Package: rScudo (via r-universe)

November 1, 2026

Type Package

Title Signature-based Clustering for Diagnostic Purposes

Version 1.29.0

Description SCUDO (Signature-based Clustering for Diagnostic Purposes) is a rank-based method for the analysis of gene expression profiles for diagnostic and classification purposes. It is based on the identification of sample-specific gene signatures composed of the most up- and down-regulated genes for that sample. Starting from gene expression data, functions in this package identify sample-specific gene signatures and use them to build a graph of samples. In this graph samples are joined by edges if they have a similar expression profile, according to a pre-computed similarity matrix. The similarity between the expression profiles of two samples is computed using a method similar to GSEA. The graph of samples can then be used to perform community clustering or to perform supervised classification of samples in a testing set.

License GPL-3

Encoding UTF-8

LazyData true

URL https://github.com/Matteo-Ciciani/scudo

BugReports https://github.com/Matteo-Ciciani/scudo/issues

Collate 'class.R' 'accessors.R' 'packageDoc.R' 'utilities.R' 'scudoClassifyUtilities.R' 'scudoClassify.R' 'scudoTrain.R' 'scudoCytoscape.R' 'scudoModel.R' 'scudoNetwork.R' 'scudoPlot.R' 'scudoTest.R' 'show.R'

RoxygenNote 6.1.1

Depends R (>= 3.6)

Imports methods, stats, igraph, stringr, grDevices, Biobase, S4Vectors, SummarizedExperiment, BiocGenerics

Suggests testthat, BiocStyle, knitr, rmarkdown, ALL, RCy3, caret, e1071, parallel, doParallel

VignetteBuilder knitr

biocViews GeneExpression, DifferentialExpression, BiomedicalInformatics, Classification, Clustering, GraphAndNetwork, Network, Proteomics, Transcriptomics, SystemsBiology, FeatureExtraction

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

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

Date/Publication 2026-04-28 12:49:16 UTC

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

RemoteRef HEAD

RemoteSha d9887b333d832625acb5cba3e00ea338b9a822c8

Contents

rScudo-package 2
scudoClassify 3
scudoCytoscape 5
scudoModel 6
scudoNetwork 8
scudoPlot 10
ScudoResults-class 11
scudoTest 12
scudoTrain 14

Index **18**

`rScudo-package` *Signature-based Clustering for Diagnostic Purposes*

Description

SCUDO (Signature-based Clustering for Diagnostic Purposes) is a rank-based method for the analysis of gene expression profiles for diagnostic and classification purposes. It is based on the identification of sample-specific gene signatures composed of the most up- and down-regulated genes for that sample. Starting from gene expression data, functions in this package identify sample-specific gene signatures and use them to build a graph of samples. In this graph samples are joined by edges if they have a similar expression profile, according to a pre-computed similarity matrix. The similarity between the expression profiles of two sample is computed using a method similar to GSEA. The graph of samples can the be used to perform community clustering or to perform supervised classification of samples in a testing set.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

scudoTrain, scudoNetwork, scudoClassify

Examples

```
# To learn more about rScudo, start with the vignette:
browseVignettes("rScudo")
```

---

scudoClassify *Performes classification using SCUDO*

Description

Performs supervised classification of samples in a testing set using a network of samples generated by SCUDO during a training step.

Usage

```
scudoClassify(trainExpData, testExpData, N, nTop, nBottom,
    trainGroups, maxDist = 1, weighted = TRUE, complete = FALSE, beta = 1,
    alpha = 0.1, foldChange = TRUE, featureSel = TRUE, logTransformed = NULL,
    parametric = FALSE, pAdj = "none", distFun = NULL)
```

Arguments

trainExpData	either an ExpressionSet, a SummarizedExperiment, a data.frame or a matrix of gene expression data, with a column for each sample and a row for each feature
testExpData	either an ExpressionSet, a SummarizedExperiment, a data.frame or a matrix of gene expression data, with a column for each sample and a row for each feature
N	a number between 0 and 1, representing the fraction of the signature-to-signature distances that will be used to draw the graph
nTop	number of up-regulated features to include in the signatures
nBottom	number of down-regulated features to include in the signatures
trainGroups	factor containing group labels for each sample in trainExpData
maxDist	an integer. Only nodes with a distance from a testing node less or equal to maxDist are used to perform the classification
weighted	logical, whether to consider the distances associated to the edges to compute the scores for the classification. For a description of the classification method, see Details below
complete	logical, whether to consider all the nodes in the training set to perform the classification. If TRUE, the arguments N, maxDist, weighted and beta are ignored. For a description of the classification method, see Details below

beta	a coefficient used to down-weight the influence of distant nodes on the classification outcome. For a description of the classification method, see Details below
alpha	p-value cutoff for the optional feature selection step. If feature selection is skipped, alpha is ignored
foldChange	logical, whether or not to compute fold-changes from expression data
featureSel	logical, whether or not to perform a feature selection. Feature selection is performed using one of four tests: Student's t-test, ANOVA, Wilcoxon-Mann-Withney test, or Kruskal-Wallis test. The test used depends on the number of groups and the parametric argument
logTransformed	logical or NULL. It indicates whether the data is log-transformed. If NULL, an attempt is made to guess if the data is log-transformed
parametric	logical, whether to use a parametric or a non-parametric test for the feature selection
pAdj	pAdj method to use to adjust the p-values in the feature selection step. See p.adjust.methods for a list of adjustment methods
distFun	the function used to compute the distance between two samples. See Details of scudoTrain for the specification of this function

Details

This function performs supervised classification of samples in a testing set, using the networks similar to the one generated by scudoTrain and scudoNetwork as a model.

For each sample S in the testing set, a new distance matrix is computed using the expression profiles in the training set and the expression profile of S. The distance matrix is computed as described in the Details of scudoTrain.

If the argument complete is TRUE, the distance matrix is converted in a similarity score matrix. Then, the similarity scores between S and all the samples in the training set are aggregated according to groups. The mean similarity scores are computed for each group and classification scores are generated dividing them by their sum, obtaining values bewteen 0 and 1.

If the argument complete is FALSE, the distance matrix obtained form S and the training set is used to generate a network of samples, using the parameter N as a threshold for edge selection (see Details of scudoNetwork for a more complete description). Then the neighbors of S in the network are explored, up to a distance controlled by the parameter maxDist. If the weighted parameter is FALSE, the classification scores for each group are computed as the number of edges connecting S or one of its neighbors to a node of that group. The scores are than rescaled dividing them by their sum, in order to obtain values between 0 and 1. If the weighted parameter is TRUE, the classification scores for each group are computed as the sum of the similarity scores associated to edges connecting S or one of its neighbors to nodes of that group. The scores are than rescaled dividing them by their sum, in order to obtain values between 0 and 1. The parameter beta can be used to down-weight the contribution to the classification scores of edges connecting nodes distant form S, both in the weighed and unweighted cases.

The predicted group for each sample is the one with the largest classification score. Both predictions and classification scores are returned. Note that if the argument complete is FALSE, the classification socres for a sample may be all zero, which happens when the correspoonding node is isolated

in the network of samples. In this case the predicted group is NA. The tuning of the parameters can be performed automatically using the train function form the package caret and the function scudoModel.

Value

A list containing a factor with the predicticted class for each sample in testExpData and a data.frame of the classification scores used to generate the predictions.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

scudoTrain, scudoModel

Examples

```
expData <- data.frame(a = 1:10, b = 2:11, c = 10:1, d = 11:2,
    e = c(1:4, 10:5), f = c(7:10, 6:1), g = c(8:4, 1:3, 10, 9),
    h = c(6:10, 5:1), i = c(5:1, 6:10))
rownames(expData) <- letters[1:10]
groups <- factor(c(1,1,1,2,2,2,1,1,1))
inTrain <- 1:5

# perform classification
res <- scudoClassify(expData[, inTrain], expData[, -inTrain], 0.9, 3, 3,
    groups[inTrain], featureSel = FALSE)

#explore predictions
predictions <- res$predicted
scores <- res$scores
```

---

scudoCytoscape *Create a Cytoscape network from the output of scudoNetwork*

Description

A wrapper to RCy3 function calls to plot the result of scudoNetwork in Cytoscape 3. Cytoscape must be open before running this function.

Usage

```
scudoCytoscape(graph, title = "Scudo Graph", collection = "SCUDO")
```

Arguments

graph	object of class igraph, like the result of scudoNetwork
title	the title of the network
collection	the name of the Cytoscape collection

Value

The network SUID (an integer).

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

scudoNetwork, RCy3

Examples

```
# generate dummy dataset
exprData <- data.frame(a = 11:20, b = 16:25,
    c = rev(1:10), d = c(1:2, rev(3:10)))
rownames(exprData) <- letters[11:20]
grps <- as.factor(c("G1", "G1", "G2", "G2"))
nTop <- 2
nBottom <- 3

# run scudoTrain and scudoNetwork
res <- scudoTrain(exprData, grps, nTop, nBottom, foldChange = FALSE,
    featureSel = FALSE)
col <- c("#FF00FF", "#FF00FF", "#00FF00", "#00FF00")
net <- scudoNetwork(res, N = 0.5, colors = col)

# run scudoCytoscape (with Cytoscape open)
## Not run: scudoCytoscape(res, title = "scudoCytoscape output")
```

---

scudoModel *Generate model for* caret::train

Description

This function generates a suitable input for the method argument of the function train from the package caret, that can be used to perform automatic parameter tuning (e.g. using cross-validation).

Usage

```
scudoModel(nTop, nBottom, N, maxDist = 1, weighted = TRUE,
    complete = FALSE, beta = 1, distFun = NULL)
```

Arguments

nTop	number of up-regulated features to include in the signatures
nBottom	number of down-regulated features to include in the signatures
N	a number between 0 and 1, representing the fraction of the signature-to-signature distances that will be used to draw the graph
maxDist	an integer. Only nodes with a distance from a testing node less or equal to maxDist are used to perform the classification
weighted	logical, whether to consider the distances associated to the edges to compute the scores for the classification
complete	logical, whether to consider all the nodes in the training set to perform the classification. If TRUE, the arguments N, maxDist, weighted and beta are ignored
beta	a coefficient used to down-weight the influence of distant nodes on the classification outcome
distFun	the function used to compute the distance between two samples. See Details for the specification of this function

Details

This function can be used in conjunction with the functions in the package caret to tune the parameters of scudoClassify. The input of this function are vector of parameter values that the tuning procedure should explore. All possible combination of parameter values are explored by default. The user can change this using the search argument of the trainControl function.

The output of this function is a list that represens a classification model using scudoClassify and that can be used as input for the method argument of the function train.

Value

A named list

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

scudoClassify, train, trainControl

Examples

```
# Generate example dataset
expData <- data.frame(a = 1:10, b = 2:11, c = 10:1, d = 11:2,
    e = c(1:4, 10:5), f = c(7:10, 6:1), g = c(8:4, 1:3, 10, 9),
    h = c(6:10, 5:1), i = c(5:1, 6:10))
rownames(expData) <- letters[1:10]
groups <- factor(c(1,1,1,2,2,2,1,1,1))

# Run bootstrap. Notice that the dataset is transposed
ctrl <- caret::trainControl(method = "boot", number = 5)
model <- scudoModel(nTop = 3:5, nBottom = 3:5, N = 0.5, complete = TRUE)
set.seed(1)
bootRes <- caret::train(x = t(expData), y = groups, method = model,
    trControl = ctrl)
```

---

scudoNetwork *Create graph from a ScudoResults object*

Description

A function to create an `igraph` object from a `ScudoResults` object. In the graph, nodes are samples and edges quantify the similarity between the nodes.

Usage

```
scudoNetwork(object, N, colors = character())
```

Arguments

`object`	a `ScudoResults` object
`N`	a number between 0 and 1, representing the fraction of the signature-to-signature distances that will be used to draw the graph
`colors`	a character vector of hexadecimal RGB color codes used to color the nodes of the graph. `length(colors)` must be equal to the number of samples. By default colors are chosen according to the groups in `object`

Details

This function uses the distance matrix in the `ScudoResults` object to generate an `igraph` object, representing a graph where nodes are samples and edges quantify the similarity between the signatures of pairs of nodes.

The distance matrix in `object` is used to generate an unweighted adjacency matrix, that is then used to generate the graph. The sample quantile of probability N, computed from all the non-zero distances in the distance matrix, is used as a threshold to generate the adjacency matrix: all the distances larger than this quantile are mapped to 0, all the distances smaller than this quantile are mapped to 1 (with the exception of the distances of any node from itself, which are equal to 0).

Distances are set as attributes of the edges. Use `igraph::E(igraphObject)$distance` to retrieve them, where `igraphObject` is the result of `scudoNetwork`.

The `color` parameter controls the color of the nodes. It must be a vector of hexadecimal RGB color codes (like `"#FFFFFF"`), with length equal to the number of samples in `object`. By default, a different color is assigned to each group. If no group is specified in `object`, all nodes are set to the same color. A vector of node colors can be accessed with `igraph::V(igraphObject)$color`. Use `igraph::V(igraphObject)$group` to access the group label of each node (it returns NULL if no group is specified in `object`).

Value

An object of class `igraph`.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

`scudoCytoscape`, `ScudoResults`, `igraph`

Examples

```
# generate dummy dataset and run scudo
exprData <- data.frame(a = 11:20, b = 16:25,
    c = rev(1:10), d = c(1:2, rev(3:10)))
rownames(exprData) <- letters[11:20]
grps <- as.factor(c("G1", "G1", "G2", "G2"))
nTop <- 2
nBottom <- 3

res <- scudoTrain(exprData, grps, nTop, nBottom, foldChange = FALSE,
    featureSel = FALSE)

# generate network
col <- c("#FF0000", "#FF0000", "#0000FF", "#0000FF")
net <- scudoNetwork(res, N = 0.5, colors = col)

# retrieve node colors and groups
nodes <- igraph::V(net)
colors <- nodes$color
groups <- nodes$group

# retrieve distances from edges
edges <- igraph::E(net)
dist <- edges$distance

# plot the network
scudoPlot(net)
```

scudoPlot *Plot scudoNetwork result*

Description

A wrapper to plot.igraph and legend. Can be used to plot the result of scudoNetwork with a color legend.

Usage

```
scudoPlot(net, x = "bottomright", y = NULL, ...)
```

Arguments

net	an igraph object returned by scudoNetwork
x, y	the x and y coordinates to be used to position the legend. They can be specified by keyword or in any way which is accepted by xy.coords. See Details of legend
...	arguments to be passed to plot.igraph

Value

Returns NULL, invisibly.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

scudoNetwork, plot.igraph

Examples

```
# generate dummy dataset, run scudoTrain and scudoNetwork
exprData <- data.frame(a = 11:20, b = 16:25,
    c = rev(1:10), d = c(1:2, rev(3:10)))
rownames(exprData) <- letters[11:20]
grps <- as.factor(c("G1", "G1", "G2", "G2"))
nTop <- 2
nBottom <- 3
res <- scudoTrain(exprData, grps, nTop, nBottom, foldChange = FALSE,
    featureSel = FALSE)
net <- scudoNetwork(res, N = 0.5)

# Plot with scudoPlot
scudoPlot(net)
```

ScudoResults-class *Class ScudoResults*

Description

This is an S4 class that represents the output of the functions scudoTrain and scudoTest.

Details

This class provides a structure to represent the results of scudoTrain and scudoTest. It contains the distance matrix and the gene signatures generated by the SCUDO analysis. It is possible, although not recommended, to manually create instances of this class (see Examples below).

Slots

distMatrix a symmetric matrix with non-negative numeric elements

upSignatures a data.frame with the same colnames as distMatrix, representing the up-regualted features in each sample

downSignatures a data.frame with the same colnames as distMatrix, representing the down-regualted features in each sample

groupsAnnotation a factor that represents the groups used for the computeFC and the feature selection

consensusUpSignatures a data.frame that contains the consensus signatures of up-regulated features for each group

consensusDownSignatures a data.frame that contains the consensus signatures of dowm-regulated features for each group

selectedFeatures a character vector of selected features. If the feature selection was not performed, it contains every feature present in the input of the scudo functions

scudoParams a list of the parameters used to run the function that created the instance of the class

Methods

distMatrix signature(object = "ScudoResults"): a method for obtaining the distance matrix.

upSignatures signature(object = "ScudoResults"): a method for obtaining the signature of up-regualted features in each sample.

downSignatures signature(object = "ScudoResults"): a method for obtaining the signature of down-regulated features in each sample.

groupsAnnotation signature(object = "ScudoResults"): a method for obtaining the groups used for computeFC and feature selection.

consensusUpSignatures signature(object = "ScudoResults"): a method for obtaining the consensus signatures of up-regualted features in each group.

consensusDownSignatures signature(object = "ScudoResults"): a method for obtaining the consensus signatures of down-regulated features in each group.

`selectedFeatures signature(object = "ScudoResults")`: a method for obtaining the names of the features seleted. If no feature selection was performed, the names of every feature are returned.

`scudoParams signature(object = "ScudoResults")`: a method for obtaining the parameters that were used to generate the result.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

Examples

```
# manually generate instance of ScudoResults class
m <- matrix(1, ncol = 4, nrow = 4)
diag(m) <- 0
rownames(m) <- colnames(m) <- letters[1:4]
SigUp <- data.frame(a = letters[1:5], b = letters[6:10], c = letters[11:15],
    d = letters[16:20], stringsAsFactors = FALSE)
SigDown <- data.frame(a = letters[1:10], b = letters[11:20],
    c = letters[1:10], d = letters[11:20],
    stringsAsFactors = FALSE)
groups <- as.factor(c("G1", "G1", "G2", "G2"))
ConsUp <- data.frame(G1 = letters[11:15], G2 = letters[21:25],
    stringsAsFactors = FALSE)
ConsDown <- data.frame(G1 = letters[16:25], G2 = letters[1:10],
    stringsAsFactors = FALSE)
Feats <- letters[1:20]
Pars <- list()

scudoR <- ScudoResults(distMatrix = m,
    upSignatures = SigUp,
    downSignatures = SigDown,
    groupsAnnotation = groups,
    consensusUpSignatures = ConsUp,
    consensusDownSignatures = ConsDown,
    selectedFeatures = Feats,
    scudoParams = Pars)
```

`scudoTest` *Performs SCUDO analysis on test dataset*

Description

A function to perform the SCUDO analysis on test data, given an object of class `ScudoResults` used as training model.

Usage

```
scudoTest(trainScudoRes, testExpData, testGroups = NULL, nTop = NULL,
    nBottom = NULL, foldChange = TRUE, groupedFoldChange = FALSE,
    logTransformed = NULL, distFun = NULL)
```

Arguments

trainScudoRes	an object of class ScudoResult used as training model
testExpData	either an ExpressionSet, a SummarizedExperiment, a data.frame or a matrix of gene expression data, with a column for each sample and a row for each feature
testGroups	factor containing group labels for each sample in testExpData
nTop	number of up-regulated features to include in the signatures. If NULL, it defaults to the value present in trainScudoRes
nBottom	number of down-regulated features to include in the signatures. If NULL, it defaults to the value present in trainScudoRes
foldChange	logical, whether or not to compute fold-changes from expression data
groupedFoldChange	logical, whether or not to take into account the groups when computing fold-changes. See Details for a description of the computation of fold-changes
logTransformed	logical or NULL. It indicates whether the data is log-transformed. If NULL, an attempt is made to guess if the data is log-transformed
distFun	the function used to compute the distance between two samples. See Details of scudoTrain for the specification of this function

Details

Given an object of class ScudoResults and a set of expression profiles with unknown classification, scudoTest performs an analysis similar to scudoTrain, computing a list of signatures composed of genes over- and under-expressed in each sample, consensus signatures for each group and a distance matrix that quantifies the similarity between the signatures of pairs of samples.

scudoTest differs from scudoTrain in the feature selection step: only the features present in the ScudoResults object taken as input are considered for the follwing steps. The computation of fold-changes, the identification of gene signatures and the computation of the distance matrix are performed as described in the Details of scudoTrain.

If the classification of samples in the testing dataset is provided, it is only used for annotation purposes.

Value

Object of class ScudoResults.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

scudoTrain, scudoNetwork, ScudoResults, scudoClassify

Examples

```
# generate dummy train dataset
exprDataTrain <- data.frame(a = 11:20, b = 16:25,
    c = rev(1:10), d = c(1:2, rev(3:10)))
exprDataTest <- data.frame(e = 1:10, f = 11:20,
    g = rev(11:20), h = c(1:2, rev(3:10)))
rownames(exprDataTrain) <- rownames(exprDataTest) <- letters[11:20]
grpsTrain <- as.factor(c("G1", "G1", "G2", "G2"))
nTop <- 2
nBottom <- 3

# run scudo
res <- scudoTrain(exprDataTrain, grpsTrain, nTop, nBottom,
    foldChange = FALSE, featureSel = FALSE)
show(res)

# run scudoTest
testRes <- scudoTest(res, exprDataTest, foldChange = FALSE)
show(testRes)
```

---

scudoTrain *Performs SCUDO analysis*

Description

SCUDO (Signature-based ClUstering for DiagnOstic purposes) is a rank-based method for the analysis of gene expression profiles. This function computes gene signatures for each sample and consensus signatures for each group specified. A distance matrix is also computed, that can be used by the function scudoNetwork to generate a graph in which each node is a sample and an edge between two nodes quantitatively represents the similarity between their respective signatures.

Usage

```
scudoTrain(expressionData, groups, nTop, nBottom, alpha = 0.1,
    foldChange = TRUE, groupedFoldChange = FALSE, featureSel = TRUE,
    logTransformed = NULL, parametric = FALSE, pAdj = "none", distFun = NULL)
```

Arguments

expressionData	either an ExpressionSet, a SummarizedExperiment, a data.frame or a matrix of gene expression data, with a column for each sample and a row for each feature
groups	factor containing group labels for each sample in expressionData

nTop	number of up-regulated features to include in the signatures
nBottom	number of down-regulated features to include in the signatures
alpha	p-value cutoff for the optional feature selection step. If feature selection is skipped, alpha is ignored
foldChange	logical, whether or not to compute fold-changes from expression data
groupedFoldChange	logical, whether or not to take into account the groups when computing fold-changes. See Details for a description of the computation of fold-changes
featureSel	logical, whether or not to perform a feature selection. Feature selection is performed using one of four tests: Student's t-test, ANOVA, Wilcoxon-Mann-Withney test, or Kruskal-Wallis test. The test used depends on the number of groups and the parametric argument
logTransformed	logical or NULL. It indicates whether the data is log-transformed. If NULL, an attempt is made to guess if the data is log-transformed
parametric	logical, whether to use a parametric or a non-parametric test for the feature selection
pAdj	pAdj method to use to adjust the p-values in the feature selection step. See p.adjust.methods for a list of adjustment methods
distFun	the function used to compute the distance between two samples. See Details for the specification of this function

Details

Given a set of expression profiles with known classification, scudoTrain computes a list of signatures composed of genes over- and under-expressed in each sample. It also compute consensus signatures for each group and uses the signatures to compute a distance matrix that quantifies the similarity between the signatures of pairs of samples.

Before computing the signatures, two optional perprocessing steps are performed. In the first step fold-changes are compured from expression values. If the parameter groupedFoldChange is TRUE, the fold-changes are computed in two steps: first the mean expression value for each feature in each group is computed. Then, the fold-changes for each feature are computed dividing the expression values by the mean of the group means. If the the parameter groupedFoldChange is FALSE, the fold-changes are computed dividing the expression value of each feature by the mean expression value of that feature (regardless of groups). If the expression values are log-transformed, subtraction is used instead of division.

The second optional preprocessing step is a feature selection. This step is performed in order to select relevant features. Feature selection is performed using one of four tests: Student's t-test, ANOVA, Wilcoxon-Mann-Withney test, or Kruskal-Wallis test. The test used depends on the number of groups and the parameter parametric. The parameter pAdj controls the method used to adjust p-values for multiple hypothesis testing. For a list of adjustment methods see p.adjust. Features with an adjusted p-value less than alpha are selected.

After these two optional steps, the signatures for each sample are computed. Selected features are ranked according to the expression values (or the fold-change, if computed). Than the first nTop and the last nBottom features are selected from the ranked list of features. Two data.frames are containing the signatures of up-regulated genes and down-regulated genes for each sample are produced and are contained in the returned object.

Consensus top and bottom signatures are computed for each group. The avreage rank for each gene is computed for each group. Features are then ranked according to the average rank in each group and the first `nTop` and the last `nBottom` genes are selected to form the consensus signatures of each group. Two `data.frames` containing the consensus signatures are produced and are contained in the returned object.

Gene signatures are used to compute an all-to-all distance matrix. The distance between two samples quantifies the degree of similarty between the signatures of the two samples. The default method used to compute the distance between two samples is based on GSEA. Specifically, the distance between two samples A and B is computed in three steps. First the enrichment score (ES) of the signaure of sample A against the whole expression profile of sample B, ES(A, B), is compted. ES(B, A) is also computed. Since a signature is composed of a top and a bottom part, the ES of a signature in a profile is computed as the average of the ES of the top and the bottom signatures. Then, the distance between two samples is computed as the average ES:

$$d(A, B) = (ES(A, B) + ES(B, A))/2$$

Finally, a rounded value of the minimum non-zero distance is subtracted from all values; the purpose of this transformation is to expand the dynamic range and increase the relative difference between distance values.

The ES employed by default is also known as the Kolmogorov-Smirnov running sum and is analogous to the ES used in the unweighted early version of GSEA. Alternatively, a user specified function can be used to compute the distance matrix, provided using the parameter `distFun`. This function should be of the form `function(expressionData, nTop, nBottom)`, where `expressionData` is a data.frame of expression profiles and `nTop` and `nBottom` are the sizes of the signatures. This function should return a symmetric square matrix, with identical names on the rows and the columns, corresponding to the names of the samples in `expressionData`.

The distance matrix is included in the returned object and can be used to generate a graph of samples using `scudoNetwork`.

Note that we use the term distance loosely: from a mathematical point of view, our "distance" is actually a semimetric (it does not satisfy the triangle inequality).

Value

Object of class `ScudoResults`.

Author(s)

Matteo Ciciani <matteo.ciciani@gmail.com>, Thomas Cantore <cantorethomas@gmail.com>

See Also

`scudoTest`, `scudoNetwork`, `ScudoResults`

Examples

```
# generate dummy dataset
exprData <- data.frame(a = 11:20, b = 16:25,
    c = rev(1:10), d = c(1:2, rev(3:10)))
rownames(exprData) <- letters[11:20]
```

```
grps <- as.factor(c("G1", "G1", "G2", "G2"))
nTop <- 2
nBottom <- 3

# run scudo
res <- scudoTrain(exprData, grps, nTop, nBottom, foldChange = FALSE,
    featureSel = FALSE)
show(res)

# examine top signatures and top consensus signatures
upSignatures(res)
consensusUpSignatures(res)

# examine distance matrix
distMatrix(res)
```

Index

SummarizedExperiment, *3*, *13, 14*

consensusDownSignatures *(*ScudoResults-class*)*, 11
consensusDownSignatures,ScudoResults-method *(*ScudoResults-class*)*, 11
consensusUpSignatures *(*ScudoResults-class*)*, 11
consensusUpSignatures,ScudoResults-method *(*ScudoResults-class*)*, 11

distMatrix *(*ScudoResults-class*)*, 11
distMatrix,ScudoResults-method *(*ScudoResults-class*)*, 11
downSignatures *(*ScudoResults-class*)*, 11
downSignatures,ScudoResults-method *(*ScudoResults-class*)*, 11

ExpressionSet, *3*, *13, 14*

groupsAnnotation *(*ScudoResults-class*)*, 11
groupsAnnotation,ScudoResults-method *(*ScudoResults-class*)*, 11

igraph, *6*, *8–10*

legend, *10*

p.adjust, *15*
p.adjust.methods, *4*, *15*
plot.igraph, *10*

RCy3, *5*, *6*
rScudo-package, 2

scudoClassify, 3, *3*, *7*, *14*
scudoCytoscape, 5, *9*
scudoModel, *5*, 6
scudoNetwork, *3–6*, 8, *10*, *14*, *16*
scudoNetwork,ScudoResults-method *(*scudoNetwork*)*, 8
scudoParams *(*ScudoResults-class*)*, 11
scudoParams,ScudoResults-method *(*ScudoResults-class*)*, 11
scudoPlot, 10
ScudoResults, *8*, *9*, *13*, *14*, *16*
ScudoResults *(*ScudoResults-class*)*, 11
ScudoResults-class, 11
scudoTest, *11*, 12, *16*
scudoTrain, *3–5*, *11*, *13*, 14, *14*
selectedFeatures *(*ScudoResults-class*)*, 11
selectedFeatures,ScudoResults-method *(*ScudoResults-class*)*, 11
show,ScudoResults-method *(*ScudoResults-class*)*, 11

train, *5–7*
trainControl, *7*

upSignatures *(*ScudoResults-class*)*, 11
upSignatures,ScudoResults-method *(*ScudoResults-class*)*, 11

xy.coords, *10*