="#">getGeneSets</a> , or the built-in data object <a href="#">escape.gene.sets</a> . List names become column names in the result.
<code>method</code>	Character. Scoring algorithm (case-insensitive). One of "GSVA", "ssGSEA", "UCell", or "AUCell". Default is "ssGSEA".
<code>groups</code>	Integer. Number of cells per processing chunk. Larger values reduce overhead but increase memory usage. Default is 1000.
<code>min.size</code>	Integer or NULL. Minimum number of genes from a set that must be detected in the expression matrix for that set to be scored. Default is 5. Use NULL to disable filtering.
<code>normalize</code>	Logical. If TRUE, the score matrix is passed to <a href="#">performNormalization</a> (dropout scaling and optional log transform). Default is FALSE.
<code>make.positive</code>	Logical. If TRUE <i>and</i> <code>normalize = TRUE</code> , shifts every gene-set column so its global minimum is zero, facilitating downstream log-ratio analyses. Default is FALSE.
<code>new.assay.name</code>	Character. Name for the assay that will store the enrichment matrix in the returned object. Default is "escape".
<code>min.expr.cells</code>	Numeric. Gene-expression filter threshold. Default is 0 (no gene filtering).
<code>min.filter.by</code>	Character or NULL. Column name in <code>meta.data</code> (Seurat) or <code>colData</code> (SCE) defining groups within which the <code>min.expr.cells</code> rule is applied. Default is NULL.
<code>BPPARAM</code>	A <b>BiocParallel</b> parameter object describing the parallel backend. Default is NULL (serial execution).
<code>...</code>	Extra arguments passed verbatim to the chosen back-end scoring function ( <a href="#">gsva()</a> , <a href="#">ScoreSignatures_UCell()</a> , or <a href="#">AUCell_calcAUC()</a> ).

**Value**

The input single-cell object with an additional assay containing the enrichment scores (cells x gene-sets). Matrix orientation follows standard single-cell conventions (gene-sets as rows inside the assay).

**Author(s)**

Nick Borcharding, Jared Andrews

**See Also**

[escape.matrix](#) for the underlying computation; [performNormalization](#) to add normalized scores; [heatmapEnrichment](#), [ridgeEnrichment](#), and related plotting helpers for visualization.

**Examples**

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

sce <- SeuratObject::pbmc_small
sce <- runEscape(sce,
                gene.sets = gs,
                method = "GSVA",
                groups = 1000,
                min.size = 3,
                new.assay.name = "escape")
```

---

scatterEnrichment

*Plot 2D Enrichment Distributions With Density or Hexplots*

---

**Description**

Visualize the relationship between two enrichment scores at single-cell resolution. By default, points are shaded by local 2-D density (`color.by = "density"`), but users can instead color by a metadata column (discrete) or by the raw gene-set scores themselves (continuous).

**Usage**

```
scatterEnrichment(
  input.data,
  assay = NULL,
  x.axis,
  y.axis,
  facet.by = NULL,
  group.by = NULL,
  color.by = c("density", "group", "x", "y"),
  style = c("point", "hex"),
  scale = FALSE,
  bins = 40,
  point.size = 1.2,
  alpha = 0.8,
  palette = "inferno",
  add.corr = FALSE
)
```

**Arguments**

input.data	Output of <code>escape.matrix</code> or a single-cell object previously processed by <code>runEscape</code> .
assay	Character. Name of the assay holding enrichment scores when <code>input.data</code> is a single-cell object. Ignored otherwise.
x.axis	Character. Gene-set name to plot on the x-axis.
y.axis	Character. Gene-set name to plot on the y-axis.
facet.by	Character or NULL. Metadata column used to facet the plot.
group.by	Character. Metadata column used when <code>color.by = "group"</code> . Defaults to the Seurat/SCE ident slot when NULL.
color.by	Character. Aesthetic mapped to point color. Options: <ul style="list-style-type: none"> <li>"density" (default): Shade points by local 2-D density.</li> <li>"group": Color by the <code>group.by</code> metadata column.</li> <li>"x": Apply a continuous gradient based on the x-axis values.</li> <li>"y": Apply a continuous gradient based on the y-axis values.</li> </ul>
style	Character. Plot style. Options: <ul style="list-style-type: none"> <li>"point" (default): Density-aware scatter plot.</li> <li>"hex": Hexagonal binning.</li> </ul>
scale	Logical. If TRUE, scores are centered and scaled (Z-score) prior to plotting. Default is FALSE.
bins	Integer. Number of hex bins along each axis when <code>style = "hex"</code> . Default is 40.
point.size	Numeric. Point size for <code>style = "point"</code> . Default is 1.2.
alpha	Numeric. Transparency for points or hexbins. Default is 0.8.
palette	Character. Color palette name from <code>hcl.pals</code> . Default is "inferno".
add.corr	Logical. If TRUE, add Pearson and Spearman correlation coefficients to the plot (top-left corner). Default is FALSE.

**Value**

A `ggplot2` object.

**Examples**

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

scatterEnrichment(
  pbmc,
  assay      = "escape",
  x.axis     = "Tcells",
  y.axis     = "Bcells",
  color.by   = "group",
  group.by   = "groups",
  add.corr   = TRUE,
  point.size = 1
)
```

splitEnrichment

*Plot Enrichment Distributions Using Split or Dodged Violin Plots***Description**

Visualize the distribution of gene set enrichment scores across groups using violin plots. When `split.by` contains exactly two levels, the function draws split violins for easy group comparison within each `group.by` category. If `split.by` has more than two levels, standard dodged violins are drawn instead.

**Usage**

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

**Arguments**

<code>input.data</code>	Output of <code>escape.matrix</code> or a single-cell object previously processed by <code>runEscape</code> .
<code>assay</code>	Character. Name of the assay holding enrichment scores when <code>input.data</code> is a single-cell object. Ignored otherwise.
<code>split.by</code>	Character. Metadata column used to split or color violins. Must contain at least two levels. If more than two levels are present, dodged violins are used instead of split violins.
<code>group.by</code>	Character. Metadata column plotted on the x-axis. Defaults to the Seurat/SCE <code>ident</code> slot when NULL.
<code>gene.set.use</code>	Character. Name of the gene set to display.
<code>order.by</code>	Character or NULL. How to arrange the x-axis: <ul style="list-style-type: none"> <li>• "mean": Groups ordered by decreasing group mean.</li> <li>• "group": Natural (alphanumeric) sort of group labels.</li> <li>• NULL (default): Keep original ordering.</li> </ul>
<code>facet.by</code>	Character or NULL. Metadata column used to facet the plot.
<code>scale</code>	Logical. If TRUE, scores are centered and scaled (Z-score) prior to plotting. Default is TRUE.
<code>palette</code>	Character. Color palette name from <code>hcl.pals</code> . Default is "inferno".

**Value**

A [ggplot2] object.

**Examples**

```
gs <- list(Bcells = c("MS4A1", "CD79B", "CD79A", "IGH1", "IGH2"),
          Tcells = c("CD3E", "CD3D", "CD3G", "CD7", "CD8A"))

pbmc <- SeuratObject::pbmc_small |>
  runEscape(gene.sets = gs, min.size = NULL)

splitEnrichment(input.data = pbmc,
                assay = "escape",
                split.by = "groups",
                gene.set.use = "Tcells")
```

# Index

## \* datasets

escape.gene.sets, 6

densityEnrichment, 2, 12

enrichIt, 3, 5

enrichItPlot, 5

escape.gene.sets, 2, 3, 6, 7, 11, 15, 19

escape.matrix, 6, 10, 12, 13, 15–18, 20–22

fgsea, 4

getGeneSets, 2–4, 7, 8, 8, 11, 15, 19

geyserEnrichment, 9

gseaEnrichment, 4, 11

hcl.pals, 2, 5, 10, 11, 13, 14, 18, 21, 22

heatmapEnrichment, 12, 20

p.adjust, 4

pcaEnrichment, 13, 16

performNormalization, 7, 8, 15, 19, 20

performPCA, 14, 16, 16

ridgeEnrichment, 17, 20

runEscape, 8, 10, 13, 16, 17, 18, 21, 22

scatterEnrichment, 20

Seurat, 2, 7, 11, 15, 19

SingleCellExperiment, 2, 7, 11, 15, 19

splitEnrichment, 22