Package: hammers (via r-universe)

October 24, 2026

Type Package

Title Utilities for scRNA-seq data analysis

Version 1.1.0

Description hammers is a utilities suite for scRNA-seq data analysis compatible with both Seurat and SingleCellExperiment. It provides simple tools to address tasks such as retrieving aggregate gene statistics, finding and removing rare genes, performing representation analysis, computing the center of mass for the expression of a gene of interest in low-dimensional space, and calculating silhouette and cluster-normalized silhouette.

License MIT + file LICENSE

Imports cluster, dplyr, ggplot2, ggrepel, grDevices, henna, LISTO, liver, rlang, scLang, stats, text2vec,

Suggests BiocStyle, knitr, qs2, rmarkdown, scater, scRNAseq, scuttle, testthat (>= 3.0.0), withr

biocViews Software, SingleCell, GeneExpression, MultipleComparison, Visualization

Encoding UTF-8

RoxygenNote 7.3.3

VignetteBuilder knitr

URL https://github.com/andrei-stoica26/hammers

BugReports https://github.com/andrei-stoica26/hammers/issues

Config/testthat/edition 3

Config/pak/sysreqs libfontconfig1-dev libfreetype6-dev libglpk-dev libicu-dev libxml2-dev zlib1g-dev

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

Date/Publication 2026-04-28 13:06:50 UTC

RemoteUrl https://github.com/bioc/hammers

RemoteRef HEAD

RemoteSha 6a2356ee43bcc0fe2ed1e2643a935ea5169a9954

Contents

addCategory . . . 2
addMetadataCategory . . . 3
addNormSilhouette . . . 4
centerOfMass . . . 5
checkGenes . . . 5
colCenters . . . 6
colsDimPlot . . . 6
computeSilhouette . . . 7
devPlot.default . . . 8
distributionPlot . . . 9
findRareGenes . . . 10
geneCellSets . . . 11
geneCenters . . . 11
genePresence . . . 12
genesDimPlot . . . 13
joinCharCombs . . . 13
keyvalMap . . . 14
normalizeSilhouette . . . 14
pointsDimPlot . . . 15
prepAlluvial . . . 16
pvalRiverPlot . . . 17
removeRareGenes . . . 18
repAnalysis . . . 19
safeMessage . . . 20
safeMinmax . . . 20

Index **22**

`addCategory` *Add a categorical column to a data frame based on another column*

Description

This function adds a categorical column to a data frame based on another column.

Usage

```
addCategory(df, col, newCol, keys, values)
```

Arguments

`df`	A data frame.
`col`	Column whose values will be used for creating the new column.
`newCol`	Column to be added.
`keys`	A list of keys. If vectors are part of the keys, each of their elements will be assigned the corresponding value.
`values`	A vector of values. Must have the same length as `keys`.

Value

A data frame with a new categorical column.

Examples

```
df <- data.frame(fruit = c('apple', 'banana', 'cherry', 'grape'))
df <- addCategory(df,
                  'fruit',
                  'color',
                  list(c('apple', 'cherry'),
                  'banana',
                  'grape'),
                  c('red', 'yellow', 'purple'))
```

`addMetadataCategory` *Add a categorical column to a Seurat metadata or SingleCellExperiment coldata*

Description

Add a categorical column to a Seurat metadata or SingleCellExperiment coldata

Usage

```
addMetadataCategory(
  scObj,
  col,
  newCol,
  keys,
  values,
  newCol2 = NULL,
  values2 = NULL
)
```

Arguments

`scObj`	A `Seurat`, `SingleCellExperiment`, `dgCMatrix` or `matrix` object.
`col`	Column whose values will be used for creating the new column.
`newCol`	Column to be added.
`keys`	A list of keys. If vectors are part of the keys, each of their elements will be assigned the corresponding value.
`values`	A vector of values. Must have the same length as `keys`.
`newCol2`	A second column to be added based on the same keys. Default is `NULL` (no second column will be added).
`values2`	A vector of values corresponding to the second column. Default is `NULL` (no second column will be added).

Value

A Seurat or SingleCellExpression object with one or two new categorical column(s) in the metadata/coldata.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
sceObj <- addMetadataCategory(sceObj, 'Cell_Cycle', 'Type',
list(c('G0', 'G1'), 'G2M', 'S'), c(2, 3, 1))
```

---

addNormSilhouette *Adds normalized silhouette column to a single-cell expression object*

Description

This function adds a normalized silhouette column to a single-cell expression object.

Usage

```
addNormSilhouette(scObj, normSilDF, normSilCol = "normSilhouette")
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
normSilDF	Normalized silhouette data frame.
normSilCol	The name of the normalized silhouette column to be added.

Value

The input object (Seurat or SingleCellExperiment) with an added metadata normalized silhouette column.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
df <- normalizeSilhouette(sceObj, 'Cluster')
sceObj <- addNormSilhouette(sceObj, df)
```

centerOfMass *Calculate the center of mass of columns*

Description

This function calculates the center of mass based on the columns of a data frame or matrix and a vector of weights.

Usage

```
centerOfMass(obj, weights)
```

Arguments

obj	A data frame or matrix.
weights	A vector of weights.

Value

A vector containing the center of mass.

Examples

```
obj <- matrix(data=c(2, 3, 1, 3, 6, 8), nrow=3, ncol=2)
weights <- c(0.8, 6, 16)
centerOfMass(obj, weights)
```

checkGenes *Check if all genes exist in the single-cell expression object*

Description

This function checks if all genes exist in the single-cell expression object.

Usage

```
checkGenes(scObj, genes)
```

Arguments

scObj	A `Seurat`, `SingleCellExperiment`, `dgCMatrix` or `matrix` object.
genes	A character vector of genes.

Value

None. This function is called for its side effect.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
checkGenes(sceObj, c('Gene_0480', 'Gene_0481', 'Gene_0482'))
```

`colCenters` *Calculate the centers of mass of metadata/coldata columns*

Description

This function calculates the centers of mass of selected metadata/coldata columns from a `Seurat` or `SingleCellExpression` object.

Usage

```
colCenters(scObj, columns)
```

Arguments

`scObj`	A `Seurat`, `SingleCellExperiment`, `dgCMatrix` or `matrix` object.
`columns`	Numeric columns.

Value

A data frame containing the coordinates of centers of mass.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
colCenters(sceObj, c('sizeFactor'))
```

`colsDimPlot` *Create a single-cell dimensionality reduction plot with added labeled points for numeric columns*

Description

This function creates a single-cell dimensionality reduction plot with added labeled points for metadata numeric columns.

Usage

```
colsDimPlot(scObj, cols, ...)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
cols	Columns whose centers of mass will be plotted.
...	Additional parameters passed to pointsDimPlot.

Value

A ggplot object.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
colsDimPlot(sceObj, c('sizeFactor', 'silhouette'))
```

---

computeSilhouette *Compute cluster silhouette for single-cell expression object*

Description

This function computes the silhouette for each cell in the Seurat or SingleCellExperiment object.

Usage

```
computeSilhouette(scObj, idClass, silCol = "silhouette")
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
idClass	Identity class. Must be present among the metadata columns of the single-cell expression object.
silCol	The name of the silhouette column to be added.

Value

The input object (Seurat or SingleCellExperiment) with an added metadata silhouette column.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
sceObj <- computeSilhouette(sceObj, 'Cluster')
```

devPlot.default *Saves plot or list of plots*

Description

This function saves a plot or list of plots as a pdf. Can also take as input a function that returns a ggplot object together with its arguments.

Usage

```
## Default S3 method:
devPlot(plotObject, ...)

## S3 method for class '`function`'
devPlot(plotObject, ...)

## S3 method for class 'ggplot'
devPlot(plotObject, ...)

## S3 method for class 'list'
devPlot(plotObject, ...)

devPlot(plotObject, ...)
```

Arguments

plotObject	A function, ggplot object, or list of ggplot objects.
...	Additional arguments.

Value

No value. This function is called for its side effect.

Examples

```
library(ggplot2)
df <- data.frame(x = c(1, 2), y = c(3, 5))
p <- ggplot(df) + geom_point(aes(x, y))
devPlot(p)

simplePlot <- function(df, title)
    return(ggplot(df) + geom_point(aes(x, y)) + ggtitle(title))

devPlot(simplePlot, df, 'Plot title')

if (file.exists('Rplots.pdf'))
    file.remove('Rplots.pdf')
if (file.exists('Rplots1.pdf'))
```

```
file.remove('Rplots1.pdf')
```

`distributionPlot` *Plot the distribution of cells across two columns*

Description

This function plots the distribution of cells across two columns.

Usage

```
distributionPlot(
  scObj,
  title = NULL,
  col1 = "seurat_clusters",
  col2 = "orig.ident",
  type = c("counts", "percs"),
  xLab = col1,
  yLab = if (type == "counts") "Count" else "Percentage",
  legendLab = col2,
  palette = "Spectral",
  legendPos = "right",
  legendTextSize = 10,
  legendTitleSize = 10,
  axisTextSize = 12,
  axisTitleSize = 12,
  sigDigits = 2
)
```

Arguments

`scObj`	A `Seurat`, `SingleCellExperiment`, `dgCMatrix` or `matrix` object.
`title`	Plot title.
`col1`	Column as string.
`col2`	Column as string.
`type`	Whether the plot should display counts ('counts', default) or percentages ('percs').
`xLab`	x axis label.
`yLab`	y axis label.
`legendLab`	Legend label.
`palette`	Color palette.
`legendPos`	Legend position.
`legendTextSize`	Legend text size.

legendTitleSize	Legend title size.
axisTextSize	Axis text size.
axisTitleSize	Axis title size.
sigDigits	Number of significant digits used by percentages displayed on the plot. Ignored if type is 'counts'.

Value

A ggplot object.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
p <- distributionPlot(sceObj, col1='Cluster', col2='Donor')
```

---

findRareGenes *Find rare genes in a Seurat or SingleCellExpression object*

Description

This function finds genes expressed in a low number of cells in a Seurat or SingleCellExpression object.

Usage

```
findRareGenes(scObj, nCells = 10)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
nCells	Minimum number of cells in which a gene must be expressed to be regarded as non-rare.

Value

A data frame with the rare genes as rownames and a single column representing their frequencies.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
df <- findRareGenes(sceObj)
```

geneCellSets *Generates cell sets for each input gene*

Description

This function constructs, for each input gene, sets of cells expressing the gene

Usage

```
geneCellSets(scObj, genes = NULL)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
genes	A character vector of genes.

Value

A named list of character vectors of cell names.

Examples

```
mat <- matrix(0, 1000, 500)
rownames(mat) <- paste0('G', seq(1000))
colnames(mat) <- paste0('C', seq(500))
mat[sample(length(mat), 70000)] <- sample(50, 70000, TRUE)
mat <- mat[paste0('G', sample(1000, 3)), ]
geneCellSets(mat)
```

geneCenters *Calculate the centers of mass of the expression of input genes*

Description

This function calculates the centers of mass of the expression of input genes.

Usage

```
geneCenters(scObj, genes)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
genes	A character vector of genes.

Value

A data frame containing the centers of mass.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
geneCenters(sceObj, c('Gene_0480', 'Gene_0481', 'Gene_0482'))
```

---

genePresence *Extract gene presence from a Seurat or SingleCellExperiment object*

Description

This function extracts the number of cells in which a gene from a Seurat or SingleCellExperiment is expressed.

Usage

```
genePresence(scObj, genes = NULL, minCutoff = NULL, maxCutoff = NULL)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
genes	Genes for which the number of cells in which the gene is expressed will be computed. If NULL (as default), all genes will be included in the assessment.
minCutoff	Minimum cutoff for gene counts. Genes with counts below this value will be omitted.
maxCutoff	Maximum cutoff for gene counts. Genes with counts above this value will be omitted.

Value

A data frame with two columns. The first column lists the genes ordered decreasingly by the number of cells in which they appear, the second lists the corresponding numbers of cells.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
df <- genePresence(sceObj)
```

genesDimPlot *Create a single-cell dimensionality reduction plot with added labeled points for genes*

Description

This function creates a single-cell dimensionality reduction plot with added labeled points for genes.

Usage

```
genesDimPlot(scObj, genes, ...)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
genes	Genes whose centers of mass will be plotted.
...	Additional parameters passed to pointsDimPlot.

Value

A ggplot object.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
genesDimPlot(sceObj, c('Gene_0364', 'Gene_0388', 'Gene_0477'))
```

joinCharCombs *Join all combinations of elements from character vectors*

Description

This function joins all combinations of elements from character vectors with a separating character.

Usage

```
joinCharCombs(..., joinChar = "_")
```

Arguments

...	Vectors passed to expandGrid.
joinChar	Character used to join combinations.

Value

A character vector.

Examples

```
joinCharCombs(c('a', 'b', 'c', 'd'), c('eee', 'ff'), c(1, 2, 3))
```

`keyvalMap` *Create a map from keys to values*

Description

This function creates a map from keys to values.

Usage

```
keyvalMap(keys, values)
```

Arguments

`keys`	A list of keys. If vectors are part of the keys, each of their elements will be assigned the corresponding value.
`values`	A vector of values. Must have the same length as `keys`.

Value

A named vector.

Examples

```
keyvalMap(list(2, c(3, 4, 5), 6, 8), c('a', 'b', 'c', 'd'))
```

`normalizeSilhouette` *Normalize silhouette by identity class for single-cell expression object*

Description

This function normalizes the already computed silhouette for each identity class in the single-cell expression object.

Usage

```
normalizeSilhouette(scObj, idClass, silCol = "silhouette")
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
idClass	Identity class. Must be present among the metadata columns of the single-cell expression object.
silCol	The name of the silhouette column.

Value

A data frame with normalized silhouettes for each unique element in the identity class.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
df <- normalizeSilhouette(sceObj, 'Cluster')
```

---

pointsDimPlot *Create a single-cell dimensionality reduction plot with added labeled points*

Description

This function creates a single-cell dimensionality reduction plot with added labeled points.

Usage

```
pointsDimPlot(
  scObj,
  title = NULL,
  pointsObj = NULL,
  alpha = 1,
  pointShape = 4,
  pointSize = 2,
  pointColor = "black",
  labelSize = 2.5,
  maxOverlaps = 30,
  ...
)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
title	Plot title.
pointsObj	A data frame or matrix of points with two columns representing x and y coordinates.

alpha	Opaqueness level.
pointShape	Point shape.
pointSize	Point size.
pointColor	Point color.
labelSize	Label size. If NULL, the points will not receive labels.
maxOverlaps	Maximum overlaps.
...	Additional parameters passed to Seurat::DimPlot.

Details

A wrapper around scLang::dimPlot.

Value

A ggplot object.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
pointsObj <- data.frame(x = c(2, 3),
y = c(1, 0),
row.names = c('P1', 'P2'))
pointsDimPlot(sceObj, pointsObj=pointsObj)
```

---

prepAlluvial *Prepare dataframe for alluvial plot*

Description

This function extracts the relevant information from a data frame and adjusts p-values to be used as weights for the alluvia.

Usage

```
prepAlluvial(
  df,
  pvalCol = "pvalAdj",
  colIndices = c(1, 2),
  weightExp = 1/2,
  pvalOffset = 1.00000023069254e-317
)
```

Arguments

df	A data frame.
pvalCol	Name of p-value column to be used by the alluvial plot.
colIndices	A vector respresenting the indices of the two categorical columns from the data frame that will be used.
weightExp	Exponent used in constructing weight from p-values.
pvalOffset	Offset used to avoid zeros inside the logarithm function.

Value

A data frame with weight scores in lieu of p-values.

Examples

```
df <- data.frame(A = c('a1', 'a2', 'a3', 'a4'),
B = c('b1', 'b2', 'b3', 'b4'),
pvalAdj = c(0.81, 1e-6, 1e-3, 0.022))
prepAlluvial(df)
```

---

pvalRiverPlot *Plot representation data frame*

Description

This function plots representation data frame as an alluvial plot.

Usage

```
pvalRiverPlot(
  df,
  pvalCol = "pvalAdj",
  colIndices = c(1, 2),
  weightExp = 1/2,
  pvalOffset = 1.00000023069254e-317,
  ...
)
```

Arguments

df	A data frame.
pvalCol	Name of p-value column to be used by the alluvial plot.
colIndices	A vector respresenting the indices of the two categorical columns from the data frame that will be used.
weightExp	Exponent used in constructing weight from p-values.
pvalOffset	Offset used to avoid zeros inside the logarithm function.
...	Additional parameters passed to henna::riverPlot

Value

A ggplot object

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
df <- repAnalysis(sceObj, 'Cluster', 'Donor')
pvalRiverPlot(df)
```

---

removeRareGenes *Remove rare genes from a Seurat or SingleCellExpression object*

Description

This function removes genes expressed in a low number of cells in a `Seurat` or `SingleCellExpression` object.

Usage

```
removeRareGenes(scObj, nCells = 10, verbose = TRUE)
```

Arguments

`scObj`	A `Seurat`, `SingleCellExperiment`, `dgCMatrix` or `matrix` object.
`nCells`	Minimum number of cells in which a gene must be expressed to be retained.
`verbose`	Logical; whether the output should be verbose.

Value

A `Seurat` or `SingleCellExpression` object with rare genes removed.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
sceObj <- removeRareGenes(sceObj, 30)
```

repAnalysis *Find the differential representation of two Seurat or SingleCellExperiment columns*

Description

This function finds the differential representation of two Seurat or SingleCellExperiment columns.

Usage

```
repAnalysis(
  scObj,
  col1 = "seurat_clusters",
  col2 = "orig.ident",
  doOverrep = TRUE,
 mtMethod = c("BY", "holm", "hochberg", "hommel", "bonferroni", "BH", "fdr", "none"),
  ...
)
```

Arguments

scObj	A Seurat, SingleCellExperiment, dgCMatrix or matrix object.
col1	Column as string.
col2	Column as string.
doOverrep	Whether to perform overrepresentation analysis (TRUE) or underrepresentation analysis (FALSE).
mtMethod	Multiple testing correction method. Choices are 'BY' (default), 'holm', hochberg', hommel', 'bonferroni', 'BH', 'fdr' and 'none'.
...	Additional parameters passed to LISTO::mtCorrectDF.

Value

An overrepresentation or underrepresentation data frame.

Examples

```
scePath <- system.file('extdata', 'sceObj.qs2', package='hammers')
sceObj <- qs2::qs_read(scePath)
repAnalysis(sceObj, 'Cluster', 'Donor')
```

safeMessage *Message an input if verbose is set to TRUE*

Description

This function messages an input if `verbose` is set to `TRUE`.

Usage

```
safeMessage(msg, verbose = TRUE)
```

Arguments

`msg`	Message
`verbose`	Whether the message should be displayed.

Value

No return value. This function is called for its side effect (messaging the input if `verbose` is set to `TRUE`).

Examples

```
safeMessage('message')
```

safeMinmax *Perform min-max normalization when possible; otherwise return a single-value vector.*

Description

This function min-max-normalizes a vector when possible, and otherwise returns a single-value vector.

Usage

```
safeMinmax(scores, safeVal = 0)
```

Arguments

`scores`	Numeric vector.
`safeVal`	Value to replace all values with when all values in the vector are the same.

Value

Min-max-normalized scores or a single-value vector.

Examples

```
safeMinmax(c(0, 3, 2, 1, 4, 5.5, 6.32, 8, 1.1))
```

Index

addCategory, 2
addMetadataCategory, 3
addNormSilhouette, 4

centerOfMass, 5
checkGenes, 5
colCenters, 6
colsDimPlot, 6
computeSilhouette, 7

devPlot *(*devPlot.default*)*, 8
devPlot.default, 8
distributionPlot, 9

findRareGenes, 10

geneCellSets, 11
geneCenters, 11
genePresence, 12
genesDimPlot, 13

joinCharCombs, 13

keyvalMap, 14

normalizeSilhouette, 14

pointsDimPlot, 15
prepAlluvial, 16
pvalRiverPlot, 17

removeRareGenes, 18
repAnalysis, 19

safeMessage, 20
safeMinmax, 20