Package 'SPADEVizR'

August 6, 2017

Type Package

Title SPADEVizR: an R package for Visualization, Analysis and Integration of SPADE results

Version 1.0-6

Author Guillaume GAUTREAU and Nicolas TCHITCHEK

Maintainer Nicolas TCHITCHEK <nicolas.tchitchek@gmail.com> and Guillaume GAUTREAU <guillaume.gautreau@free.fr>

Description The SPADE algorithm has been proposed as a new way to analysis high-dimensional cytometry data and to identified clusters of cells having similar phenotype. This algorithm performs a density-based down-sampling combined with an agglomerative hierarchical clustering. While SPADE offers unique opportunities for identifying cell populations, complementary approaches are needed to improve the characterization of identified cell populations. SPADEVizR is an R package designed to better visualize and analyze SPADE clustering results. This package extends the original SPADE outputs with techniques such as parallel coordinates, heatmaps, multidimensional scaling, volcano plots or streamgraph representations. Moreover, several statistical methods allow the identification of SPADE clusters with relevant biological behaviors. SPADEVizR can generate generalized linear models, Cox proportional hazards regression models and random forest models to predict biological outcomes, based on cluster abundances. SPADEVizR also has several features allowing to quantify and to visualize the quality of imported cell clustering results and can be used with results generated by algorithms different from SPADE.

License GPL-3 | file LICENSE

Imports biclust,
data.table,
diptest,
evtree,
flowCore,
ggdendro,
ggfortify,
ggplot2,
ggRandomForests,
ggrepel,
grDevices,
grid,
gridExtra,
gtable,
gtools,
igraph,
MASS,

methods,
packcircles,
plyr,
randomForestSRC,
reshape2,
survival

LazyData true

Suggests knitr

VignetteBuilder knitr

RoxygenNote 6.0.1

biocViews FlowCytometry, Visualization, StatisticalMethod, Clustering, MultidimensionalScaling, Regression

R topics documented:

`abundantClustersViewer`

Visualization of abundant clusters

Description

Generates a scatter plot representation showing for each cluster: its mean abundance and associated p-value.

This representation displays the p-value (shown as -log10(p-value)) in the X-axis and the mean of cells abundance in the Y-axis, in a two-dimensional chart. Each dot in the representation corresponds to a cell cluster, and both p-value and mean thresholds are shown using red dashed lines. Abundant Clusters are highlighted in red and labeled. The size of the dots is proportional to the total number of associated cells in the considered samples.

Usage

```
abundantClustersViewer(AC, show.cluster_sizes = TRUE,
  show.all_labels = FALSE, show.on_device = TRUE,
  max.dots_size = max(AC@cluster.size))
```

Arguments

`AC`	an object of class AC (object returned by the 'computeAC()' function)
`show.cluster_sizes`	a logical specifying if dot sizes are proportional to number of associated cells
`show.all_labels`	a logical specifying if all cluster labels must be shown. Only labels of significant clusters are displayed otherwise
`show.on_device`	a logical specifying if the ggplot representation must be displayed on the device
`max.dots_size`	a numeric specifying the number of associated cells in the largest dot

Details

By default, only significant abundant clusters are labeled. Labels for all clusters can be displayed by setting the 'show.all_labels' parameter to TRUE.

Value

a 'ggplot' object

AC-class *Abundant Clusters (AC) class definition*

Description

The 'AC' object is a S4 object containing the information related to the abundant clusters in a given biological condition. Moreover, this object contains parameters and results used in the statistical analysis.

Details

A cluster is considered as a significant abundant cluster if its associated p-value and mean are below the specific thresholds 'th.pvalue'.

The 'print()' and 'show()' can be used to display a summary of this object. Moreover all information about this object could be saved as a tab separated file using the 'export()' method. This object is returned by the 'identifyAC()' function.

Slots

`sample.names` a character vector containing the samples used to compute the abundant clusters

`cluster.size` a numeric vector containing the number of cells for each cluster

`use.percentages` a logical specifying if computation was performed on percentage of cell abundance

`method` a character containing the name of the statistical test used to identify the abundant clusters

`method.adjust` a character containing the name of the multiple correction method used (if any)

`mu` a numeric specifying the theoretical value of the one sample statistical test

`th.pvalue` a numeric value specifying the p-value threshold

`results` a data.frame containing for each cluster (first column): the mean (second column) and the standard deviation (third column) of the biological condition, the associated p-value (fourth column) and a logical (fifth column) specifying if the cluster is significantly abundant.

annotateClusters *Annotating cell clusters*

Description

This function is used to annotate cell clusters based on their marker expression categories. The annotations of cell clusters must be specifyed using a character dataframe indicating for each marker of each population the matching categorial values.

Usage

```
annotateClusters(Results, annotations, num = 5, display.annotations = TRUE)
```

Arguments

`Results`	a Results object
`annotations`	a character dataframe specifying the annotations
`num`	a numeric value specifying the number of markers expression categories to be used
`display.annotations`	a logical value indicating if a graphic must be generated to display the annotations of the cell clusters

Details

The annotation data.frame must have the annotations in rows and the markers in colomuns. Matching values must be specified as a character list of acceptable values.

Value

a Results object

`AP-class` *Abundance Profiles (AP) class definition*

Description

The 'AP' object is a S4 object containing the information related to the cluster classification based on theirs abundance profiles. Moreover, this object contains parameters and results used in the statistical analysis.

Details

Five methods are available to classify cellular clusters: 'hierarchical_k', 'hierarchical_h', 'kmeans', 'eigencell' and 'clique'. Each method can parameterized using the 'method.parameter' parameter.

The 'print()' and 'show()' can be used to display a summary of this object. Moreover all information about this object could be saved as a tab separated file using the 'export()' method. This object is returned by the 'classifyClusteringResults()' function.

Slots

`class.number` a numeric value specifying the number of clusters

`cluster.size` a numeric vector containing the number of cells for each cluster

`method` a character specifying the method used to classify cluster

`method.parameter` a named list of parameters used by the classification method

`classes` a two column dataframe with the cluster in first column and corresponding class in the second column

assignContext *Assignement of a context into a 'Results' object*

Description

This function allows to assigns three kinds of contextual information to samples: biological condition, timepoints and individuals.

Usage

```
assignContext(Results, assignments)

## S4 method for signature 'Results'
assignContext(Results, assignments)
```

Arguments

Results	a Results or Results object
assignments	a data.frame containing all samples names (in rownames) and columns providing contextual associations like "bc" (biological conditions), "tp" (biological conditions) and "ind" (individuals)

Details

The order of rownames (sample names) in this assignments dataframe will be the display order of the elements.

Value

none

biplotViewer *Visualization of marker co-expression*

Description

The Biplot Viewer aims to visualize co-expressions between 2 markers using a biplot representation. In such representation, each cell is represented by a dot which is positioned in a two-dimensional space corresponding to the marker expressions.

Usage

```
biplotViewer(Results, x.marker, y.marker, samples = NULL, clusters = NULL,
  resample.ratio = NULL, sample.merge = FALSE, show.on_device = TRUE,
  use.percentage = FALSE)
```

Arguments

Results	a 'Results' object (with 'flowset' slot not null)
x.marker	a character indicating the marker name of the first dimension
y.marker	a character indicating the marker name of the second dimension
samples	a character vector providing the sample names to used (all samples by default)
clusters	a character vector containing the clusters names to be visualized (by default all clusters will be used)
resample.ratio	a numeric ratio (between 0 and 1) specifying the down-sampling ratio to show less dots (or NULL)
sample.merge	a logical specifying if the selected samples must be merged in a single biplot
show.on_device	a logical specifying if the representation will be displayed on device
use.percentage	a logical specifying if the number of cells must be provided as percentages of parents

Details

Cells coming from specific clusters or samples can be selected using the 'clusters' and 'samples' parameters. Moreover, samples can be displayed independently (default behaviour) or merged. In order to seed-up the computations, the number of cells to display can be down-sampled using the 'resample.ratio' parameter.

Value

a 'ggplot' object

---

boxplotViewer *Visualization of cluster enrichment profiles conditions*

Description

The Boxplot Viewer aims to visualize and compare the abundances between several biological conditions for one single cluster or for a set of combined clusters.

This representation displays cell cluster abundances of each biological condition using boxplots. Additionally, the abundance density of each biological condition can be visualize using violin representation. Cluster abundance of each sample can also be visualized via colored dots.

Abundance from several clusters can be a by providing one or several cluster names to the 'clusters' parameter. Biological conditions information must be assigned to the samples in order to use this viewer (provided by the slot 'assignments' in the 'Results' object). The cell cluster abundances could be displayed as percentages or absolute numbers using the 'use.percentages' parameter (TRUE by default).

Usage

```
boxplotViewer(Results, samples = NULL, clusters = NULL,
  use.percentages = TRUE, show.legend = TRUE, show.violin = TRUE,
  show.on_device = TRUE)
```

Arguments

Results	a 'Results' object
samples	a character vector providing the sample names to used (all samples by default)
clusters	a character vector containing the clusters names to be visualized (required)
use.percentages	a logical specifying if the visualization must be performed on percentage
show.legend	a logical specifying if the legend must be displayed
show.violin	a logical specifying if the count distribution must be displayed
show.on_device	a logical specifying if the representation will be displayed on device

Details

Biological conditions are specified in the Results object at the importation or using the 'assignContext()' function. Cells clusters are colored based on theirs associated biological samples.

Value

a 'ggplot' object

CC-class *Correlated Clusters (CC) class definition*

Description

The 'CC' object is a S4 object containing coefficient of correlation associated between each cluster to a phenotypic or functional variable. Moreover, this object contains parameters and results used in the statistical analysis.

Details

A cluster is considered as a significant correlated cluster if its associated p-value and correlation threshold are below the specific thresholds 'th.pvalue' and 'th.correlation'.

The 'print()' and 'show()' can be used to display a summary of this object. Moreover all information about this object could be saved as a tab separated file using the 'export()' method. This object is returned by the 'identifyCC()' function.

Slots

`sample.names` a character vector containing the samples used to compute correlated clusters

`variable` a numeric vector containing the expression values of the associated variable

`cluster.size` a numeric vector containing the number of cells for each cluster

`use.percentages` a logical specifying if computation was performed on percentage of cell abundance

`method` a character containing the name of the statistical test used to identify the CC

`method.adjust` a character containing the name of the multiple correction method used (if any)

`th.correlation` a numeric value specifying the correlation threshold (R)

`th.pvalue` a numeric value specifying the p-value threshold

`results` a data.frame containing for each cluster (first column): the coefficient of correlation R (second column) , the associated p-value (third column) and a logical (fourth column) specifying if the cluster is significantly correlated.

`circlesPackingViewer` *Visualization of abundance profiles classification*

Description

Generates a colored circle packing representation showing for each cluster: its associated class based on its abundance profile.

Usage

```
circlesPackingViewer(AP, show.cluster_sizes = TRUE, show.on_device = TRUE)
```

Arguments

`AP`	an object of class 'AP' (object returned by the 'classifyAbundanceProfiles()' function)
`show.cluster_sizes`	a logical specifying if dot sizes are proportional to number of associated cells
`show.on_device`	a logical specifying if the representation will be displayed on device

Details

The sizes of the dots are proportionals to the total number of associated cells to each cluster. Circle packing classes are sorted by the number of clusters in each class.

Value

a 'ggplot' object

`classifyAbundanceProfiles` *Classification of clustering results based on the abundance profiles*

Description

Classifies clusters based on their abundance profiles (number of cells for each cluster).

Usage

```
classifyAbundanceProfiles(Results, method = "hierarchical_h",
  method.parameter = NULL)
```

Arguments

`Results`	a 'Results' object
`method`	a character specifying the clustering method among: "hierarchical_h", "hierarchical_k","k-means","eigencell","clique"
`method.parameter`	a numeric specifying the numeric value required by the selected method

Details

The classification is done on cell abundances of each clusters and could be performed using 5 methods:

- "hierarchical_k" This method first compute the Pearson correlation matrix and then use this matrix to performs a hierarchical classification. The hierarchical classification is cut in order to return the desired number of classes. This number of classes must be provided as a numeric integer using the 'method.parameter' parameter. It is to note that negative correlations are considered as uncorrelated
- "hierarchical_h" (default method) This method works in the same way than 'hierarchical_k' but the height where the hierarchical tree is specified. This height is a correlation threshold (a numeric double between 0 and 1 included, default is 0.7) provided using the 'method.parameter' parameter.
- "kmeans" This method works as described in the R stats documentation (?kmeans) using the 'method.parameter' parameter to specify the desired number of classes.
- "eigencell" This method performs an eigen vector decomposition and then calculate the correlations between cluster values and these vectors. Clusters which correlate above a specific threshold with the same eigen vector are classified together. This correlation threshold (a numeric double between 0 and 1 included, default is 0.8) provided using the 'method.parameter' parameter.
- "clique" This method first computes the Pearson correlation matrix and then use this matrix to generate an undirected graph. In this graph, an edge is drawn between two nodes if the correlation coefficient in the adjacency matrix is above a specific threshold. This correlation threshold (a numeric double between 0 and 1 included, default is 0.7) provided using the 'method.parameter' parameter. After building the graph, the method looking for the largest cliques which are considered as classes of nodes. Cliques correspond to subgraph in which every two distinct vertices are adjacent.

Value

a S4 object of class 'AP'

`correlatedClustersViewer` *Visualization of correlated clusters*

Description

Generates a volcano plot representation showing for each cluster: its correlation coefficient and associated p-value.

This representation displays the p-value (shown as -log10(p-value)) in the Y-axis and the correlation coefficient in the X-axis, in a two-dimensional chart. Each dot in the representation corresponds to a cell cluster, and both correlation coefficient (positive and negative) and p-value thresholds are shown using red dashed lines. Correlated Clusters are highlighted in red and labeled. The size of dots is proportional to the total number of associated cells in the considered samples.

Usage

```
correlatedClustersViewer(CC, show.cluster_sizes = TRUE,
  show.all_labels = FALSE, show.on_device = TRUE,
  max.dots_size = max(CC@cluster.size), varname = NULL)
```

Arguments

CC	an object of class 'CC' (object returned by the 'computeCC()' function)
show.cluster_sizes	a logical specifying if dot sizes are proportional to number of associated cells
show.all_labels	a logical specifying if all cluster labels must be shown. Only labels of significant clusters are displayed otherwise
show.on_device	a logical specifying if the ggplot representation must be displayed on the device
max.dots_size	a numeric specifying the number of associated cells in the largest dot
varname	a character indicating the name of the variable to display in the title

Details

By default, only significant correlated clusters are labeled. Labels for all clusters can be displayed by setting the 'all.label' parameter to TRUE.

Value

a 'ggplot' object

---

countViewer *Visualization of cluster sizes*

Description

The Count Viewer aims to visualize the number of cells associated to each cluster. This representation displays the clusters in the X-axis and the total number of associated cells in the Y-axis. Additionally, the numbers of cells associated to each cluster for each sample are also displayed using a jitter representation. The size of the dots is proportional to the total number of associated cells.

Usage

```
countViewer(Results, samples = NULL, clusters = NULL, min.cells = 0,
  sort = TRUE, max.dots_size = NULL, show.samples = TRUE,
  show.on_device = TRUE)
```

Arguments

Results	a 'Results' object
samples	a character vector providing the sample names to used (all samples by default)
clusters	a character vector containing the clusters names to be visualized (by default all clusters will be displayed)
min.cells	a numeric specifying the minimum number of cell (sum of all selected samples) to display a cluster
sort	a logical specifying if the clusters will be to be sorted (descending) based on the sum of all selected samples for each cluster
max.dots_size	a numeric specifying the number of cells in the largest dot
show.samples	a logical specifying if the number of cells for all selected samples will be displayed
show.on_device	a logical specifying if the representation will be displayed on device

Details

By default, all clusters will be displayed but the representation can be restricted to a set of selected samples (using the 'samples' parameter) or to a set of selected clusters (using the 'clusters' parameter).

Value

a 'ggplot' object

---

createReport *Generating SPADEVizR reports*

Description

Generates a customizable PDF report based on SPADEVizR visualization features.

Usage

```
createReport(Results, PDFfile = "report.pdf", select.plots = c("count",
  "heatmap", "MDSclusters", "pheno"), clusters = NULL, markers = NULL,
  samples = NULL, stat.objects = list(), width = 30, height = 10,
  verbose = TRUE)
```

Arguments

Results	a 'Results' object
PDFfile	a character specifying the output path
select.plots	a vector combining character and stat objects ('AC', 'DAC', 'CC' and 'AP') specifying the order of the desired plots (see details)
clusters	a character vector of clusters to include in the report (all will be included by default)
markers	a character vector of markers to include in the report (all will be included by default)
samples	a character vector providing the sample names to used (all samples by default)
stat.objects	a list containing one or several AC, DEC, CC or AP objects to plot in the report
width	a numeric specifying the width of the PDF file
height	a numeric specifying the height of the PDF file
verbose	a boolean specifying if some messages must be displayed during the generation of the report

Details

Available plots are :

- "count" (included by default):Displays a Count Viewer representation;
- "tree":Displays a SPADE Tree Viewer representation;
- "heatmap" (included by default):Displays an Heatmap Viewer representation;
- "boxplot":Displays a Boxplot Viewer representation;
- "kinetics":Displays a Kinetic Viewer representation;
- "streamgraph":Display a Streamgraph Viewer representation;
- "pheno" (included by default):Displays a Pheno Viewer representation;
- "MDSclusters" (included by default):Displays a MDS Viewer representation at the cluster level
- "MDSsamples":Displays a MDS Viewer representation at the sample level
- "distogram":Displays a Distogram Viewer representation;
- "kinetics_pheno":Displays two "kinetics" and "cluster" representations juxtaposed (one on the side of the other) for each cluster;
- "boxplot_pheno":Displays two "boxplot" and "cluster" representations juxtaposed (one on the side of the other) for each cluster;
- AC, DAC, CC and AP objects

Value

none

DAC-class *Differentially Abundant Clusters (DAC) class definition*

Description

The 'DAC' object is a S4 object containing the information related to the differentially abundant clusters between two given biological conditions. Moreover, this object contains parameters and results used in the statistical analysis.

Details

A cluster is considered as a differentially enriched cluster if its associated p-value and fold-change are below the specific thresholds 'th.pvalue' and 'th.fc'.

The 'print()' and 'show()' can be used to display a summary of this object. Moreover all information about this object could be saved as a tab separated file using the 'export()' method. This object is returned by the 'identifyDAC()' function.

Slots

`sample.cond1` a character specifying the names of the samples of the first biological condition

`sample.cond2` a character specifying the names of the samples of the second biological condition

`cluster.size` a numeric vector containing the number of cells for each cluster

`use.percentages` a logical specifying if computation was performed on percentage of cell abundance

`method` a character containing the name of the statistical test used to identify the DAC

`method.adjust` a character containing the name of the multiple correction method used (if any)

`method.paired` a logical indicating if the statistical test have been performed in a paired manner

`th.fc` a numeric value specifying the fold-change threshold

`th.pvalue` a numeric value specifying the p-value threshold

`results` a data.frame containing for each cluster (first column): the fold-change (second column) and the standard deviation (third column) for the first biological condition, the fold-change (fourth column) and the standard deviation (fifth column) for the second biological condition, the associated p-value (sixth column) and a logical (seventh column) specifying if the cluster is significantly differentially abundant.

distogramViewer *Visualization of marker co-expressions*

Description

The Distogram Viewer aims to visualize the pairwise co-expressions between all markers using a distogram representation. In such representation, each tile corresponds to the co-expression between 2 markers and is gradient-colored based on the Pearson correlation between the expressions of those 2 markers (as stored in the phenotype matrix). Markers used as clustering markers are shown in blue.

Usage

```
distogramViewer(Results, samples = NULL, clusters = NULL, markers = NULL,
  show.on_device = TRUE)
```

Arguments

Results	a 'Results' object
samples	a character vector providing the sample names to used (all samples by default)
clusters	a character vector containing the clusters names to be use (by default all clusters will be used)
markers	a character vector specifying the markers to be displayed
show.on_device	a logical specifying if the representation will be displayed on device

Details

The Pearson correlation is computed based on the marker expressions. The visualization can be restricted to specific clusters, samples and markers by using respectively the 'clusters', 'samples' and 'markers' parameters.

Value

a 'ggplot' object. This object also contains the correlations for each pair of marker (cor attribute).

---

export *Exportation of SPADEVizR objects*

Description

Exports a SPADEVizR object into a tab separated file.

Usage

```
export(object, filename = "export.txt")

## S4 method for signature 'Results'
export(object, filename = "export.txt")

## S4 method for signature 'AC'
export(object, filename = "export.txt")

## S4 method for signature 'DAC'
export(object, filename = "export.txt")

## S4 method for signature 'CC'
export(object, filename = "export.txt")

## S4 method for signature 'AP'
export(object, filename = "export.txt")
```

Arguments

object	a SPADEVizR object
filename	a character indicating the location of the output file

Value

none

`generateCPHM` *Generation of a Cox proportional hazards regression model of cluster abundances*

Description

This function generates a Cox proportional hazards regression model between a provided vector of values associated to samples and the abundance of each clusters. This generated model can then be used to predict biological outcomes in the context of survival/progression studies.

Usage

```
generateCPHM(Results, variable, status, use.percentages = FALSE,
  clusters = NULL, th.pvalue = 1, show.error = FALSE, ...)
```

Arguments

Results	a 'Results' object
variable	a numerical named vector providing the correspondence between sample names and specific phenotypes (or NA values to infer the phenotypes)
status	a numeric vector providing the correspondence between sample names and specific status (or NA values to infer the status)
use.percentages	a logical specifying if the computations should be performed on percentage
clusters	a character vector specifying the names of the clusters used to compute the Cox model (all clusters are selected by default)
th.pvalue	a numeric between 0 and 1 specifying the maximal p-value of each term in the returned model
show.error	a logical indicating if error bars should be used to display the coefficients standard deviations
...	further parameters passed to the R coxph method

Details

The 'clusters' parameter provides the name of the clusters to include in the model.

The number of clusters allowed in the model can vary depending the number of values to infer. Please refer to the documentation of R coxph and coxph.predict functions for more details.

Firstly, the function computed the glm model using all clusters as terms of the model. Then, the model is iteratively regenerated by discarding at each step the terms with the highest p-value higher than the 'th.pvalue' thershold. In this way, the model can correctly fit the data while being parsimonious. By default, the 'p-value' thershold parameter is set to 1 in order to include all terms in the model. If no terms having a p-value below the threshold, both the returned model and the prediction are set to NULL.

Value

a list of 5 elements corresponding to: a Cox proportional hazards regression model object as provided by the R coxph function ('model' element), a named vector of predicted values for each sample ('variable.predictions' element), the representation of clusters coeficients ('plot.clusters' element), and the representation of samples predictions values ('plot.samples' element), and a representation of the provided survival curve ('plot.provided_survival_curve' element).

`generateGLM` *Generation a generalized linear models of cluster abundances*

Description

This function generates a generalized linear model between a provided vector of values associated to samples and the abundance of each cluster.

Usage

```
generateGLM(Results, variable, use.percentages = FALSE, clusters = NULL,
  th.pvalue = 1, show.error = FALSE, verbose = FALSE, ...)
```

Arguments

`Results`	a 'Results' object
`variable`	a numerical named vector providing the correspondence between sample names and specific phenotypes (or NA values to infer the phenotypes)
`use.percentages`	a logical specifying if the computations should be performed on percentage
`clusters`	a character vector specifying the names of the clusters used to compute the linear model (all clusters are selected by default)
`th.pvalue`	a numeric between 0 and 1 specifying the maximal p-value of each term in the returned model
`show.error`	a logical indicating if error bars should be used to display the coefficients standard deviations
`verbose`	a logical indicating if debug messages must be displayed
`...`	further parameters passed to the R glm method

Details

The 'clusters' parameter provides the name of the clusters to include in the model.

The number of clusters allowed in the model can vary depending the number of values to infer. Please refer to the documentation of R coxph and coxph.predict functions for more details.

Firstly, the function computed the glm model using all clusters as terms of the model. Then, the model is iteratively regenerated by discarding at each step the terms with the highest p-value higher than the 'th.pvalue' thershold. In this way, the model can correctly fit the data while being parsimonious. By default, the 'p-value' thershold parameter is set to 1 in order to include all terms in the model. If no terms having a p-value below the threshold, both the returned model and the prediction are set to NULL.

Value

a list of 5 elements corresponding to: a generalized linear model object as provided by the R glm function ('model' element), a named vector of predicted values ('variable.predictions' element), a named vector of predicted cluster abundance coefficiants ('cluster.coefficients' element), the representation of clusters coeficients ('plot.cluster' element), and the representation of samples predictions values ('plot.samples' element).

`generateRFM` *Generation of a random forest of cluster abundances*

Description

This function generates a random forest model between the values associated to samples and the abundance of each clusters. This generated model can then be used to predict biological outcomes in the context of survival/progression studies.

Usage

```
generateRFM(Results, variable, status, use.percentages = FALSE,
  clusters = NULL, ntree = 1000, ...)
```

Arguments

`Results`	a 'Results' object
`variable`	a numerical named vector providing the correspondence between sample names and specific phenotypes (or NA values to infer the phenotypes)
`status`	a numerical named vector providing the correspondence between sample names and specific status (or NA values to infer the status)
`use.percentages`	a logical specifying if the computations should be performed on percentage
`clusters`	a character vector specifying the names of the clusters used to compute the model (all clusters are selected by default)
`ntree`	a numerical value specifying the number of tree to generate
`...`	further parameters passed to the R randomForest method

Details

The 'clusters' parameter provide the name of the clusters to include in the model.

The named vector containg the value associated to each sample are provided to the 'variable' parameter. In order to infer unknown values associated to samples, it is possible to set to NA for these samples. In this way, the function will infer, based on the computed model, all values associated to these samples.

The function involve the randomForestSRC to compute a random forest model using all clusters as variable of the model. Several visualisation based on the "ggfortify" package are returned allowing to identify the most important variables, their minimum depth, the OOB Error Rate, as well as the variables predictions.

Value

a list of 4 elements corresponding to: random forest model object as provided by the R randomForest function ('model' element), and a named vector of predicted values ('variable.predictions' element), the representation of clusters coeficients ('plot.vimp' element), and the representation of samples predictions values ('plot.samples' element)

`heatmapViewer` *Visualization of all clusters phenotypes using categorical heatmap*

Description

The Heatmap Viewer aims to visualize the phenotypes of all cell clusters. This representation displays marker expressions of all clusters using a categorical heatmap (5 categories are computed by default). The range expression of each cell marker is discretized in several categories between bounds of marker expressions. Each marker of each cluster is then categorized into one category based on the mean of median marker expressions. Markers used as clustering markers are shown in blue.

Usage

```
heatmapViewer(Results, clusters = NULL, markers = NULL, num = 5,
  clustering = "both", tile.color = "black", show.on_device = TRUE)
```

Arguments

`Results`	a 'Results' object
`clusters`	a character vector providing the clusters to be used (all clusters by default)
`markers`	a character vector providing the markers to be used (all markers by default)
`num`	a numeric value specifying the number of markers expression categories to be used
`clustering`	a character specifying which clustering must be build ("markers", "clusters", "both" or "none")
`tile.color`	a character specifying the border color of the tiles (NA to remove tile borders)
`show.on_device`	a logical specifying if the representation will be displayed on device

Details

Both cell clusters and cell markers can be clustered using a hierarchical clustering (using the 'dendrograms', generating marker and cluster dendrograms by default).

It is to note that the Heatmap Viewer is the default viewer for 'Results' objects.

The structures of the marker and cluster hierarchical clustering dendrograms are also returned by this fucntion.

Value

a 'ggplot' object. This object also contains the categorical phenotipical heatmap (pheno.table attribute) and the marker and cluster hierarchical clustering dendrograms (markers.hc and clusters.hc attributes)

identifyAC *Identification of the Abundant Clusters*

Description

This function is used to identify the Abundant Clusters (AC), that is to say clusters having cell abundance (absolute or relative) statistically greater than a specific threshold. Abundant Clusters are identified using a one-sample test (parametrized or non-parametrized). P-values can be corrected for multiple comparisons.

Usage

```
identifyAC(Results, samples, use.percentages = TRUE, method = "t.test",
  method.adjust = NULL, mu = 0, th.pvalue = 0.05)
```

Arguments

Results	a 'Results' object
samples	a character vector providing the sample names to used
use.percentages	a logical specifying if the computations should be performed on percentage
method	a character specifying the statistical method used to identify the Abundant Clusters. The parameter can take the values "t.test" or "wilcox.test"
method.adjust	a character specifying if the p-values should be corrected using multiple correction methods among: "holm", "hochberg", "hommel", "bonferroni", "BH", "BY" and "fdr" (from 'stats::p.adjust' method)
mu	a numeric specifying the theoretical value (mu) of the one sample statistical test
th.pvalue	a numeric specifying the p-value threshold

Value

a S4 object of class 'AC'

identifyCC *Identification of the correlation of SPADE cluster with a phenotype*

Description

This function is used to identify Correlated Clusters, that is to say clusters having an abundance (absolute or relative) statistically correlated with a biological variable. Correlated Clusters are identified using Pearson or Spearman coefficients of correlation. P-values can be corrected for multiple comparisons.

Usage

```
identifyCC(Results, variable, use.percentages = TRUE, method = "pearson",
  method.adjust = NULL, th.correlation = 0.75, th.pvalue = 0.05)
```

Arguments

Results	a 'Results' object
variable	a numerical named vector providing the correspondence between a sample name (in rownames) and the specific numerical phenotype
use.percentages	a logical specifying if the computations should be performed on percentage
method	a character indicating the correlation method to use: "pearson", "spearman"
method.adjust	a character specifying if the p-values should be corrected using multiple correction methods among: "holm", "hochberg", "hommel", "bonferroni", "BH", "BY" and "fdr" (from 'stats::p.adjust' method)
th.correlation	a numeric specifying the absolute value of the correlation coefficient threshold
th.pvalue	a numeric specifying the p-value threshold

Value

a S4 object of class 'CC'

---

identifyDAC *Identification of the Differentially Abundant Clusters*

Description

This function is used to identify differentially abundant clusters, that is to say clusters having abundance (absolute or relative) statistically different between two biological conditions. Differentially Abundant Clusters are identified using a two-sample test (parametrized or non-parametrized). P-values can be corrected for multiple comparisons.

Usage

```
identifyDAC(Results, condition1, condition2, use.percentages = TRUE,
  method = "t.test", method.adjust = NULL, method.paired = FALSE,
  th.pvalue = 0.05, th.fc = 2)
```

Arguments

Results	a 'Results' object
condition1	a character vector providing the sample names defined as the first condition
condition2	a character vector providing the sample names defined as the second condition
use.percentages	a logical specifying if the computations should be performed on percentage
method	a character specifying the name of the statistical test to use "t.test" or "wilcox.test"
method.adjust	a character specifying if the p-values should be corrected using multiple correction methods among: "holm", "hochberg", "hommel", "bonferroni", "BH", "BY" and "fdr" (from 'stats::p.adjust' method)
method.paired	a logical indicating if the statistical test must be performed in a paired manner
th.pvalue	a numeric specifying the p-value threshold
th.fc	a numeric specifying the fold-change threshold

Value

a S4 object of class 'DAC'

`importResultsFromCLR.ACS` *Importation of clustering results from a folder containing ACS files*

Description

The 'importResultsFromCLR.ACS()' function imports cell clustering results from CLR files contained in a specified folder. CLR files must be in ACS format. This function imports the cluster phenotype and abundance.

Usage

```
importResultsFromCLR.ACS(path, prob.th = 0.8, th.min_cells = 0,
  probs = c(0.05, 0.95), use.raw.medians = TRUE,
  quantile.approximation = FALSE, exclude.markers = NULL,
  assignments = NULL)
```

Arguments

`path`	a character specify the path of the ACS CLR files
`prob.th`	a numeric specifying the probability threshold to use for considering considering the cluster of each each cells
`th.min_cells`	a numeric specifying the minimum number of cell in a cluster of a sample to take its phenotype in account
`probs`	a vector of probabilities with 2 values in [0,1] to compute marker range quantiles. First is the lower bound and second is the upper bound.
`use.raw.medians`	a logical specifying if arcsinh transformed or raw medians will be used in the cluster expression matrix (FALSE by default)
`quantile.approximation`	a logical specifying if marker range quantiles are computed using all cells (FALSE), or is the means of the quantile of each samples (TRUE)
`exclude.markers`	a character vector of markers to exclude (case insensitive)
`assignments`	a data.frame containing all samples names (in rownames) and columns providing contextual associations like "bc" (biological conditions), "tp" (biological conditions) and "ind" (individuals)

Details

This function returns a 'Results' object including 'flowset', 'fcs.files' slots but not 'graph' and 'graph.layout' slots. The computation of marker range quantiles can be approximated using 'quantile.approximation' parameter which is more efficient in term of loading time and memory usage. The name of each file will be used as the name of the sample. All FCS files corresponding to all the samples must be contained in a single folder.

For each cell, the selected cluster will be one having the greater probability higher than the 'prob.th' parameter.

Value

a S4 object of class 'Results'

`importResultsFromCLR.CSV` *Importation of clustering results from a folder containing CVS files*

Description

The 'importResultsFromCLR.CSV()' function imports cell clustering results from CLR files contained in a specified folder. CLR files must be in CSV format (). This function imports only the cluster abundance.

Usage

```
importResultsFromCLR.CSV(path, prob.th = 0.8, th.min_cells = 0,
  assignments = NULL)
```

Arguments

`path`	a character specify the path of the CVS CLR files
`prob.th`	a numeric specifying the probability threshold to use for considering considering the cluster of each each cells
`th.min_cells`	a numeric specifying the minimum number of cell in a cluster of a sample to take its phenotype in account
`assignments`	a data.frame containing all samples names (in rownames) and columns providing contextual associations like "bc" (biological conditions), "tp" (biological conditions) and "ind" (individuals)

Details

This function returns a 'Results' object with only cluster abundance information (no phenotype). The name of each file will be used as the name of the sample.

For each cell, the selected cluster will be one having the greater probability higher than the 'prob.th' parameter.

Value

a S4 object of class 'Results'

`importResultsFromFCS` *Importation of clustering results from a folder containing FCS files*

Description

The 'importResultsFromFCS()' function imports cell clustering results from FCS files contained in a specified folder. This function imports the cluster phenotype matrix and count matrix. This function apply an hyperbolic sine transformation to imported FCS data (unless 'use.raw.medians' = TRUE) and compute the marker range quantiles.

Usage

```
importResultsFromFCS(path, exclude.markers = c("cell_length", "FileNum",
  "density", "time"), clustering.markers = NULL, probs = c(0.05, 0.95),
  use.raw.medians = FALSE, quantile.approximation = FALSE,
  th.min_cells = 0, assignments = NULL)
```

Arguments

`path`	a character specify the path of the FCS files
`exclude.markers`	a character vector of markers to exclude (case insensitive)
`clustering.markers`	a character vector specifying markers that have been used during the clustering procedure (if NULL, all markers will be considered as clustering markers)
`probs`	a vector of probabilities with 2 values in [0,1] to compute marker range quantiles. First is the lower bound and second is the upper bound.
`use.raw.medians`	a logical specifying if arcsinh transformed or raw medians will be used in the cluster expression matrix (FALSE by default)
`quantile.approximation`	a logical specifying if marker range quantiles are computed using all cells (FALSE), or is the means of the quantile of each samples (TRUE)
`th.min_cells`	a numeric specifying the minimum number of cell in a cluster of a sample to take its phenotype in account
`assignments`	a data.frame containing all samples names (in rownames) and columns providing contextual associations like "bc" (biological conditions), "tp" (biological conditions) and "ind" (individuals)

Details

This function returns a 'Results' object including 'flowset', 'fcs.files' slots but not 'graph' and 'graph.layout' slots. The computation of marker range quantiles can be approximated using 'quantile.approximation' parameter which is more efficient in term of loading time and memory usage. The name of each file will be used as the name of the sample. All FCS files corresponding to all the samples must be contained in a single folder. Importantly, FCS files must contain a channel called "cluster" allowing to identify which cluster is associated to each cell.

Value

a S4 object of class 'Results'

importResultsFromSPADE *Importation of clustering results generated by SPADE*

Description

The 'importResultsFromSPADE()' function imports SPADE cell clustering results from a specified path. This function imports the cluster phenotype matrix and count matrix as well as the SPADE tree. This function apply an hyperbolic sine transformation to imported FCS data (unless 'use.raw.medians' = TRUE) and compute the marker range quantiles.

Usage

```
importResultsFromSPADE(path, exclude.markers = c("cell_length", "FileNum",
  "density", "time"), probs = c(0.05, 0.95), use.raw.medians = FALSE,
  quantile.approximation = FALSE, th.min_cells = 0, load.phenotype = TRUE,
  assignments = NULL)
```

Arguments

path	a character specify the path of SPADE results folder
exclude.markers	a character vector of markers to exclude (case insensitive)
probs	a vector of probabilities with 2 values in [0,1] to compute marker range quantiles. First is the lower bound and second is the upper bound.
use.raw.medians	a logical specifying if arcsinh transformed or raw medians will be used in the cluster expression matrix (FALSE by default)
quantile.approximation	a logical specifying if marker range quantiles are computed using all cells (FALSE), or is the means of the quantile of each samples (TRUE)
th.min_cells	a numeric specifying the minimum number of cell in a cluster of a sample to take its phenotype in account
load.phenotype	a logical specifying if the phenotype matrix and fcs file will be loaded
assignments	a data.frame containing all samples names (in rownames) and columns providing contextual associations like "bc" (biological conditions), "tp" (biological conditions) and "ind" (individuals)

Details

This function returns a 'Results' object including 'flowset', 'fcs.files', 'graph' and 'graph.layout' slots. The computation of marker range quantiles can be approximated using 'quantile.approximation' parameter which is more efficient in term of loading time and memory usage.

Value

a S4 object of class 'Results'

`importResultsFromTables` *Importation of clustering results from text files*

Description

The 'importResultsFromTables()' function imports cell clustering results from two dataframes ('cluster.abundances' and 'cluster.phenotypes'). This function returns a 'Results' object.

Usage

```
importResultsFromTables(cluster.abundances, cluster.phenotypes,
  clustering.markers = NULL, th.min_cells = 0, assignments = NULL)
```

Arguments

`cluster.abundances`	a dataframe of cells abundances with clusters in row and samples in column
`cluster.phenotypes`	a dataframe containing median marker expression values for each cluster of each sample. In additions of markers, the two first columns are are dedicated to "cluster" and "sample" identifiers.
`clustering.markers`	a character vector specifying markers that have been used during the clustering procedure (if NULL, all markers will be considered as clustering markers)
`th.min_cells`	a numeric specifying the minimum number of cell in a cluster of a sample to take its phenotype in account
`assignments`	a data.frame containing all samples names (in rownames) and columns providing contextual associations like "bc" (biological conditions), "tp" (biological conditions) and "ind" (individuals)

Details

This function returns a 'Results' object without 'flowset', 'fcs.files', 'graph' and 'graph.layout' slots.

The 'cluster.abundances' dataframe must be formatted with the cluster names in rownames as following:

x	sample1	sample2	sample3	...
cluster1	749	5421	8424	...
cluster2	450	412	614	...
cluster3	288	782	478	...
...	...	...	...	...

The 'cluster.phenotypes' dataframe must be formatted as following:

sample	cluster	marker1	marker2	marker3	...
sample1	cluster1	0.212	0.445	1.756	...
sample1	cluster3	0.434	0.785	0.654	...
sample2	cluster1	0.574	2.641	3.854	...

sample2	cluster2	1.454	1.755	-0.568	...
sample2	cluster3	1.445	1.875	-0.786	...
sample3	cluster1	0.157	2.121	1.648	...
sample3	cluster2	1.415	1.963	0.786	...
sample3	cluster3	1.275	1.427	0.754	...
...	...	...	...	...	...

Value

a S4 object of class 'Results'

`kineticsViewer` *Visualization of cluster enrichment profiles kinetics*

Description

The Kinetics Viewer aims to visualize the cell cluster abundances in a kinetics manner. This representation displays the cell abundances over the time for each individual. The cell cluster abundances could be displayed as percentages or absolute numbers using the 'use.percentages' parameter (TRUE by default).

Usage

```
kineticsViewer(Results, samples = NULL, clusters = NULL,
  use.percentages = TRUE, show.on_device = TRUE, scale_y = NULL)
```

Arguments

`Results`	a 'Results' object
`samples`	a character vector providing the sample names to used (all samples by default)
`clusters`	a character or a character vector containing the cluster to be shown or if multiple clusters which will be merged
`use.percentages`	a logical specifying if the visualization should be performed on percentage
`show.on_device`	a logical specifying if the representation will be displayed on device
`scale_y`	a numeric value specifying the maximal value in y axsis

Details

Timepoints and indivuals information must be assigned to the samples in order to use this viewer (provided by the slot 'assignments' in the 'Results' object).

Value

a 'ggplot' object

`load.flowSet` *Loading of FCS files object into a 'Results' object*

Description

This function loads the FCS files to the 'flowset' slot of the 'Results' object.

Usage

```
load.flowSet(Results = NULL, fcs.files, exclude.markers, use.raw.medians,
  pattern = ".fcs.density.fcs.cluster.fcs")
```

Arguments

`Results`	a Results object (with 'fcs.files' slot not null) (optional)
`fcs.files`	a character vector containing the absolute path of the original FCS files
`exclude.markers`	a character vector of markers to exclude (case insensitive)
`use.raw.medians`	a logical specifying if the arcsinh transformation must be performed or not
`pattern`	a character specifying the pattern of the FCS file

Details

If a 'Results' object is provided, others parameters ('fcs.files', 'exclude.markers' and 'use.raw.medians') will be ignored (they will be retrieved from the 'Results' object).

Value

a S4 'flowSet' object

`MDSViewer` *Visualization of SPADE cluster or sample similarities using MDS*

Description

Multidimensional Scaling (MDS) methods aim to represent the similarities and differences among high-dimensional objects into a space of a lower number of dimensions, generally in two or three dimensions for visualization purposes. In MDS representations, the Kruskal Stress (KS) indicates the percentage of information lost during the dimensionality reduction process.

The MDS Viewer aims to visualize the similarities between samples or clusters based on their abundances. In such representation, each dot represents a sample or a cluster and the distances between the dots are proportional to the Euclidean distance between these objects.

The representation space can be specified using the 'space' parameter ("samples" or "clusters").

Usage

```
MDSViewer(Results, space = "clusters", samples = NULL, clusters = NULL,
  use.percentages = TRUE, dist.method = "euclidean",
  show.on_device = TRUE)
```

Arguments

Results	a 'Results' object
space	a character specifying the space ("clusters" or "samples", "cluster" by default)
samples	a character vector providing the sample names to used (all samples by default)
clusters	a character vector containing the clusters names to be visualized (by default all clusters will be displayed)
use.percentages	a logical specifying if the visualization should be performed on percentage
dist.method	a character string containing the name of the distance measure to use
show.on_device	a logical specifying if the representation will be displayed on device

Details

In the case of "samples" space, biological conditions information can be assigned to the samples (provided by the slot 'assignments' in the 'Results' object). This parameter must be a dataframe with sample names in row names and 2 other columns specifying the biological conditions and individuals.

Value

a 'ggplot' object

---

mergeClusters *Merging of cell clusters from a Results object*

Description

This function is used to merge one or more clusters in a Results object.

Usage

```
mergeClusters(Results, clusters, name)
```

Arguments

Results	a Results object
clusters	a character vector containing the names of the clusters to merge
name	a character specifiyng the name of the new cluster to create

Details

The function merges the abundances and the phenotypes of clusters into a new one. Clusters to merge are not removed from the Results object after the merging.

Value

a Results object

`phenoViewer` *Visualization of cluster phenotypes*

Description

The Pheno Viewer aims to visualize the phenotype of single cluster or a set of combined clusters. This representation displays median marker expressions of each sample using parallel coordinates. In such viewer, each line corresponds to a biological sample and lines are positioned on a space where the X-axis represents the cell markers and the Y-axis represents the marker expressions. If biological conditions have been assigned to the biological samples, then samples belonging to the same condition will be shown using the same color.

Marker expressions from several clusters can be combined by providing one or several cluster names to the ‘clusters‘ parameter. In this case, the displayed marker expressions are the means of the median expressions for each cluster of each marker.

Markers used as clustering markers are shown in blue. A dashed line indicates the mean marker expressions of all samples. Importantly, a grey ribbon indicates the bounds of marker expressions in the overall dataset. The visualization can be restricted to specific markers using the ‘markers‘ parameter.

Usage

```
phenoViewer(Results, samples = NULL, clusters = NULL, markers = NULL,
  show.mean = "both", show.on_device = TRUE, sort.markers = TRUE)
```

Arguments

`Results`	a Results object
`samples`	a character vector providing the sample names to used (all samples by default)
`clusters`	a character vector containing the clusters names to be visualized (by default all clusters will be displayed)
`markers`	a character vector specifying the markers to be displayed assignments
`show.mean`	a character specifying if marker means expression should be displayed, possible value are among: "none", "only" or "both"
`show.on_device`	a logical specifying if the representation will be displayed on device
`sort.markers`	a logical specifying if the markers must be sorted by names in the representation

Details

The ranges of value between marker bounds (using the ’bounds’ slot) will be displayed using a grey ribbon.

The ’show.mean’ parameter allows to visualize three kinds of information:

- "none" value will show marker median expressions for each selected samples;
- "only" value will show only the mean of median maker expressions for all selected samples (displayed as black dashed line);
- "both" value will show marker median expressions for each selected samples together with the mean of median maker expressions for all selected samples.

Value

a 'ggplot' object

`plot` *Visualization for all SPADEVizR objects*

Description

Generates a graphical representation for all SPADEVizR objects ('Results', 'AC', 'DAC', 'CC', and 'AP' objects).

Usage

```
plot(x, y = NULL, ...)

## S4 method for signature 'Results,missing'
plot(x, y = NULL, ...)

## S4 method for signature 'AC,missing'
plot(x, y = NULL, ...)

## S4 method for signature 'DAC,missing'
plot(x, y = NULL, ...)

## S4 method for signature 'CC,missing'
plot(x, y = NULL, ...)

## S4 method for signature 'AP,missing'
plot(x, y = NULL, ...)
```

Arguments

`x`	a 'Results', 'AC', 'DAC', 'CC' and 'AP' object
`y`	unused parameter
`...`	supplementary parameters transmitted to the 'heatmapViewer()', 'abundantClustersViewer()', 'volcanoViewer()', 'correlatedClustersViewer()' or 'circlesPackingViewer()' functions

Details

'Results' objects are represented using the 'heatmapViewer()' function. 'AC' objects are represented using the 'abundantClustersViewer()' function. 'DAC' objects are represented using the 'volcanoViewer()' function. 'CC' objects are represented using the 'correlatedClustersViewer()' function. 'AP' objects are represented using the 'circlesPackingViewer()' function.

Value

a 'ggplot' object

print *Textual previews for all SPADEVizR objects*

Description

Prints a preview for a SPADEVizR object.

Usage

```
## S4 method for signature 'Results'
print(x)

## S4 method for signature 'AC'
print(x)

## S4 method for signature 'DAC'
print(x)

## S4 method for signature 'CC'
print(x)

## S4 method for signature 'AP'
print(x)
```

Arguments

x	a SPADEVizR object

Value

none

qcSmallClusters *Computing of the fraction of clusters with low number of associated cells*

Description

Computes the fraction of clusters having a number of associated cells less than a specific threshold (th.size parameter).

Usage

```
qcSmallClusters(Results, clusters = NULL, th.size = 50,
  PDFfile = "qcSmallClusters.pdf", width = ncol(Results@cluster.abundances),
  height = nrow(Results@cluster.abundances)/4, tile.color = "black")
```

Arguments

Results	a Results object
clusters	a character vector containing the clusters names to be computed (by default all clusters will be computed)
th.size	a numeric value specifying the minimum number of cells needed for a cluster to be considered a a small cluster
PDFfile	a character specifying the location of the output PDF path
width	a numeric specifying the plot width in the output PDF file
height	a numeric specifying the plot height in the output PDF file
tile.color	a character specifying the border color of the tiles (NA to remove tile borders)

Details

This function can also generate a PDF file summarizing the results.

Value

a list containing a numeric (perc numeric element) containing the fraction of clusters having a number of associated cells less than the threshold and a dataframe (small.clusters element) specifying the clusters having a number of associated cells less than the threshold

---

qcUniformClusters *Computing of the fraction of cluster with uniform phenotypes*

Description

The 'qcUniformClusters()' function allows to report as PDF files 2 kinds of information. Firstly, the identification of clusters having non-unimodal marker expression densities is performs using a Hartigan's dip test. Secondly, markers of cluster having a large spread of expressions (above a threshold) are identified.

If all the clustering marker expressions of a cluster checks this assessments (depending of the 'uniform.test' parameter), the cluster is considered as uniform.

Usage

```
qcUniformClusters(Results, clusters = NULL, only.clustering_markers = FALSE,
  uniform.test = "both", th.pvalue = 0.05, th.IQR = 2,
  density.PDFfile = "qcUniformClusters_density.pdf",
  density.PDFfile.dim = c(17, 10),
  heatmap.PDFfile = "qcUniformClusters_heatmap.pdf",
  heatmap.PDFfile.dim = c(length(Results@marker.names),
  length(Results@cluster.names)/4), tile.color = "black", verbose = TRUE,
  ...)
```

Arguments

Results	a Results object (with 'flowset' slot not null)
clusters	a character vector containing the clusters names to be computed (by default all clusters will be computed)
only.clustering_markers	a logical specifying if only clustering marker densities must be displayed.
uniform.test	a character specifying the qc assessment to perform
th.pvalue	a numeric value specifying the pvalue threshold of the Hartigan's dip test (multimodal if p.value < th.pvalue)
th.IQR	a numeric value specifying the IQR (interquartile range) threshold to assume a distribution as uniform
density.PDFfile	a character specifying the output path of the marker expression density plots (or NULL to avoid this step)
density.PDFfile.dim	a numeric vector specifying the width and the height of the density PDF file
heatmap.PDFfile	a character specifying the output path of the marker expression accuracy heatmap (or NULL to avoid this step)
heatmap.PDFfile.dim	a numeric vector specifying the width and the height of the heatmap PDF file
tile.color	a character specifying the border color of the tiles (NA to remove tile borders)
verbose	a boolean specifying if some messages must be displayed during the generation of the qc report
...	supplemental parameters passed to the dip.test function

Details

The test of multimodal distribution is performs using a Hartigan's dip test. The null hypothesis of this test assume the distribution as unimodal Supplemental parameters passed to the dip.test function are ('simulate.p.value' and 'B'). 'simulate.p.value' is a logical indicating whether to compute p-values by Monte Carlo simulation (FALSE by default). 'B' is an integer specifying the number of replicates used in the Monte Carlo test (2000 by default).

The spread of a marker expression is considered as high if the IQR (interquartile range) is higher than a provided threshold th.IQR.

The 'uniform.test' parameter can take 3 values: "unimodality": check if the distribution of the marker expression is unimodal or not using a Hartigans dip test "spread": check if the spread of the marker expression is below an IQR threshold (interquantile range) "both": check "unimodality" and "spread"

Value

a list containing a numeric (perc numeric element) specifying the percentage of clusters having only uniform clustering marker expressions and a dataframe (accuracy.matrix element) specifying if each marker of each cluster is uniform or not.

removeClusters *Removing of cell clusters from a Results object*

Description

This function is used to remove one or more cell clusters from a Results object.

Usage

```
removeClusters(Results, clusters)
```

Arguments

Results	a Results object
clusters	a character vector containing the names of the clusters to remove

Value

a Results object

Results-class *Results class definition*

Description

The Results object is a S4 object containing cell clustering results.

This object mainly stores the cluster abundance matrix (i.e. the number of cells associated to each sample for each cluster) and the cluster phenotypes matrix (i.e. the median expressions for each marker of each cluster).

In addition, this object can contain information about clustering results, such a SPADE tree.

Details

The 'cluster.abundances' dataframe contains the number of cells associated to each sample for each cluster. This dataframe stores the clusters in rows and the samples in columns.

The 'cluster.phenotypes' dataframe stores the median expressions for each marker of each cluster. This dataframe stores in the first column the sample names, in the second column the cluster names, and in the others columns the maker median expressions.

The 'bounds' dataframe contains the marker expressions boundaries (minimum and maximum, or specific percentiles) for each marker.

The 'print()' and 'show()' can be used to display a summary of this object.

Slots

`cluster.abundances` a dataframe containing the number of cells associated to each sample for each cluster

`cluster.phenotypes` a dataframe containing the median expressions for each marker of each cluster

`sample.names` a character vector containing the sample names

`cluster.names` a character vector containing the cluster names

`cluster.number` a numeric specifying the number of clusters

`marker.names` a character vector containing the marker names

`clustering.markers` a character vector specifying the markers that have been used by the clustering algorithms

`bounds` a numeric data.frame containing the marker expressions boundaries for each marker

`use.raw.medians` a logical specifying if the marker expressions correspond to raw or transformed data

`flowset` a flowSet object containing the imported SPADE FCS files

`fcs.files` a character vector containing the location of the imported FCS files

`graph` a igraph object containing the SPADE tree structure

`graph.layout` a numeric matrix containing the SPADE tree layout

`assignments` a dataframe containing annotations for each sample samples such as a biological condition ("bc"), a timepoint condition ("tp") or an individual ("ind") assignment

`th.min_cells` a numeric specifying the minimal number of cells that a cluster for a given samples needs to have to be taken into consideration in its phenotypical characterization

---

show *Textual previews for SPADEVizR objects*

Description

Show a preview for a SPADEVizR object.

Usage

```
## S4 method for signature 'Results'
show(object)

## S4 method for signature 'AC'
show(object)

## S4 method for signature 'DAC'
show(object)

## S4 method for signature 'CC'
show(object)

## S4 method for signature 'AP'
show(object)
```

Arguments

object	a SPADEVizR object

Value

none

`streamgraphViewer` *Visualization of cluster abundance dynamics*

Description

The Streamgraph Viewer aims to visualize both absolute and relative abundance of clusters across the samples. This representation displays cell abundances using a stacked area graph which is placed around a central axis.

The cell clusters to visualize must be specified using the ‘clusters‘ parameter. Moreover, samples to be represented and their orders can be specified using the ‘samples‘ parameter.

Usage

```
streamgraphViewer(Results, samples = NULL, clusters = NULL,
  use.relative = FALSE, show.on_device = TRUE)
```

Arguments

Results	a ’Results’ object
samples	a character vector providing the sample names to used (all samples by default)
clusters	a character vector containing the clusters names to be visualized
use.relative	a logical specifying if the visualization should be performed on relative abundance
show.on_device	a logical specifying if the representation will be displayed on device

Details

The order of samples in the ’samples’ vector correspond to the order where the sample will be displayed.

Value

a ’ggplot’ object

treeViewer *Visualization of combined SPADE trees*

Description

The Tree Viewer aims to visualize the SPADE tree representations. This representation displays the SPADE cell clusters using this minimal spanning tree layout computed by SPADE. In such tree, each node represents a cell cluster and nodes are linked based on theirs phenotype similarities. This viewer improves the original SPADE tree representations by allowing to combine SPADE trees from multiple samples.

Significant clusters can be highlighted (node borders are then colored in blue) by providing a 'AC', 'DAC', or 'CC' object (using the 'highlight' parameter). As with the original SPADE tree representations, nodes can be colored based on the marker median expression of a specific marker (using the 'marker' parameter).

Usage

```
treeViewer(Results, samples = NULL, highlight = NULL, marker = NULL,
  show.on_device = TRUE)
```

Arguments

Results	a 'Results' object (with 'graph' and 'graph.layout' slots not null)
samples	a character vector providing the sample names to used (all samples by default)
highlight	an AC, DAC or CC object to highlight identified significant clusters in the SPADE tree
marker	a character specifying the marker to be displayed
show.on_device	a logical specifying if the representation will be displayed on device

Details

The size of tree nodes is proportional to the number of cells in each cluster. If the 'stat.object' parameter is provided node outlines are colored according to clusters significance.

Value

a 'ggplot' object

unload.flowSet *Unload 'flowSet' object from a 'Results' object*

Description

This function unloads the 'flowSet' object in a 'Results' object.

Usage

```
unload.flowSet(Results)

## S4 method for signature 'Results'
unload.flowSet(Results)
```

Arguments

Results	a 'Results' object

Value

The new 'Results' object

---

volcanoViewer *Visualization of differentially abundant clusters*

Description

Generates a volcano plot representation showing for each cluster: its mean abundance fold-change and associated p-value.

This representation displays the p-value (shown as -log10(p-value)) in the Y-axis and the fold-change of cell abundances, in the X-axis in a two-dimensional chart. Each dot in the representation corresponds to a cell cluster, and both p-value and fold-change thresholds are shown using red dashed lines. Differentially Abundant Clusters are highlighted in red and labeled. The size of dots is proportional to the total number of associated cells in the 2 conditions merged.

Usage

```
volcanoViewer(DAC = NULL, fc.log2 = TRUE, show.cluster_sizes = TRUE,
  show.all_labels = FALSE, show.on_device = TRUE,
  max.dots_size = max(DAC@cluster.size), y.max = NULL)
```

Arguments

DAC	an object of class 'DAC' (object returned by the 'computeDAC()' function)
fc.log2	a logical specifying if the fold-change or the log2(fold-change) must be used
show.cluster_sizes	a logical specifying if dot sizes are proportional to number of associated cells
show.all_labels	a logical specifying if all cluster labels must be shown. Only labels of significant clusters are displayed otherwise
show.on_device	a logical specifying if the ggplot representation must be displayed on the device
max.dots_size	a numeric specifying the number of associated cells in the largest dot
y.max	a numeric specifying the maximal value of the y axis

Details

By default, only significant differentially abundant clusters are labeled. Labels for all clusters can be displayed by setting the 'all.label' parameter to TRUE.

Value

a 'ggplot' object

Index