Package ‘CSOA’

October 31, 2025

Type Package

Title Calculate per-cell gene signature scores in scRNA-seq data using cell set overlaps

Version 1.0.0

Description Cell Set Overlap Analysis (CSOA) is a tool for calculating per-cell gene signature scores in an scRNA-seq dataset. CSOA constructs a set for each gene in the signature, consisting of the cells that highly express the gene. Next, all overlaps of pairs of cell sets are computed, ranked, filtered and scored. The CSOA per-cell score is calculated by summing up all products of the overlap scores and the min-max-normalized expression of the two involved genes. CSOA can run on a Seurat object, a SingleCellExperiment object, a matrix and a dgCMatrix.

License MIT + file LICENSE

Imports bayesbio, dplyr, ggplot2, henna, kerntools, methods, qs, reshape2, rlang, Seurat, SeuratObject, SummarizedExperiment, sgof, spatstat.utils, stats, textshape, wesanderson

Encoding UTF-8

RoxygenNote 7.3.3

Suggests BiocStyle, knitr, patchwork, rmarkdown, scRNAseq, scuttle, stringr, testthat (>= 3.0.0)

biocViews Software, SingleCell, GeneSetEnrichment, GeneExpression

VignetteBuilder knitr

URL `https://github.com/andrei-stoica26/CSOA`

BugReports `https://github.com/andrei-stoica26/CSOA/issues`

Config/testthat/edition 3

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

git_branch RELEASE_3_22

git_last_commit 9b11f08

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2025-10-31

Author Andrei-Florian Stoica [aut, cre] (ORCID: <`https://orcid.org/0000-0002-5253-0826`>)

Maintainer Andrei-Florian Stoica <`andreistoica@foxmail.com`>

Contents

`attachCellScores.default` *Attach CSOA scores to object*

Description

This function attaches the data frame of CSOA scores to the input object.

Usage

```
## Default S3 method:
attachCellScores(scObj, scoreDF, ...)

## S3 method for class 'Seurat'
attachCellScores(scObj, scoreDF, ...)
```

```
## S3 method for class 'SingleCellExperiment'
attachCellScores(scObj, scoreDF, ...)

## S3 method for class 'matrix'
attachCellScores(scObj, scoreDF, ...)

## S3 method for class 'dgCMatrix'
attachCellScores(scObj, scoreDF, ...)

attachCellScores(scObj, ...)
```

Arguments

`scObj`	A Seurat object, SingleCellExperiment object, or expression matrix.
`scoreDF`	Data frame of CSOA scores.
`...`	Additional arguments.

Details

If the input object is of the Seurat or SingleCellExpression class, it will be returned with added CSOA scores. Otherwise, a list containing the expression matrix and the CSOA scores data frame will be returned.

Value

A Seurat object with CSOA scores added to metadata.

A SingleCellExperiment object with CSOA scores added to `colData`.

A list containing the expression matrix and the CSOA scores data frame.

A list containing the expression matrix and the CSOA scores data frame.

Examples

```
library(Seurat)
mat <- matrix(0, 500, 300)
rownames(mat) <- paste0('G', seq(500))
colnames(mat) <- paste0('C', seq(300))
mat[sample(8000)] <- sample(20, 8000, TRUE)
seuratObj <- CreateSeuratObject(mat)
seuratObj <- NormalizeData(seuratObj)
scores <- data.frame(CSOA = runif(300))
seuratObj <- attachCellScores(seuratObj, scores)
head(seuratObj$CSOA)
```

`basicHeatmap` *Plot a simple heatmap*

Description

This function plots a simple heatmap, with clustering but no dendograms.

Usage

```
basicHeatmap(
  mat,
  aesNames = c("x", "y", "Score"),
  title = "Heatmap",
  axisTextSize = 7,
  palType = "fillCont",
  wesPal = "Royal1",
  wesLow = 3,
  wesHigh = 2,
  ...
)
```

Arguments

`mat`	A matrix.
`aesNames`	A character vector of length 3 representing the y, x and fill aes elements.
`title`	Plot title.
`axisTextSize`	Axis text size.
`palType`	Palette type: color or fill, continuous or discrete. Accepted values are 'colorCont', 'fillCont', 'colDis' and 'fillDis'. The function shows a warning and does not change the color scheme if a different value is passed here.
`wesPal`	A wesanderson palette.
`wesLow`	Index of color marking low values.
`wesHigh`	Index of color marking high values.
`...`	Other arguments passed to `henna::centerTitle`.

Value

A ggplot object.

Examples

```
mat <- matrix(0, 10, 20)
mat[sample(length(mat), 50)] <- runif(50, max = 2.5)
basicHeatmap(mat)
```

`bfCorrectDF` *Perform multiple testing correction and filtering with Bonferroni*

Description

This function performs the Bonferroni correction for multiple testing in a dataframe column of p-values and filters the data-frame based on p-values.

Usage

```
bfCorrectDF(
  df,
  nTests = nrow(df),
  pvalThr = 0.05,
  colStr = "pval",
  newColStr = "pvalAdj"
)
```

Arguments

`df`	A dataframe with a column of p-values.
`nTests`	Number of tests. Default to the number of rows in the data frame.
`pvalThr`	p-value threshold.
`colStr`	Name of the column of p-values.
`newColStr`	Name of the column of adjusted p-values that will be created.

Value

The data frame with Benjamini-Yekutieli-corrected p-values.

`bhCorrectDF` *Perform multiple testing correction and filtering with Benjamini-Hochberg*

Description

This function performs the Benjamini-Hochberg correction for multiple testing in a dataframe column of p-values and filters the data-frame based on p-values.

Usage

```
bhCorrectDF(df, ...)
```

Arguments

`df`	A dataframe with a column of p-values.
`...`	Additional arguments passed to `fdrCorrectDF`.

Value

The data frame with Benjamini-Hochberg-corrected p-values.

`breakWeakTies` *Remove overlap pairs with low Jaccard scores*

Description

This function iteratively removes all overlap pairs with neighbor Jaccard score below a fixed cutoff until no overlap pairs can be removed. Subsequently, overlap ranks are recalculated.

Usage

```
breakWeakTies(overlapDF, cutoff = 1/3, doConnComp = FALSE)
```

Arguments

`overlapDF`	An overlap data frame.
`cutoff`	A cutoff used in the filtering of edges with low Jaccard scores.
`doConnComp`	Whether to calculate the connected components.

Details

The functions removes overlaps for which the two involved genes record too few shared neighbors—genes whose cell set significantly overlaps with those of both overlap genes.

Value

An overlap data frame in which edges with low Jaccard scores have been removed.

Examples

```
overlapDF <- data.frame(gene1=paste0('G', c(1, 3, 7, 6, 8, 2, 4, 3, 4, 5)),
gene2=paste0('G', c(2, 7, 2, 5, 4, 5, 1, 2, 2, 8)),
ratio=runif(10, 2, 10),
pval=runif(10, 0, 1e-10))
breakWeakTies(overlapDF, cutoff=0.1)
```

`byCorrectDF` *Perform multiple testing correction and filtering with Benjamini-Yekutieli*

Description

This function performs the Benjamini-Yekutieli correction for multiple testing in a dataframe column of p-values and filters the data-frame based on p-values.

Usage

```
byCorrectDF(df, ...)
```

Arguments

`df`	A dataframe with a column of p-values.
`...`	Additional arguments passed to `fdrCorrectDF`.

Value

The data frame with Benjamini-Yekutieli-corrected p-values.

`cellDistribution` *Show the distribution of cell sets among cells*

Description

This function returns a logical matrix that shows the representation of cell sets among all cells.

Usage

```
cellDistribution(cellSets, allCells)
```

Arguments

`cellSets`	A list of character vectors.
`allCells`	Names of all cells in the dataset.

Value

A logical matrix with genes as rows and cells as columns.

Examples

```
cellSets <- list(c('A', 'H', 'J'),
c('B', 'D', 'E', 'F', 'J'),
c('C', 'I', 'L'))
allCells <- LETTERS[seq(15)]
cellDistribution(cellSets, allCells)
```

`cellSetsOverlaps` *Calculates the significance of overlaps of pairs of cells sets*

Description

This function computes the statistical significance of overlaps of pairs of cell sets.

Usage

```
cellSetsOverlaps(cellSets, nCells, pairs = NULL, overlapFileName = NULL)
```

Arguments

`cellSets`	A list of character arrays.
`nCells`	The total number of cells in the Seurat object.
`pairs`	Pairs of cell sets to be assessed. If `NULL` (as default), all pairs will be assessed.
`overlapFileName`	The name of the file where the overlap data frame will be saved. This option can be used to save time when performing exploratory analyses such as trying different `jaccardCutoff` parameters in `breakWeakTies`. Default is `NULL` (the overlap data frame will not be saved).

Value

A data frame listing statistics for all cell set overlaps: cell set sizes, recorded and expected shared cells, the recorded-over-expected ratio and the hypergeometric p-value.

Examples

```
cellSets <- list(G1 = c('A', 'H', 'J'),
G2 = c('B', 'D', 'E', 'F', 'J'),
G3 = c('C', 'I', 'L'))
cellSetsOverlaps(cellSets, 40)
```

`computePairScores` *Compute aggregate gene pair scores*

Description

This function assesses the relative contribution of each gene pair to the CSOA score

Usage

```
computePairScores(
  overlapDF,
  pcPairScores,
  pairFileName = "pairs",
  keepOverlapOrder = FALSE
)
```

Arguments

`overlapDF`	An overlap data frame.
`pcPairScores`	A date frame of pair scores in each cell for each pair in the overlap data frame.
`pairFileName`	The name of the file where the pair data frame will be saved.
`keepOverlapOrder`	Whether to keep the rank-based order of overlaps in the pair score file, as opposed to changing it to a pair score-based order.

Value

A data frame with overlap and pair scores and ranks.

`computePCPairScores` *Compute per-cell gene pair scores*

Description

This function scores each gene pair corresponding to a top overlap in each cell.

Usage

```
computePCPairScores(overlapDF, normExp)
```

Arguments

`overlapDF`	An overlap data frame.
`normExp`	A min-max normalized expression matrix of the genes involved in top overlaps.

Details

The score is calculated by multiplying the overlap score with the min-max-normalized expression of the two corresponding genes.

Value

A data frame with gene pairs as rows and cells as columns.

`edgeLists.default` *Extract the edge list from overlap data frame or list of overlap data frames*

Description

This function creates a list of data frames with three columns: gene1, gene2 and group. If `overlapObj` is an overlap data frame, the groups correspond to the connected components. If it is a list of overlap data frames, the groups must be specified as groupNames.

Usage

```
## Default S3 method:
edgeLists(overlapObj, ...)

## S3 method for class 'data.frame'
edgeLists(overlapObj, ...)

## S3 method for class 'list'
edgeLists(overlapObj, groupNames, cutoff = NULL, ...)

edgeLists(overlapObj, ...)
```

Arguments

`overlapObj`	An overlap data frame or list of overlap data frames.
`...`	Additional arguments.
`groupNames`	Names of groups. If provided, must be a vector of the same length as the list of overlap data frames.
`cutoff`	Number of retained edges from each overlap data frame after refiltering. If `NULL` (as default), no refiltering will be performed.

Value

A list of data frames.

`expMat` *Extracts the data expression matrix from object*

Description

This function extracts the data expression matrix from object as a non-sparse matrix. Selected genes can be specified as input.

Usage

```
expMat(scObj, ...)

## Default S3 method:
expMat(scObj, genes = NULL, ...)

## S3 method for class 'Seurat'
expMat(scObj, ...)

## S3 method for class 'SingleCellExperiment'
expMat(scObj, ...)

## S3 method for class 'dgCMatrix'
expMat(scObj, ...)

## S3 method for class 'matrix'
expMat(scObj, ...)
```

Arguments

`scObj`	A Seurat object, SingleCellExperiment object, or expression matrix.
`...`	Additional arguments.
`genes`	Genes retained in the expression matrix. If NULL, all genes will be retained

Value

An expression matrix.

Examples

```
library(Seurat)
mat <- matrix(0, 6, 4)
mat[sample(length(mat), 7)] <- sample(3, 7, TRUE)
seuratObj <- CreateSeuratObject(counts = mat)
seuratObj <- NormalizeData(seuratObj)
expMat(seuratObj)
```

`fdrCorrectDF` *Perform multiple testing correction by controlling the false discovery rate*

Description

This function perform multiple testing correction by controlling the false discovery rate.

Usage

```
fdrCorrectDF(
  df,
  fdrControlFun,
  pvalThr = 0.05,
  colStr = "pval",
  newColStr = "pvalAdj"
)
```

Arguments

`df`	A dataframe with a column of p-values.
`pvalThr`	p-value threshold.
`colStr`	Name of the column of p-values.
`newColStr`	Name of the column of adjusted p-values that will be created.

Value

The data frame with p-values corrected using the method of choice (Benjamini-Hochberg or Benjamini-Yekutieli)

`featureWes` *A feature plot with a more distinctive color scheme.*

Description

This function customizes the appearance of `Seurat::FeaturePlot` for improved distinctiveness and aesthetics.

Usage

```
featureWes(
  seuratObj,
  feature,
  title = feature,
  idClass = NULL,
  labelSize = 3.5,
  titleSize = 12,
  wesPal = "Royal1",
  wesLow = 3,
  wesHigh = 2,
  ...
)
```

Arguments

`seuratObj`	A Seurat object.
`feature`	Seurat feature.
`title`	Plot title.
`idClass`	Column to be used for labelling. If NULL, no column-based labels will be generated.
`labelSize`	Size of labels. Ignored if idClass is NULL.
`titleSize`	Title size.
`wesPal`	A wesanderson palette.
`wesLow`	Index of color marking low values.
`wesHigh`	Index of color marking high values.
`...`	Additional arguments passed to `Seurat::FeaturePlot`.

Value

A ggplot object.

Examples

```
library(Seurat)
mat <- matrix(0, 3000, 800)
mat[sample(length(mat), 90000)] <- sample(8, 90000, TRUE)
seuratObj <- CreateSeuratObject(counts = mat)
seuratObj <- FindVariableFeatures(seuratObj, nfeatures=200)
seuratObj <- NormalizeData(seuratObj)
seuratObj <- ScaleData(seuratObj)
seuratObj <- RunPCA(seuratObj, verbose=FALSE)
seuratObj <- RunUMAP(seuratObj, dims=1:20, verbose=FALSE)
featureWes(seuratObj, 'Feature3')
```

`geneRadialPlot` *Radial plot for an overlap data frame*

Description

This function draws a radial plot for an overlap data frame to illustrate gene participation in top overlaps.

Usage

```
geneRadialPlot(
  overlapObj,
  title = "Top overlap genes plot",
  degreeLegendTitle = "Number of top overlaps",
  groupLegendTitle = "Group",
  extraCircles = 2,
  groupNames = NULL,
  cutoff = NULL,
  ...
)
```

Arguments

`overlapObj`	An overlap data frame or list of overlap data frames.
`title`	Plot title.
`degreeLegendTitle`	The title of the degree legend.
`groupLegendTitle`	The title of the group legend. If `NULL`, no groups will be distinguished.
`extraCircles`	Number of extra circles to be displayed on the plot.
`groupNames`	Names of groups. If provided, must be a vector of the same length as the list of overlap data frames.
`cutoff`	Number of retained edges from each overlap data frame after refiltering. If `NULL` (as default), no refiltering will be performed.
`...`	Additional parameters passed to `henna::radialPlot`.

Details

The function can separate genes by groups. The groups can be, for instance, different gene sets, or different connected components of the same overlap data frame. A wrapper around `henna::radialPlot`

Value

A ggplot object.

Examples

```
edgesDF <- data.frame(gene1 = paste0('G', c(1, 2, 3, 4, 7, 8, 10,
11, 11, 10, 10, 10)),
gene2 = paste0('G', c(2, 5, 1, 8, 4, 9, 12,
13, 14, 13, 16, 14)))
edgesDF <- henna::connectedComponents(edgesDF, 'group')
geneRadialPlot(edgesDF, groupLegendTitle='Component', extraCircles=1)
```

`generateOverlaps` *Generate overlaps of cell sets for input genes*

Description

This function constructs, for each gene in the expression matrix, a set of cells expressing the gene at or above the input percentile. Subsequently, overlaps of pairs of the constructed cell sets are assessed for statistical significance.

Usage

```
generateOverlaps(
  geneSetExp,
  percentile = 90,
  pairs = NULL,
  overlapFileName = NULL
)
```

Arguments

`geneSetExp`	A gene expression non-sparse matrix with the rows restricted to the genes for which cell sets will be computed.
`percentile`	A positive number under 100.
`pairs`	Pairs of cell sets to be assessed. If `NULL` (as default), all pairs will be assessed.
`overlapFileName`	The name of the file where the overlap data frame will be saved. This option can be used to save time when performing exploratory analyses such as trying different `jaccardCutoff` parameters in `breakWeakTies`. Default is `NULL` (the overlap data frame will not be saved).

Details

Wrapper around `percentileSets` and `cellSetsOverlaps`.

Value

A data frame listing statistics for all cell set overlaps

Examples

```
mat <- matrix(0, 2000, 500)
rownames(mat) <- paste0('G', seq(2000))
colnames(mat) <- paste0('C', seq(500))
mat[sample(length(mat), 270000)] <- sample(50, 270000, TRUE)
mat <- mat[paste0('G', sample(2000, 5)), ]
generateOverlaps(mat)
```

`getPairs` *Get all unordered pairs of two elements from a vector*

Description

This function returns all unorderded pairs of two elements from a vector.

Usage

```
getPairs(v)
```

Arguments

`v`	A vector.

Value

A list of vectors of length 2.

Examples

```
v <- c('ASD', 'VBN', 'HJKL')
getPairs(v)
```

`mtCorrectDF` *Perform multiple testing correction*

Description

This function performs correction for multiple testing in a dataframe column of p-values and filters the data-frame based on p-values.

Usage

```
mtCorrectDF(df, mtMethod = c("bf", "bh", "by"), ...)
```

Arguments

`df`	A dataframe with a column of p-values.
`mtMethod`	Multiple testing correction method. Options are Bonferroni ('bf'), Benjamini-Hochberg('bh'), and Benjamini-Yekutieli ('by').
`...`	Additional arguments passed to the multiple testing correction method.

Details

This function calls `bfCorrectDF`, `bhCorrectDF` or `byCorrectDF`, depending on the selected option. See their documentation for additional parameters.

`networkPlotDF` *Prepare overlap data frame for network plot*

Description

This function prepares a ranked and filtered overlap data frame for network plot.

Usage

```
networkPlotDF(overlapDF, rankCol = "rank", edgeScale = 2)
```

Arguments

`overlapDF`	Overlap data frame.
`rankCol`	Name of the rank column.
`edgeScale`	Scaling factor used in generating edge weights.

Value

A data frame ready to serve as input to networkPlot.

`overlapCutoffPlot` *Plot the selection of overlaps*

Description

This function illustrates the process of selecting the overlap rank cutoff by plotting rank frequencies against ranks and showcasing the convex hull of the rank-frequency points.

Usage

```
overlapCutoffPlot(
  overlapDF,
  title = "Overlap cutoff plot",
  palette = c("purple", "yellow"),
  hullWidth = 0.8,
  xLab = "Overlap rank",
  yLab = "Frequency",
  legendLabs = c("Accepted overlaps", "Discarded overlaps"),
  pointShape = 24,
  ...
)
```

Arguments

`overlapDF`	Processed overlap data frame created with `processOverlaps`.
`title`	Plot title.
`palette`	Color palette. Must have two colors, the first one representing accepted overlaps and the other representing discarded overlaps.
`hullWidth`	Width of the convex hull.
`xLab`	x axis label.
`yLab`	y axis label.
`legendLabs`	Legend labels.
`pointShape`	Point shape.
`...`	Additional arguments passed to `henna::hullPlot`.

Details

A wrapper around `henna::hullPlot`.

Value

A ggplot object.

Examples

```
overlapDF <- data.frame(gene1=paste0('G', c(1, 3, 7, 6, 8, 2, 4, 3, 4, 5)),
gene2=paste0('G', c(2, 7, 2, 5, 4, 5, 1, 2, 2, 8)),
rank=c(1, 2, 3, 4, 4, 6, 7, 7, 7, 10))
overlapCutoffPlot(overlapDF)
```

`overlapGenes` *Get all genes from an overlap data frame*

Description

This function gets all genes from an overlap data frame.

Usage

```
overlapGenes(overlapDF, components = NULL)
```

Arguments

`overlapDF`	Overlap data frame.
`components`	A numeric vector representing the connected components of the overlap data frame graph.

Value

A character vector of genes.

Examples

```
overlapDF <- data.frame(gene1 = paste0('G', c(1, 2, 3)),
gene1 = paste0('G', c(2, 7, 8)))
overlapGenes(overlapDF)
```

`overlapNetworkPlot` *Plot the overlaps as a network*

Description

This function plots the graph of the overlap data frame, with genes as vertices and overlaps as edges.

Usage

```
overlapNetworkPlot(
  overlapDF,
  title = "Top overlaps network plot",
  nodeColor = "orange",
  edgeColor = "green4",
  ...
)
```

Arguments

`overlapDF`	Overlap data frame.
`title`	Plot title.
`nodeColor`	The color of nodes. If `NULL`, the default `henna::networkPlot` color scheme will be used, which uses different colors for nodes belonging to different connected components.
`edgeColor`	The color of edges.
`...`	Additional parameters passed to `henna::networkPlot`.

Details

A thin wrapper around `henna::networkPlot`.

Value

An overlap network plot.

Examples

```
overlapDF <- data.frame(gene1 = paste0('G', c(1, 2, 5, 6, 7, 17)),
gene2 = paste0('G', c(2, 5, 8, 11, 11, 11)),
rank = c(1, 1, 3, 3, 3, 3))
overlapNetworkPlot(overlapDF)
```

`overlapPairs` *Extract gene pairs from overlap data frame*

Description

This function extracts the gene pairs from an overlap data frame.

Usage

```
overlapPairs(overlapDF)
```

Arguments

`overlapDF`	Overlap data frame.

Value

A list of gene pairs.

Examples

```
overlapDF <- data.frame(gene1 = paste0('G', c(1, 2, 3)),
gene1 = paste0('G', c(2, 7, 8)))
overlapPairs(overlapDF)
```

`percentileSets` *Generates cell expressing input genes at an input percentile*

Description

This function constructs, for each gene in the expression matrix, a set of cells expressing the gene at or above the input percentile.

Usage

```
percentileSets(geneSetExp, percentile = 90)
```

Arguments

`geneSetExp`	A gene expression non-sparse matrix with the rows restricted to the genes for which cell sets will be computed.
`percentile`	A positive number under 100.

Value

A named list of character vectors of length equaling the number of input genes. Each vector stores the cells expressing the gene at or above the input percentile.

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)), ]
percentileSets(mat)
```

`processOverlaps` *Process data frame of overlaps of cell sets*

Description

This function filters, ranks and scores previously generated overlaps of cell sets.

Usage

```
processOverlaps(
  overlapDF,
  mtMethod = c("by", "bh", "bf"),
  jaccardCutoff = NULL,
  osMethod = c("log", "minmax"),
  ...
)
```

Arguments

`overlapDF`	Overlap data frame.
`mtMethod`	Multiple testing correction method. Options are Bonferroni ('bf'), Benjamini-Hochberg('bh'), and the default Benjamini-Yekutieli ('by').
`jaccardCutoff`	A cutoff used in the filtering of edges with low Jaccard scores. If `NULL` (as default), no filtering of such edges will be performed.
`osMethod`	Method used to compute overlap scores. Options are "log" and "minmax".
`...`	Additional arguments passed to `mtCorrectDF`.

Details

Wrapper around `byCorrectDF`, `rankOverlaps`, `prepareFiltering`, `filterOverlaps` and `scoreOverlaps`.

If `jaccardCutoff` is not `NULL`, it also calls `breakWeakTies` between `filterOverlaps` and `scoreOverlaps`.

Value

A data frame consisting of filtered, ranked and scored cell sets overlaps

Examples

```
overlapDF <- data.frame(gene1=paste0('G',
c(1, 3, 7, 6, 8, 2, 4, 3, 4, 5)),
gene2=paste0('G',
c(2, 7, 2, 5, 4, 5, 1, 2, 2, 8)),
ratio=runif(10, 2, 10),
pval=runif(10, 0, 1e-10))
processOverlaps(overlapDF)
```

`qGrab` *Read and delete a .qs file*

Description

This functions reads a .qs file, deletes it, and returns its content.

Usage

```
qGrab(qsFile)
```

Arguments

`qsFile`	Name of .qs file with path.

Value

The content of the .qs file.

Examples

```
library(qs)
qsave(c(1, 2, 3), 'temp.qs')
qGrab('temp.qs')
```

`runCSOA` *Run the CSOA pipeline*

Description

This function generates cell set overlaps for input gene sets based on percentiles of gene expression, computes the significance of these overlaps, ranks, filters and scores the overlaps, and builds a per-cell score by summing the products of overlap scores and the min-max-normalized expression of the corresponding pairs of genes.

Usage

```
runCSOA(
  scObj,
  geneSets,
  percentile = 90,
  mtMethod = c("by", "bh", "bf"),
  jaccardCutoff = NULL,
  osMethod = c("log", "minmax"),
  overlapFileName = NULL,
  pairFileTemplate = NULL,
  keepOverlapOrder = FALSE,
  ...
)
```

Arguments

`scObj`	A Seurat object, SingleCellExperiment object, or expression matrix.
`geneSets`	Named list of character vectors of which each must contain at least two genes.
`percentile`	A positive number under 100.
`mtMethod`	Multiple testing correction method. Options are Bonferroni ('bf'), Benjamini-Hochberg('bh'), and the default Benjamini-Yekutieli ('by').
`jaccardCutoff`	A cutoff used in the filtering of edges with low Jaccard scores. If `NULL` (as default), no filtering of such edges will be performed.
`osMethod`	Method used to compute overlap scores. Options are "log" and "minmax".
`overlapFileName`	The name of the file where the overlap data frame will be saved. This option can be used to save time when performing exploratory analyses such as trying different `jaccardCutoff` parameters in `breakWeakTies`. Default is `NULL` (the overlap data frame will not be saved).
`pairFileTemplate`	Character object used in the naming of the files where the pair data frames will be saved. Default is `NULL` (the pair data frames will not be saved).
`keepOverlapOrder`	Keep the rank-based order of overlaps in the pair score file, as opposed to changing it to a pair score-based order. Ignored if `pairFileTemplate` is `NULL`.
`...`	Additional arguments.

Details

Wrapper around `expMat`, `generateOverlaps`, `scoreCells` and `attachCellScores`.

Value

An object of the same class as scObj with per-gene-set CSOA scores assigned for each cell.

Examples

```
mat <- matrix(0, 500, 300)
rownames(mat) <- paste0('G', seq(500))
colnames(mat) <- paste0('C', seq(300))
mat[sample(8000)] <- runif(8000, max=13)
```

```
genes <- paste0('G', seq(200))
mat[genes, 20:50] <- matrix(runif(200 * 31, min = 14, max = 15),
nrow = 200, ncol = 31)
geneSet1 <- paste0('G', seq(1, 150))
geneSet2 <- paste0('G', seq(50, 200))
df <- runCSOA(mat, list(a = geneSet1, b = geneSet2))
head(df)
```

`scoreCells` *Generate CSOA scores from overlap data frame and list of pairs*

Description

This function scores an overlap data frame using its associated list of pairs. The overlap data frame is split based on the overlaps corresponding to each gene set and scored, and the output is rejoined as a data frame.

Usage

```
scoreCells(
  geneSetExp,
  overlapDF,
  setPairs,
  geneSetNames,
  mtMethod = c("by", "bh", "bf"),
  jaccardCutoff = NULL,
  osMethod = c("log", "minmax"),
  pairFileTemplate = NULL,
  keepOverlapOrder = FALSE,
  ...
)
```

Arguments

`geneSetExp`	A gene expression non-sparse matrix with the rows restricted to the genes for which cell sets will be computed.
`overlapDF`	Overlap data frame.
`setPairs`	A list of overlaps corresponding to each input gene set.
`geneSetNames`	Character vector of names of gene sets.
`mtMethod`	Multiple testing correction method. Options are Bonferroni ('bf'), Benjamini-Hochberg('bh'), and the default Benjamini-Yekutieli ('by').
`jaccardCutoff`	A cutoff used in the filtering of edges with low Jaccard scores. If `NULL` (as default), no filtering of such edges will be performed.
`osMethod`	Method used to compute overlap scores. Options are "log" and "minmax".
`pairFileTemplate`	Character object used in the naming of the files where the pair data frames will be saved. Default is `NULL` (the pair data frames will not be saved).
`keepOverlapOrder`	Keep the rank-based order of overlaps in the pair score file, as opposed to changing it to a pair score-based order. Ignored if pairFileTemplate is `NULL`.
`...`	Additional arguments passed to `mtCorrectDF`.

Details

This function calls `scoreCells` to score each gene set data frame split from the full overlap data frame.

Value

A data frame whose columns correspond to the CSOA scores of the input gene sets.

Examples

```
mat <- matrix(0, 500, 300)
rownames(mat) <- paste0('G', seq(500))
colnames(mat) <- paste0('C', seq(300))
mat[sample(8000)] <- runif(8000, max=13)
genes <- paste0('G', seq(200))
mat[genes, 20:50] <- matrix(runif(200 * 31, min=14, max=15),
nrow=200, ncol=31)
geneSet1 <- paste0('G', seq(1, 150))
geneSet2 <- paste0('G', seq(50, 200))
geneSets <- list(geneSet1, geneSet2)
geneSets <- lapply(geneSets, sort)
setPairs <- lapply(geneSets, getPairs)
pairs <- Reduce(union, setPairs)
genes <- union(geneSet1, geneSet2)
mat <- mat[genes, ]
overlapDF <- generateOverlaps(mat, pairs=pairs)
scoreDF <- scoreCells(mat, overlapDF, setPairs, c('set1', 'set2'))
head(scoreDF)
```

`scoreCellsCore` *Generate CSOA scores from overlap data frame for a single gene set*

Description

This function computes per-cell CSOA scores from overlap data frame for a single gene set.

Usage

```
scoreCellsCore(
  geneSetExp,
  overlapDF,
  colStr = "CSOA",
  mtMethod = c("by", "bh", "bf"),
  jaccardCutoff = NULL,
  osMethod = c("log", "minmax"),
  pairFileName = NULL,
  keepOverlapOrder = FALSE,
  ...
)
```

Arguments

`geneSetExp`	A gene expression non-sparse matrix with the rows restricted to the genes for which cell sets will be computed.
`overlapDF`	Overlap data frame.
`colStr`	Name of column where CSOA scores will be stored.
`mtMethod`	Multiple testing correction method. Options are Bonferroni ('bf'), Benjamini-Hochberg('bh'), and the default Benjamini-Yekutieli ('by').
`jaccardCutoff`	A cutoff used in the filtering of edges with low Jaccard scores. If `NULL` (as default), no filtering of such edges will be performed.
`osMethod`	Method used to compute overlap scores. Options are "log" and "minmax".
`pairFileName`	The name of the file where the pair data frame will be saved.
`keepOverlapOrder`	Whether to keep the rank-based order of overlaps in the pair score file, as opposed to changing it to a pair score-based order.
`...`	Additional arguments passed to `mtCorrectDF`.

Value

A data frame with a column corresponding to the CSOA scores.

`scoreModules` *Run CSOA separately on the connected components of the overlap graph*

Description

This function runs CSOA on the connected components of the graph having the filtered overlaps as edges.

Usage

```
scoreModules(
  scObj,
  networkDF,
  components,
  colStrTemplate = "CSOA_component",
  ...
)
```

Arguments

`scObj`	A Seurat object, SingleCellExperiment object, or expression matrix.
`networkDF`	A data frame with `gene1`, `gene2` and component columns.
`components`	A numeric vector representing the connected components of the overlap data frame graph.
`colStrTemplate`	Character used in the naming of the component gene sets.
`...`	Additional parameters passed to `runCSOAMultiple`.

Value

An object of the same class as scObj with CSOA scores corresponding to the genes defining each connected components assigned for each cell.

Examples

```
mat <- matrix(0, 500, 300)
rownames(mat) <- paste0('G', seq(500))
colnames(mat) <- paste0('C', seq(300))
mat[sample(8000)] <- runif(8000, max=13)
genes1 <- paste0('G', seq(100))
mat[genes1, 20:50] <- matrix(runif(100 * 31, min = 14, max = 15),
nrow = 100, ncol = 31)
genes2 <- paste0('G', seq(101, 200))
mat[genes2, 70:100] <- matrix(runif(100 * 31, min = 14, max = 15),
nrow = 100, ncol = 31)
genes <- union(genes1, genes2)
mat <- mat[genes, ]
overlapDF <- generateOverlaps(mat)
overlapDF <- processOverlaps(overlapDF)
overlapDF <- henna::connectedComponents(overlapDF)
df <- scoreModules(mat, overlapDF, unique(overlapDF$component))[[2]]
head(df)
```

`wesBinaryGradient` *Adds a gradient color scale using two wesanderson colors*

Description

This function a gradient color scale to a ggplot object using a wesanderson palette, an index marking low values, and an index marking high values. The indices are used to select colors from the wesanderson palette of choice.

Usage

```
wesBinaryGradient(p, palType, wesPal = "Royal1", wesLow = 3, wesHigh = 2, ...)
```

Arguments

`p`	A ggplot object.
`palType`	Palette type: color or fill, continuous or discrete. Accepted values are 'colorCont', 'fillCont', 'colDis' and 'fillDis'. The function shows a warning and does not change the color scheme if a different value is passed here.
`wesPal`	A wesanderson palette.
`wesLow`	Index of color marking low values.
`wesHigh`	Index of color marking high values.
`...`	Arguments passed to other functions.

Value

A ggplot object with a new color scheme.

Index