Package 'CytoCompare'

April 13, 2017

Type Package

Title Computational comparisons of cytometry profiles

Version 1.0-0

Author Ludovic PLATON, Guillaume GAUTREAU and Nicolas TCHITCHEK

Maintainer Nicolas TCHITCHEK <nicolas.tchitchek@gmail.com>, Guillaume GAUTREAU <guillaume.gautreau@free.fr> and Ludovic PLATON <platon.ludovic@gmail.com>

Description The comparison of the phenotypes of cytometry cell clusters is an important aspect in high-dimensional cytometry analysis. Such comparaisons are mainly done to identify similar cell cluster in different cclustering approches. CytoCompare allows the comparison of cell cluster based on the density of expression markers. For each comparison of two cytometry profiles, CytoCompare computes a p-value asserting the significance of the similarity. An aggregated distance measure is also computed for each comparison. Automatic gating result files from SPADE, viSNE/ACCENSE or Citrus algorithms can be imported into CytoCompare. Moreover, CytoCompare has many visualization representations that can be used to make comparison results and intermediary results easily understandable. Importantly, users can also define their own statistical methods for the comparisons of the different types of profiles.

License GPL-3 | file LICENSE

Depends R (>= 3.1),

Imports BiocInstaller,
diptest,
flowCore,
flowUtils,
ggplot2,
ggrepel,
grid,
igraph,
MASS,
methods,
stats,
RJSONIO,
tools,
utils,
XML

biocViews FlowCytometry, Classification, Visualization

VignetteBuilder knitr

Suggests knitr,rmarkdown

RoxygenNote 5.0.1

R topics documented:

`as.CELL` *Coercion to a CELL object*

Description

Coerces a numeric matrix into a CELL object.

This function transforms a numeric matrix into one or several cell profiles.

Usage

```
as.CELL(object, name = "cell")

## S4 method for signature 'matrix'
as.CELL(object, name = "cell")
```

Arguments

object	a numeric matrix
name	a character specifying the internal name of the CELL object to create

Details

The matrix must have its column names corresponding to the cell markers.

Value

a S4 object of class CELL

---

as.CLUSTER *Coercion to a CLUSTER object*

Description

Coerces a CELL object or a numeric matrix into a CLUSTER object.

This function transforms the cell profiles from a CELL object or from a numeric matrix into one or several cell cluster profiles by computing the means, the standard deviations, and the densities of each marker.

Usage

```
as.CLUSTER(object, name = object@name, cluster = NULL, bin.width = 0.05)

## S4 method for signature 'CELL'
as.CLUSTER(object, name = object@name, cluster = NULL,
  bin.width = 0.05)

## S4 method for signature 'matrix'
as.CLUSTER(object, name = "cell_cluster", cluster = NULL,
  bin.width = 0.05)
```

Arguments

object	a CELL object or a numeric matrix
name	a character specifying the internal name of the CLUSTER object to create
cluster	a character indicating a channel name that can be used to gather the cell profiles into several cluster profiles. If a channel named is specified then the created CLUSTER object will contain as many profiles as different values present in this channel. If this parameter is NULL then a CLUSTER object with only one profile will be created
bin.width	a numeric value indicating the width of the bins in the density estimation computations (default=0.05)

Details

The 'cluster' parameter is especially useful when importing FCS files containing the SPADE clustering results (where an additional channel is used to indicate the associations between cells and cell clusters) or FCS files from any other automatic gating algorithm.

In the context of a numeric matrix coercion, the matrix must have its column names corresponding to the cell markers.

Value

a S4 object of class CLUSTER

---

c *Combination of CytoCompare objects*

Description

Combines two or several CELL, CLUSTER, GATE or RES objects.

Usage

```
## S4 method for signature 'CELL'
c(x, ..., recursive = FALSE)

## S4 method for signature 'CLUSTER'
c(x, ..., recursive = FALSE)

## S4 method for signature 'RES'
c(x, ..., recursive = FALSE)
```

Arguments

x	a first CELL, CLUSTER, GATE or RES object
...	further objects of the same class as x to be combined
recursive	a logical value indicating if the function recursively descends through lists combining all their elements into a vector. Not implemented and should be set to FALSE

Details

All the different objects to combine must be of the same type.

This function is especially useful when combining comparison results from different RES objects into a single RES object (the RES objects must share the same markers and marker weights). RES objects can be combined to an empty RES object (i.e. RES()).

This function is also especially useful when combining cell profiles obtained from different FCS files into one single CELL object.

Value

a S4 object of class CELL, CLUSTER, GATE or RES

cdf.density *Cumulative distribution function of a DENSITY object*

Description

Provides the cumulative distribution function (CDF) of a DENSITY object at specific values.

Usage

```
cdf.density(density, values)
```

Arguments

density	a DENSITY object
values	a numeric value or a numeric vector specifying the densities to compute

Details

This function is used internally when comparing the marker expression densities of cluster profiles with the default comparison approach. This function can also be used by users willing to define their own statistical functions when comparing cell, cell cluster or gate profiles.

Value

a numeric value of the cumulative distribution values

cdf.uniform *Cumulative distribution function of a uniform distribution*

Description

Provides the cumulative distribution function (CDF) of a uniform distribution at a specific value.

Usage

```
cdf.uniform(bounds, value)
```

Arguments

bounds	a numeric vector indicating the support (lower and upper bounds) of the uniform distribution
value	a numeric value specifying the densities to compute

Details

This function is used internally when comparing the marker expression densities of gate profiles with the default comparison approach. This function can also be used by users willing to define their own statistical functions for comparing cell, cell cluster or gate profiles.

Value

a numeric value of the cumulative distribution value

CLUSTER-class *CLUSTER class definition*

Description

CLUSTER is a S4 object containing one or several cell cluster profiles.

Details

This object mainly stores for each cell cluster profile, the means, the standard deviations and the densities of each marker.

Slots

`name` a character indicating the internal name of the CLUSTER object

`profiles` a character vector containing the names of the cell cluster profiles

`profiles.nb` an integer value indicating the number of cell cluster profiles

`profiles.sizes` an integer vector indicating the number of cells associated to each cluster profile

`markers` a character vector containing the marker names

`markers.nb` an integer value indicating the number of markers

`markers.clustering` a logical vector specifying the makers used as clustering markers

`means` a numeric matrix containing the means of each maker for each cluster profile

`sd` a numeric matrix containing the standard deviations of each maker for each cluster profile

`densities` a matrix of DENSITY objects containing the densities of each marker for each cluster profile

`overview.function` a character specifying the name of a function to call when plotting the CLUSTER object overview (please refer to the documentation of the 'plot()' function)

`graph` an object that can be used to store a visual representation of the cell clusters (e.g. a SPADE tree)

`graph.layout` a numeric matrix that can be used to store the positions of cell clusters in a 2-dimensional space (e.g. a SPADE tree layout)

compare *Compare two cytometry profiles.*

Description

Cytometry profiles contained in CELL, CLUSTER, or GATE objects can be compared using the 'compare()' function. Comparison results are stored in a RES object.

Comparisons can be performed between profiles of same types or between profiles of different types. In the default statistical approach:
* if the comparisons are performed on profiles of same type then profiles will be compared to identify similar profiles
* if the comparisons are performed on profiles of different types then profiles will be compared to identify included profiles

In the context of a similarity comparison, a distance is computed between each marker of the two profiles (`$D_i$`). Marker distances below a distance threshold, specified by the user, will correspond to a marker similarity success. Euclidean distance will be used when comparing cell profiles while the Kolmogorov-Smirnov distance will be used when comparing cluster or gate profiles. A weight can be associated to each marker, in order to modulate their importance. An aggregation of marker distances is performed using an exact binomial test where marker successes are considered as successful Bernoulli experiments. Thereby, the proportion of marker successes is compared to a probability of success (`$P$`) specified by the user. A aggregated distance (`$D$`), corresponding to the weigthed mean of marker distances, is additionaly returned.

In the context of an inclusion comparison, an inclusion assessment is performed for each marker of the two profiles. A cell profile marker is considered as included in a gate profile when its expression value is within the range of the marker boundaries. Similarly, a cell profile marker is considered as included in a cell cluster profile when its expression value is within the range of the marker cluster defined based on quantiles of marker expression densities. Finally, a cell cluster profile marker is considered as included in a gate profile when its expression boundaries is within the range of the marker gate. As for similarity comparisons, weights associated to each marker. The aggregation of marker inclusion is also performed using an exact binomial test.

Comparisons can be performed based on the whole set of common markers between the two profiles, or based on a subset of markers specified by the user. Moreover, markers can be weighted in the comparison procedure, via a MWEIGHTS object.

If only one object is provided to the ‘compare()‘ function then the comparisons will be performed between all profiles of this object. If two objects are provided to the ‘compare()‘ function then the comparisons will be performed between all possible pairs of profiles between these two objects.

Importantly, users can define their own function to perform the statistical comparisons of the profiles, using the ‘method‘ parameter. Please refer to the user tutorial for more details about this feature.

Usage

```
compare(object1, object2, ...)

## S4 method for signature 'CLUSTER,missing'
compare(object1, mweights = NULL,
  method = "compare_default", method.params = NULL)

## S4 method for signature 'CLUSTER,CLUSTER'
compare(object1, object2, mweights = NULL,
  method = "compare_default", method.params = NULL)
```

Arguments

`object1`	a CELL, CLUSTER or GATE object
`object2`	a CELL, CLUSTER or GATE object
`...`	other parameters
`mweights`	a MWEIGHTS object specifying the markers to use in the comparison procedure with theirs associated weights
`method`	a function or a character specifying the name of a function to use when performing the statistical comparisons between the cytometry profiles
`method.params`	a named character list used to parametrize the comparison function (please see the details section)

Details

Different parameters can be defined, via the method.params named list, to specify the behaviour of the comparisons:
* the D.th parameter indicates the distance threshold
* the P parameter indicates the expected proportion of marker successes
* the nbcells.th parameter indicates the number of cells per cluster below which the marker expression density of a cell cluster profile will be approximated by a normal distribution
* the cluster.quantiles parameter indicates the quantiles that will define the marker expression ranges for the cell cluster profiles

In the case of comparisons between two cell profiles, the marker distances are calculated based on the Euclidean distance. The parameter 'D.th' is set to 1.50 by default and the parameter 'P' is set to 0.75 default.

In the case of comparisons between two cell cluster profiles, the marker distances are calculated based on the Kolmogorov-Smirnov distance. The parameter 'D.th' is set to 0.30 by default and the parameter 'P' is set to 0.75 default. The nbcells.th parameter indicates the number of cells per cluster below which the density will be approximated by a normal distribution (set to 50 by default)

In the case of comparisons between two gate profiles, gates are modeled by uniform distributions, and the marker distances are calculated based on the Kolmogorov-Smirnov distance. The parameter 'D.th' is set to 0.30 by default and the parameter 'P' is set to 0.75 default.

In the case of comparisons between a cell profile and a gate profile, a cell profile marker is considered as included in the gate profile when its expression value is within the range of the marker boundaries. The parameter 'P' is set to 0.75 default.

In the case of comparisons between a cell profile and cell cluster profile, a cell profile marker is considered as included in a cell cluster profile when its expression value is within the range of the marker cluster defined based on quantiles of marker expression densities. The parameter 'P' is set to 0.75 default. The 'cluster.quantiles' parameter indicates the quantiles that will define the marker expression ranges for the cell cluster profile (set to 0.10 and 0.90 by default).

In the case of comparisons between a cell cluster profile and gate profile, a cell cluster profile marker is considered as included in a gate profile when its expression boundaries is within the range of the marker gate. The parameter 'P' is set to 0.75 default. The 'cluster.quantiles' parameter indicates the quantiles that will define the marker expression ranges for the cell cluster profile (set to 0.10 and 0.90 by default).

Importantly, in the case of comparisons involving CLUSTER profiles, Hartigan's dip tests and InterQuartile Ranges (IQR) can be computed in order to estimate if the marker expression densities are unimodales with low spreads. The Hartigan's dip test p-value threshold and IQR threshold can be both parametrized using the 'dip.pvalue' and 'IQR.th' parameters. If a marker density do not respect these constraints, the distance is set to 1.

Value

a S4 object of class RES

`create.MWEIGHTS` *Creation of a MWEIGHTS object*

Description

Creates a MWEIGHTS object based on a set of marker names, where all marker weights are set to 1.

Usage

```
create.MWEIGHTS(markers)
```

Arguments

markers	a character vector specifying the names of the markers

Details

This function is a short-cut to the following code:
markers <- c("marker1","marker2","marker3","markeri...","markern")
weights <- rep(1,length(markers))
mweights <- MWEIGHTS(markers=markers,weights=weights)

The weight of each marker is set to 1 but can be changed afterwards using the function set.

Value

a MWEIGHTS object containing the marker weights

DENSITY-class *DENSITY class definition*

Description

DENSITY is a S4 object used to stores a marker expression density.

Details

This object mainly stores for each marker: the bin characteristics, the negative and positive marker densities values, and the number of cells used in the density estimation. Densities are stored using two numeric vectors: 'values.neg' for the negative densities and 'values.pos' for the positive densities. This strategy allows to compute and store densities without defining an absolute minimal value or an absolute maximal value.

Slots

`name` a character indicating the internal name of the CELL object

`bin.interval` a numeric vector of two values specifying the density boundaries

`bin.nb` a numeric vector of two values specifying the numbers of negative and positive bins

`values.pos` a numeric vector containing the positive density bins

`values.neg` a numeric vector containing the negative density bins

`point.nb` a numeric value indicating the number of point used to compute the expression density

`bin.width` a numeric value indicating the width of the bins used in the density estimation

dheatmap *Density heatmap of the marker expression*

Description

Generates a marker density heatmap for the cell cluster profiles stored in a 'CLUSTER' object.

Usage

```
dheatmap(cluster, density.max = NULL, return.gg = FALSE)
```

Arguments

cluster	a CLUSTER object containing one or several cell cluster profiles
density.max	a numeric specifying the maximal density gradient value in the heatmap
return.gg	a logical indicating if the function should return a list of ggplot objects

Details

In such representation, each bar corresponds to a marker and the color gradient is proportional to the marker expression density.

Value

if return.gg is TRUE, the function returns a list of ggplot objects

extract *Extraction of subsets of data from CytoCompare objects*

Description

Extracts subsets of CELL, CLUSTER, GATE, MWEIGHTS or RES object.

Usage

```
## S4 method for signature 'CLUSTER,ANY,ANY,ANY'
x[i, j]

## S4 method for signature 'MWEIGHTS,ANY,ANY,ANY'
x[i]

## S4 method for signature 'RES,ANY,ANY,ANY'
x[i]
```

Arguments

x	a CELL, CLUSTER, GATE, MWEIGHTS or RES object
i	a numeric, logical or character vector
j	a numeric, logical or character vector

Details

For cytometry objects (CELL, CLUSTER, or GATE objects), the parameter 'i' represents a vector of profiles to extract and the parameter 'j' represents a vector of markers to extract.

For MWEIGHTS objects, the parameter 'i' represents a vector of markers to extract.

For RES objects, the parameter 'i' represents a vector of comparisons to extract.

Value

a S4 object of class CELL, CLUSTER, GATE, MWEIGHTS or RES

---

import.CITRUS *Importation of cell cluster profiles from a Citrus result*

Description

Imports one or several cell cluster profiles identified by the Citrus algorithm into a CLUSTER object.

Usage

```
import.CITRUS(file, dictionary = NULL, exclude = NULL, bin.width = 0.05,
  minimumClusterSizePercent = 0.05, cluster.selection = NULL)
```

Arguments

file	a character indicating the location of the citrusClustering.Rdata file
dictionary	a two-column data.frame providing the correspondence between the original marker names (first column) and the new marker names (second column)
exclude	a vector containing the marker names to be excluded in the import procedure
bin.width	a numeric value indicating the width of the bins for the marker expression densities computations
minimumClusterSizePercent	a numeric value indicating the minimal ratio of cells per cluster to import
cluster.selection	a character vector containing the names of the clusters to import

Details

Citrus is an algorithm that clusters cells using a hierarchical clustering procedure (similarly to SPADE) and then identifies the cell clusters that are significantly associated with different biological condition phenotypes (PMID:24979804).

Value

a S4 object of class CLUSTER

import.CLUSTER *Importation of cell cluster profiles from a tab separated file*

Description

Imports one or several cell cluster profiles from a tab separated file into a CLUSTER object. In this case, the marker expressions of each cluster are assumed to be normally distributed.

Usage

```
import.CLUSTER(file, exclude = NULL)
```

Arguments

file	a character specifying the location of a tab separated file to import
exclude	a character vector containing the marker names to be excluded in the import procedure

Details

Tab separated file to import must contain for each cell cluster profile the means and the standard deviations of the expression markers and must be formatted as the following:
* each row must represent a cell cluster profile;
* each column must represent a marker;
* each cell in the table must contain the marker expression means and the standard deviations for a given cell cluster separated by a semicolon;
The first column must contain the cell cluster names and the first row must contain the marker names.

It is to note that 'CLUSTER' objects constructed via the 'import.CLUSTER()' function do not contain the densities of expression markers (please refer to the documentation of the 'compare()' function).

Value

a S4 object of class CLUSTER

import.SPADE *Importation of cell cluster profiles from SPADE results*

Description

Imports one or several cell cluster profiles identified by the SPADE algorithm into a CLUSTER object.

Usage

```
import.SPADE(path, exclude = NULL, trans = "arcsinh",
  trans.para = switch(trans, arcsinh = list(arcsinh.scale = 5), log =
  list(log.shift = "auto", log.base = 10), none = NULL), trans.exclude = NULL,
  bin.width = 0.05, extract.folder = NULL, extract.folder.del = FALSE,
  zip = FALSE, rescale = FALSE, rescale.quantiles = c(0, 1))
```

Arguments

path	a character indicating the location to a zip or a folder containing the SPADE results
exclude	a character vector containing the marker names to be excluded in the import procedure
trans	a character specifying the name of a transformation function to apply on the marker expression intensities. Possible functions are "arcsinh" for arc sin hyperbolic transformation (default), "log" for logarithmic transformation, or "none" for no transformation
trans.para	a named list containing parameters for the transformation. Please refer to the details section for more details
trans.exclude	a character vector containing the marker names for which no transformation must be applied on
bin.width	a numeric value indicating the width of the bins for the marker expression densities computations
extract.folder	a folder path for extracting the SPADE zip archive (temporary folder by default)
extract.folder.del	a logical value indicating if the extracted SPADE results should be removed after the extraction
zip	a logical value that specify if the path specifies a zip file
rescale	a logical specifying if marker expression intensities must be rescale between 0 and 1
rescale.quantiles	a numeric vector of two values specifying the quantiles of the marker expression intensities used to rescale

Details

SPADE is a popular visualization and analysis algorithm that identifies clusters of cells having similar expression profiles for selected markers using an agglomerative hierarchical clustering-based algorithm combined with a density-based down-sampling procedure (PMID:21964415). Given a set of FCS files (usually one file per sample), SPADE identifies cell clusters based on the whole dataset and provides then for each sample the amount of cells present within each cluster.

The 'rescale' parameter can be used to rescale the marker expression intensities from 0 to 1 (with respect to the distribution proportion). The rescaling can be performed based on minimal and maximal expression values or based on specified quantiles using the 'rescale.quantiles' parameter. This strategy is especialy usefull when comparing cell or cell cluster profiles obtained from different experimental/staining conditions.

Value

a S4 object of class CLUSTER

import.VISNE_ACCENSE *Importation of cell cluster profiles from viSNE/ACCENSE results*

Description

Imports one or several cell cluster profiles identified by the viSNE/ACCENSE algorithm into a CLUSTER object.

Usage

```
import.VISNE_ACCENSE(file, exclude = NULL, trans = "arcsinh",
  trans.para = switch(trans, arcsinh = list(arcsinh.scale = 5), log =
  list(log.shift = "auto", log.base = 10), none = NULL),
  trans.exclude = "population", bin.width = 0.05)
```

Arguments

file	a character indicating the location to a zip or a folder containing the SPADE results
exclude	a character vector containing the marker names to be excluded in the import procedure
trans	a character specifying the name of a transformation function to apply on the marker expression intensities. Possible functions are "arcsinh" for arc sin hyperbolic transformation (default), "log" for logarithmic transformation, or "none" for no transformation
trans.para	a named list containing parameters for the transformation. Please refer to the details section for more details
trans.exclude	a character vector containing the marker names for which no transformation must be applied on
bin.width	a numeric value indicating the width of the bins for the marker expression densities computations

Value

a S4 object of class CLUSTER

intersect *Identification of common markers between two cytometry objects*

Description

Identifies the common markers between two cytometry objects (CELL, CLUSTER or GATE objects) and store the results in a MWEIGHTS object.

Usage

```
## S4 method for signature 'CLUSTER,CLUSTER'
intersect(x, y)
```

Arguments

x	a CELL, CLUSTER or GATE object
y	a CELL, CLUSTER or GATE object

Details

The weight of each marker is set to 1 but can be changed afterwards using the function set.

Value

a S4 object of class MWEIGHTS

---

load.examples *Retrieving of an example dataset of CytoCompare objects*

Description

Downloads and loads an example dataset of CytoCompare objects constructed based on cytometry profiles obtained from healthy human bone marrow unstimulated or stimulated (PMID:21964415).

This example dataset consists on three cytometry profiles of healthy human bone marrow, unstimulated or stimulated by BCR-inductor or IL-7, measured using a mass cytometry panel of more than 30 cell markers. This panel has been designed to identify a large spectrum of immune cell types like monocytes, B, or CD4+ and CD8+ T cells. A SPADE analysis has been performed to identify cell clusters, that have been then manually labelled based on theirs profiles. SPADE cell clusters corresponding to 6 majors cell types have been extracted and a set of rectangle gates have been constructed based these cell types.

Once downloaded, the following objects will be available:
* 'bm_example.cells.b', a 'CELL' object containing the cell profiles of the B cell populations;
* 'bm_example.cells.mono', a 'CELL' object containing the cell profiles of the monocyte cell populations;
* 'bm_example.cells.tCD4naive', a 'CELL' object containing the cell profiles of the naive CD4+ T cell populations;
* 'bm_example.cells.tCD8naive', a 'CELL' object containing the cell profiles of the naive CD8+ T cell populations;
* 'bm_example.cells.tCD4mem', a 'CELL' object containing the cell profiles of the memory CD4+ T cell populations;
* 'bm_example.cells.tCD8mem', a 'CELL' object containing the cell profiles of the memory CD8+ T cell populations;
* 'bm_example.clusters', a 'CLUSTER' object containing the cell cluster profiles for all the different cell populations, identified by SPADE;
* 'bm_example.clusters.b', a 'CLUSTER' object containing the cell cluster profiles of the B cell populations, identified by SPADE;
* 'bm_example.clusters.mono', a 'CLUSTER' object containing the cell cluster profiles of the monocyte cell cluster profiles, identified by SPADE;
* 'bm_example.clusters.tCD4naive', a 'CLUSTER' object containing the cell cluster profiles of the naive CD4+ T cell populations, identified by SPADE;
* 'bm_example.clusters.tCD8naive', a 'CLUSTER' object containing the cell cluster profiles of the naive CD8+ T cell populations, identified by SPADE;
* 'bm_example.clusters.tCD4mem', a 'CLUSTER' object containing the cell cluster profiles of the memory CD4+ T cell populations, identified by SPADE;

* ‘bm_example.clusters.tCD8mem‘, a ‘CLUSTER‘ object containing the cell cluster profiles of the memory CD8+ T cell populations, identified by SPADE;
* ‘bm_example.gates‘, a ‘GATE‘ object containing the gate profiles constructed based on the six main cell populations identified by SPADE;
* ‘bm_example.gates.b‘, a ‘GATE‘ object containing the gate profiles constructed based on the B cell populations;
* ‘bm_example.gates.mono‘, a ‘GATE‘ object containing the gate profiles constructed based on the monocyte cell populations;
* ‘bm_example.gates.tCD4naive‘, a ‘GATE‘ object containing the gate profiles constructed based on the naive CD4T cell populations;
* ‘bm_example.gates.tCD8naive‘, a ‘GATE‘ object containing the gate profiles constructed based on the naive CD8T cell populations;
* ‘bm_example.gates.tCD4mem‘, a ‘GATE‘ object containing the gate profiles constructed based on the memory CD4T cell populations;
* ‘bm_example.gates.tCD8mem‘, a ‘GATE‘ object containing the gate profiles constructed based on the memory CD8T cell populations;
* ‘bm_example.mweights‘, a ‘MWEIGHTS‘ object containing cell markers that can be used in for comparison computations;
* ‘bm_example.visne‘, a list of three ‘CELL‘ objects containing the viSNE cell profiles for each biological sample.

Usage

```
load.examples(del.file = FALSE)
```

Arguments

`del.file`	a logical specifying if the download CytoCompareExample.rdata file must be erased after the loading

Details

This function downloads a CytoCompareExample.rdata file containing the different CytoCompare objects (from a public ftp server "ftp://ftp.cytocompare.org/public/rdata/").

Value

none

`MWEIGHTS-class` *MWEIGHTS class definition*

Description

MWEIGHTS is a S4 object containing the marker weights to use in the comparison computations.

Details

This object mainly stores for each marker: the markers names and marker weights.

Slots

`markers` a character vector containing the marker names

`weights` a numeric vector containing the marker weights

`plot` *Plot for all S4 CytoCompare objects*

Description

Makes visual representations for CELL, CLUSTER, GATE, MWEIGHTS or DENSITY objects.

Usage

```
plot(object1, object2, ...)

## S4 method for signature 'CLUSTER,missing'
plot(object1, object2, overview = FALSE,
  return.gg = FALSE)

## S4 method for signature 'CLUSTER,CLUSTER'
plot(object1, object2, return.gg = FALSE)

## S4 method for signature 'MWEIGHTS,missing'
plot(object1, object2, return.gg = FALSE)

## S4 method for signature 'DENSITY,missing'
plot(object1, object2, return.gg = FALSE)

## S4 method for signature 'RES,missing'
plot(object1, return.gg = FALSE, ...)
```

Arguments

`object1`	a CELL, CLUSTER, GATE, MWEIGHTS or DENSITY object to plot
`object2`	another object to plot over the first object (only for CELL, CLUSTER or GATE objects)
`...`	other parameters
`overview`	a logical indicating is an overview of the object must be plotted (only available for single CELL and CLUSTER)
`return.gg`	a logical indicating if the function should return a list of ggplot objects

Details

Profiles contained in CELL, CLUSTER, GATE objects can be represented alone (object2=NULL) or in combination with another CELL, CLUSTER, GATE objects (via object2).

Cell profiles are represented via parallel coordinates where the x-axis represents the different markers and where the y-axis represents the marker expressions.

Cell cluster profiles are represented via parallel coordinates where the x-axis represents the different markers, where the y-axis represents the marker expressions, and where error bars indicate the marker expression standard deviations.

Gate profiles are represented via ribbons where the x-axis represents the different markers and where the y-axis represents the marker intensity ranges

In the case of a single CELL object, the parameter 'overview' indicates if an overview of the CELL object must be represented (e.g. viSNE map). In the case of a single CLUSTER object, the parameter 'overview' indicates if an overview of the CLUSTER object must be represented (e.g. a SPADE tree). In both cases, the plot function will call the function indicated in the 'overview.function' slot of the CELL or CLUSTER objects.

Marker weights contained in MWEIGHTS objects can be represented via bar plots where each bar corresponds to a marker and where the bar heights are proportional to the marker weights.

Density profiles contained in DENSITY objects can be represented via histogram plots where each bar corresponds to a density bin and where a smooth line represents the average estimation of the marker expression density.

If several profiles are present in the CELL, CLUSTER, GATE objects, all profiles or combination of profiles will be plotted. If several comparison results are present in the RES objects, all comparison results will be plotted.

Comparison results contained in RES object can also be ploted using this function. In such representation, each bar corresponds to a marker with a height proportional to the marker distance or inclusion assessment, and where the bars are colored if they model a success.

Value

if return.gg is TRUE, the function returns a list of ggplot objects

---

`print` *Textual preview for all S4 CytoCompare objects*

Description

Prints a preview for a CELL, CLUSTER, GATE, RES, MWEIGHTS or DENSITY object.

Usage

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

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

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

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

Arguments

`x`	a CELL, CLUSTER, GATE, RES, MWEIGHTS or DENSITY object

Value

none

`quantiles.density` *Quantiles of a DENSITY object*

Description

Provides the quantiles of a DENSITY object, at specific values.

Usage

```
quantiles.density(density, values)
```

Arguments

`density`	a DENSITY object
`values`	a numeric value or a numeric vector specifying the quantiles to compute

Details

This function is used internally when comparing the expression densities of cell markers with the proposed comparison approach. This function can also be used by users willing to define their own statistical functions for comparing cell, cell cluster or gate profiles.

Value

a numeric vector of two values or a numeric matrix containing the bin ranges of the quantiles values

---

`quantiles.uniform` *Quantiles of a uniform distribution*

Description

Provides the quantiles of a uniform distribution

Usage

```
quantiles.uniform(bounds, value)
```

Arguments

`bounds`	a numeric vector indicating the support (lower and upper bounds) of the uniform distribution
`value`	a numeric value specifying the quantile to compute

Details

This function is used internally when comparing the marker expression ranges of gate profiles with the default comparison approach. This function can also be used by users willing to define their own statistical functions for comparing cell, cell cluster or gate profiles.

Value

a numeric value of the quantiles value

`RES-class` *RES class definition*

Description

RES is a S4 object containing one or several comparison results.

Details

This object mainly stores for each comparison result: the associated distances (with the associated distance threshold) or inclusion p-value, and the marker successes. In the case of similarity comparisons, the aggregated distance between the two profiles and the marker distances are also stored in this object.

Slots

`comparisons` a data.frame containing for each comparison: the profile names, the type of the comparison (similarity or inclusion), the aggregated distance (or NA in case of inclusion), the distance threshold used (`$D_th$`) and the associated p-value

`comparisons.nb` is an integer indicating the number of comparisons

`markers` a character vector containing the marker names used in the comparisons

`marker.weights` a numeric vector containing the weigths associated to each marker involded in the comparisons

`marker.distances` a data.frame containing the marker distances for each comparison (or NA in case of inclusion assessments)

`marker.successes` a data.frame containing the marker successes for each comparison

---

`res.graph` *Circular graph representation of a RES object*

Description

Creates a circular graph representation of the comparison results. In such graph representation, each cytometry profile is represented by a node and links between the nodes represent significant similarities or inclusions between the profiles. Nodes are positioned on a circular layout and organized based on their object names.

Usage

```
res.graph(res, filename = "res.html", svgsize = 1000, pvalue.th = 0.2)
```

Arguments

res	a RES object
filename	a character specifying a file location where to save the HTML file of the representation
svgsize	a numeric value specifying the size of the SVG representation in pixels
pvalue.th	a numeric value specifying a p-value cutoff. Only the associations below this specific value will be returned

Details

The representation is provided as an interactive HTML file via a Scalable Vector Graphics (SVG) element created with the D3.js library. Users can interact with the representation by modifying the link tensions and the p-value cutoff for the significant similarities or inclusions.

Value

none

---

res.mds *Create a Multidimensional scaling representation of a RES object*

Description

Creates a Multidimensional scaling (MDS) representation of the comparison results. In such MDS representation each cytometry profile is represented by a dot in a two-dimensional space and the distances between the nodes are proportional to the distance measures between the profiles. The Kruskal Stress displayed at the left bottom of the representation quantifies the quality of the representation as the percentage of information lost in the dimensionality reduction process.

Usage

```
res.mds(res, filename = "res.html", cols = NULL, sizes = NULL,
  svgsize = 1000)
```

Arguments

res	a RES object
filename	a file location where to save the objects
cols	a character vector specifying the colours of the node in the SVG representation
sizes	a numeric vector specifying the sizes of the nodes in pixels in the SVG representation
svgsize	a numeric value specifying the size of the SVG representation in pixels

Details

The representation is provided as a HTML file via a Scalable Vector Graphics (SVG) element created with the D3.js library.

Value

none

---

set *Change marker weights in a MWEIGHTS object*

Description

Sets the weights of the markers in a MWEIGHTS object.

Usage

```
## S4 replacement method for signature 'MWEIGHTS,numeric,missing,numeric'
x[i] <- value

## S4 replacement method for signature 'MWEIGHTS,character,missing,numeric'
x[i] <- value

## S4 replacement method for signature 'MWEIGHTS,logical,missing,numeric'
x[i] <- value
```

Arguments

x	a MWEIGHTS object
i	a numeric, logical or character vector
value	a numeric vector containing the new marker weight values

Details

Marker weights can be set based on their indexes or names in the MWEIGHTS object.

Value

a S4 object of class MWEIGHTS

show *Textual preview for all S4 CytoCompare objects*

Description

Shows a preview for a CELL, CLUSTER, GATE, DENSITY, MWEIGHTS or RES object.

Usage

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

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

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

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

Arguments

object	a CELL, CLUSTER, GATE, DENSITY, MWEIGHTS or RES object

Value

none

Index