RTools4TB

October 25, 2011

`DBF` *Density-Based Filtering.*

Description

This function is an internal function used by `DBFMCL` to detect informative elements (*i.e.*, those that belong to dense regions). User should not use this function. Instead they can use the `DBFMCL` function with `clustering` argument set to `FALSE`.

Usage

```
DBF(data, name = NULL, distance.method = c("spearman", "pearson", "euclidean", "
```

Arguments

`data`	a matrix or data.frame
`name`	a prefix for the file name
`distance.method`	a method to compute the distance to the k-th nearest neighbor. One of "pearson" (Pearson's correlation coefficient-based distance), "spearman" (Spearman's rho-based distance), "euclidean", "spm" or "spgm". Note that the "spm" distance corresponds to the arithmetic mean of pearson- and spearman-based distance : ("pearson"+"spearman")/2 whereas "spgm" computes their geometric mean : sqrt("pearson"*"spearman").
`silent`	if set to TRUE (default), the progression of distance matrix calculation is not displayed.
`k`	the neighborhood size.
`random`	the number of simulated distributions S to compute. By default `random = FALSE`.
`fdr`	a value for the false discovery rate.
`memory.used`	size of the memory used to store part of the distance matrix. The subsequent sub-matrix is used to computed simulated distances to the k-th nearest neighbor (see detail section).
`set.seed`	specify seeds for random number generator.
`returnRank`	This argument modifies the output. Given a set of elements conserved after the filtering step of the DBFMCL algorithm, if `returnRank = TRUE` their expression values are replaced by their corresponding ranks in the input matrix.

Details

See DBFMCL

Warnings

Works only on UNIX-alikes platforms.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

DBFMCL, createSignatures4TB

DBFMCL *The "Density Based Filtering and Markov CLustering" algorithm*

Description

DBF-MCL is a tree-steps adaptative algorithm (http://tagc.univ-mrs.fr/tbrowser/) that *(i)*find elements located in dense areas (DBF), *(ii)*uses selected items to construct a graph, *(iii)*performs graph partitioning using the Markov CLustering Algorithm (MCL).

This function requires installation of the mcl program (http://www.micans.org/mcl). See "Warnings" section for more informations.

Usage

```
DBFMCL(data = NULL, filename = NULL, path = ".", name = NULL, distance.method =
 clustering = TRUE, silent = FALSE, verbose = TRUE, k = 150, random = 3, memory.
```

Arguments

data	a matrix, data.frame or ExpressionSet object.
filename	a character string representing the file name.
name	a prefix for the names of the intermediary files created by DBF and MCL.
path	a character string representing the data directory where intermediary files are to be stored. Default to current working directory.
distance.method	a method to compute the distance to the k-th nearest neighbor. One of "pearson" (Pearson's correlation coefficient-based distance), "spearman" (Spearman's rho-based distance), "euclidean". The "spm" distance corresponds to the arithmetic mean :("pearson"+"spearman")/2 whereas "spgm" is the geometric mean : sqrt("pearson"*"spearman).

clustering	indicates whether partitioning step (MCL) should be applied to the data. If clustering = FALSE, the function returns a DBFMCLresult object that contains informative elements (as detected by the DBF step) coerced into a single cluster.
silent	if set to TRUE, the progression of distance matrix calculation is not displayed.
verbose	if set to TRUE the function runs verbosely.
k	the neighborhood size.
random	the number of simulated distributions S to compute. By default random = 3.
memory.used	size of the memory used to store part of the distance matrix. The subsequent sub-matrix is used to computed simulated distances to the k-th nearest neighbor (see detail section).
fdr	an integer value corresponding to the false discovery rate (range: 0 to 100).
inflation	the main control of MCL. Inflation affects cluster granularity. It is usually chosen somewhere in the range [1.2-5.0]. inflation = 5.0 will tend to result in fine-grained clusterings, and whereas inflation = 1.2 will tend to result in very coarse grained clusterings. By default, inflation = 2.0. Default setting gives very good results for microarray data when k is set between 70 and 250.
set.seed	specify seeds for random number generator.
returnRank	allows to obtain in the DBFMCLresult object, a rank's matrix. The output files are processed using the normalization argument.

Details

When analyzing a noisy dataset, one is interested in isolating dense regions as they are populated with genes/elements that display weak distances to their nearest neighbors (i.e. strong profile similarities). To isolate these regions DBF-MCL computes, for each gene/element, the distance with its kth nearest neighbor (DKNN).In order to define a critical DKNN value that will depend on the dataset and below which a gene/element will be considered as falling in a dense area, DBF-MCL computes simulated DKNN values by using an empirical randomization procedure. Given a dataset containing n genes and p samples, a simulated DKNN value is obtained by sampling n distance values from the gene-gene distance matrix D and by extracting the kth-smallest value. This procedure is repeated n times to obtain a set of simulated DKNN values S. Computed distributions of simulated DKNN are used to compute a FDR value for each observed DKNN value. The critical value of DKNN is the one for which a user-defined FDR value (typically 10%) is observed. Genes with DKNN value below this threshold are selected and used to construct a graph. In this graph, edges are constructed between two genes (nodes) if one of them belongs to the k-nearest neighbors of the other. Edges are weighted based on the respective coefficient of correlation (*i.e.*, similarity) and the graph obtained is partitioned using the Markov CLustering Algorithm (MCL).

Value

a DBFMCLresults class object.

Warnings

With the current implementation, this function only works only on UNIX-like plateforms.

MCL should be installed. One can used the following command lines in a terminal:

```
# Download the latest version of mcl (the script has been tested successfully
with the 06-058 version).
```

```
wget http://micans.org/mcl/src/mcl-latest.tar.gz
# Uncompress and install mcl
tar xvfz mcl-latest.tar.gz
cd mcl-xx-xxx
./configure
make
sudo make install
# You should get mcl in your path
mcl -h
```

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

Van Dongen S. (2000) A cluster algorithm for graphs. National Research Institute for Mathematics and Computer Science in the 1386-3681.

See Also

createSignatures4TB

Examples

```
## Not run:
## with an artificial dataset

m <- matrix(rnorm(80000), nc=20)
m[1:100,1:10] <- m[1:100,1:10] + 4
m[101:200,11:20] <- m[101:200,11:20] + 3
m[201:300,5:15] <- m[201:300,5:15] + -2

res <- DBFMCL(data = m, distance.method = "pearson", clustering = TRUE, k = 25)
plotGeneExpProfiles(res)
plotGeneExpProfiles(res,signatures=1)

## with a real dataset
library(ALL)
data(ALL)
sub <- exprs(ALL)[1:3000,]

#First, we will normalize the data set using the doNormalScore function.
subNorm <- doNormalScore(sub)
res <- DBFMCL(subNorm, distance.method="pearson", memory=512)

#The results are stored in an instance of class DBFMCLresult.
class(res)
res
```

```
# The expression matrix is stored in the data slot.
# This matrix contains only genes detected as informative (that is falling into a cluster
head(res@data[,1:2])

# The partitioning results are stored in the cluster slot.
slotNames(res)

# Here, 3 TS were found.
res@size

# The following instruction can be used to get the expression matrix corresponding to the
res@data[res@cluster ==1,]

# The high level function plotGeneExpProfilescan be used to visualize,
# for instance, gene expression profiles corresponding to the first signature.
plotGeneExpProfiles(res, sign=1)

#To stored the partitioning results onto disk (as a tab-delimited file),
# use the writeDBFMCLresult function as show below.
writeDBFMCLresult(res, filename.out="ALL.sign.txt")


## End(Not run)
```

`DBFMCLresult-class` *Class to store DBFMCL results.*

Description

This class represents the results of the `DBFMCL` algorithm.

Objects from the Class

Objects can be created by calls of the form `new('DBFMCLresult',`

```
name = ...., # Object of class character
data = ...., # Object of class matrix
cluster = ...., # Object of class vector
size = ...., # Object of class vector
center = ...., # Object of class matrix
parameters = ...., # Object of class list
)
```

Slots

`name`**:** Object of class `"character"`, an analysis identifiant (by default "exprs").

`data`**:** Object of class `"matrix"`, a subset of the original matrix containing the coordinates of the selected elements.

`cluster`**:** Object of class `"vector"`, a vector of integers indicating the cluster to which each point is allocated.

`size`**:** Object of class `"vector"`, the number of points in each cluster.

`center`**:** Object of class `"matrix"`, a matrix of cluster centres.

`parameters`**:** Object of class `"list"`, a list of all used DBFMCL parameters: normalizationMethod, distanceMethod, k, random, fdr, set.seed, inflation.

Methods

show `signature(object = "DBFMCLresult")`, provides informations about the object.

Author(s)

Bergon A., Lopez F., Textoris J. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

`writeDBFMCLresult`, `DBFMCL`

Examples

```
obj <- new("DBFMCLresult")
obj
```

MCL *Invokes the Markov CLustering algorithm (MCL).*

Description

This function invokes the mcl system command. MCL is a clustering algorithm for graphs that was developped by Stijn van Dongen (see references for further informations).

Usage

```
MCL(name, inflation = 2.0, silent = FALSE)
```

Arguments

`name`	a character string corresponding to the file name.
`inflation`	the main control of MCL. Inflation affects cluster granularity. It is usually chosen somewhere in the range `[1.2-5.0]`. `inflation = 5.0` will tend to result in fine-grained clusterings, and whereas `inflation = 1.2` will tend to result in very coarse grained clusterings. By default, `inflation = 2.0`. Default setting gives very good results for microarray data when k is set around 100.
`silent`	if set to TRUE, the progression of the MCL partitionning is not displayed.

Value

Returns a file with the ".mcl_out.txt" extension.

warning

Works only on UNIX-like plateforms. MCL should be installed. The following command lines can be used for installation.

```
# Download the latest version of mcl (RTools4TB has been tested successfully
with the 06-058 version).
wget http://micans.org/mcl/src/mcl-latest.tar.gz
# Uncompress and install mcl
tar xvfz mcl-latest.tar.gz
cd mcl-xx-xxx
./configure
make
sudo make install
# You should get mcl in your path
mcl -h
```

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Stijn van Dongen. A cluster algorithm for graphs. Technical Report INS-R0010, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, May 2000. http://www.cwi.nl/ftp/CWIreports/INS/INS-R0010.ps.Z

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

`RTools4TB_1.1.4-package`

The RTools4TB package: data mining of public microarray data through

---

Description

TranscriptomeBrowser (TBrowser, http://tagc.univ-mrs.fr/tbrowser) hosts a large collection of transcriptional signatures (TS) automatically extracted from the Gene Expression Omnibus (GEO) database. Each GEO experiment (GSE) was processed so that a subset of the original expression matrix containing the most relevant/informative genes was kept and organized into a set of homogeneous signatures. Each signature was tested for functional enrichment using annotations terms obtained from numerous ontologies or curated databases (Gene Ontology, KEGG, BioCarta, Swiss-Prot, BBID, SMART, NIH Genetic Association DB, COG/KOG...) using the DAVID knowledgebase. The RTools4TB package can be used to perform complexe queries to the database. RTools4TB can be helpful (i) to define the biological contexts (i.e, experiments) in which a set of genes are co-expressed and (ii) to define their most frequent neighbors.

In addition, RTools4TB comes with a new algoritm, "Density Based Filtering And Markov Clustering" (DBF-MCL), whose goal is to partition large and noisy datasets. DBF-MCL is a tree-step

adaptative algorithm that (i) find elements located in dense areas (ie. clusters) (ii) uses selected items to construct a graph and (iii) performs graph partitioning using MCL. This algorithm is implemented in the RTools4TB package although it requires a UNIX-like systems.

Details

Package:	RTools4TB
Type:	Package
Version:	1.1.43
Date:	2008-10-08
License:	LGPL
Notes:	Please note that a web connection is required to fetch expression data from the TrancriptomeBrowser database.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D. Maintainer: Aurelie Bergon <bergon@tagc.univ-mrs

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

---

annotationList *The keywords used to get information about functional enrichment of*

Description

A dataframe containing the keywords used to get information about functional enrichment of transcriptional signatures.

Usage

```
data(annotationList)
```

Examples

```
data(annotationList)
names(annotationList)
attach(annotationList)
table(TableName)
annotationList[1:4,]
head(annotationList[TableName=="KEGG_PATHWAY",])
```

clusterEisen *Performs system call to the Cluster 3.0 program.*

Description

This function does a system call to the Cluster 3.0 program. It requires Cluster 3.0 to be installed on the machine and to be available in the path (as "cluster"). This internal function is used by createSignatures4TB. it performs only clustering of the columns (using a Pearson's correlation coefficient-based distance).

Usage

```
clusterEisen(filename, median.center = FALSE, silent = FALSE)
```

Arguments

filename	a character string representing the file name.
median.center	a logical indicating whether rows should be median-centered.
silent	if set to TRUE, the progression of clustering is not displayed.

Details

Cluster was originally written by Michael Eisen (http://rana.lbl.gov/EisenSoftware.htm) The command line version of Cluster version 3.0 (for Windows, Mac OS X, Unix, and Linux) was created by Michiel de Hoon, together with Seiya Imoto and Satoru Miyano.

Warnings

Only tested on UNIX-alikes plateforms.

Cluster 3.0 should be installed in its command-line only version:

```
wget http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/cluster-1.36.tar.gz
tar xvfz cluster-1.36.tar.gz
cd cluster-1.36/
./configure --without-x
make
sudo make install
# You should get cluster in your path
cluster -v
```

Please see http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/software.htm for further informations.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Open source clustering software. De Hoon MJ, Imoto S, Nolan J, Miyano S. Bioinformatics. 2004 Jun 12;20(9):1453-4.

Cluster analysis and display of genome-wide expression patterns. Eisen MB, Spellman PT, Brown PO, Botstein D. Proc Natl Acad Sci U S A. 1998 Dec 8;95(25):14863-8.

`colorScale` *Calibration bar for color images*

Description

This function is the same as the `maPalette` function in the marray library. It is used to produce a color scale.

Usage

```
colorScale(low = "white", high = c("green", "red"), mid = NULL, k = 50)
```

Arguments

`low`	Color for the lower end of the color palette, specified using any of the three kinds of R colors, i.e., either a color name (an element of colors), a hexadecimal string of the form `#rrggbb`, or an integer i meaning `palette()[i]`
`high`	Color for the upper end of the color palette, specified using any of the three kinds of R colors, i.e., either a color name (an element of colors), a hexadecimal string of the form `#rrggbb`, or an integer i meaning `palette()[i]`
`mid`	Color for the middle portion of the color palette, specified using any of the three kinds of R colors, i.e., either a color name (an element of colors), a hexadecimal string of the form `#rrggbb`, or an integer i meaning `palette()[i]`
`k`	Number of colors in the palette

Author(s)

see the `maPalette`

Examples

```
Rcol <- colorScale(low="white", high="red", k=10)
Gcol <- colorScale(low="white", high="green", k=50)
RGcol <- colorScale(low="green", high="red", k=100)
```

createGraph4BioC *Creates a graph based on a request to the TrancriptomeBrowser database.*

Description

Results from a request to TBrowserDB can be displayed as a graph using the createGraph4BioC function. Given a request (e.g., "XBP1 & ESR1 & GATA3"), createGraph4BioC retrieves the list of signature IDs that verify the constrain. A list of gene falling in at least one of the clusters is next computed. A matrix is created that will record for each gene the number of time they have been observed in the same signature (only genes falling in a significant proportion of signatures are conserved). This adjacency matrix is used to create a graph.

Usage

```
createGraph4BioC(request = NULL, prop = 50)
```

Arguments

request	a Boolean query with HUGO IDs (e.g., "CD4 & CD3E & CD3D")
prop	only genes falling in a significant proportion of signatures are conserved. For instance, prop=50 indicates that only genes falling in at least 50 % of the signatures will be conserved.

Details

In order to create an adjacence matrix, this function use the both getSignatures and getExpressionMatrix functions.

The "value" argument to this function may contain Boolean operators (see help section on TBrowser web site for more informations, http://tagc.univ-mrs.fr/tbrowser or the details section of getSignatures).

& : AND | : OR ! : NOT , (used in conjonction with &)

Value

an adjacence matrix.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

Examples

```
## Not run:
# Create a graph based on expression signatures that contain "ESR1 & GATA3 & XBP1".
# Only genes observed in 80
library(biocGraph)
adjMat <- createGraph4BioC(request="ESR1 & GATA3 & XBP1", prop=80)
g1 <- new("graphAM", adjMat=adjMat)
plot(g1,"fdp")
g2 <- as(g1,"graphNEL")

## End(Not run)
```

`createSignatures4TB` *Creates a set of transcriptional signatures from a microarray dataset.*

Description

This function is a wrapper to create sets of transcriptional signatures (as in the Transcriptome-Browser Project, TBrowser,http://tagc.univ-mrs.fr/tbrowser). This function creates a "cdt" file containing a set of expression matrices (transcriptional signatures) separated by blank lines. Please note that it requires both MCL and Cluster 3.0 (see 'warnings section'). It accepts both a matrix or file name as input.

Usage

```
createSignatures4TB(data = NULL, filename = NULL, path = ".", name = NULL, norma
 distance.method = c("spearman", "pearson", "euclidean", "spm", "spgm"), silent
 inflation = 2.0, median.center = FALSE, set.seed = 123, returnRank = FALSE)
```

Arguments

`data`	a `matrix`, `data.frame` or `ExpressionSet` object.
`filename`	a character string representing the file name to load.
`path`	a character string representing the data directory.
`name`	a prefix for the name of the created files.
`normalization`	indicates whether data should be normalized prior to analysis (see details).
`distance.method`	a method to compute the distance to the k-th nearest neighbor. One of "pearson" (Pearson's correlation coefficient-based distance), "spearman" (Spearman's rho-based distance), "euclidean", "spm" or "spgm". Note that the "spm" distance corresponds to the arithmetic mean of pearson- and spearman-based distance : ("pearson"+"spearman")/2 whereas "spgm" computes their geometric mean : sqrt("pearson"*"spearman").
`silent`	if set to TRUE, the progression of distance matrix calculation is not displayed.
`verbose`	if set to TRUE the function runs verbosely.
`k`	the neighborhood size.
`random`	the number of simulated distributions S to compute. By default `random = 3`.

`memory.used` size of the memory used to store part of the distance matrix. The subsequent sub-matrix is used to computed simulated distances to the k-th nearest neighbor (see detail section of DBFMCL function).

`fdr` an integer value corresponding to the false discovery rate (range: 0 to 100).

`inflation` the main control of MCL. Inflation affects cluster granularity. It is usually chosen somewhere in the range `[1.2-5.0]`. `inflation = 5.0` will tend to result in fine-grained clusterings whereas `inflation = 1.2` will tend to result in very coarse grained clusterings. By default, `inflation = 2.0`. Default setting gives very good results for microarray data.

`median.center` if set to TRUE, median-centering is applied to the rows of the matrix.

`set.seed` specify seeds for random number generator.

`returnRank` This argument modifies the output. Given a set of elements conserved after the filtering step of the DBFMCL algorithm, if `returnRank = TRUE` their expression values are replaced by their corresponding ranks in the input matrix.

Details

The Markov Cluster Algorithm was written by S. Van Dongen (see reference section). Cluster was originally written by Michael Eisen (http://rana.lbl.gov/EisenSoftware.htm). The command line version of Cluster version 3.0 was created by Michiel de Hoon, together with Seiya Imoto and Satoru Miyano.

Warnings

With the current implementation, this function works only on UNIX-like plateforms.

Cluster 3.0 should be installed in its command-line only version:

Please see http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/software.htm for further informations.

```
wget http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/cluster-1.36.tar.gz
tar xvfz cluster-1.36.tar.gz
cd cluster-1.36/
./configure --without-x
make
sudo make install
# You should get cluster in your path
cluster -v
```

MCL should be installed:

```
# Download the latest version of mcl (the script has been tested successfully with the 06-058 version).
wget http://micans.org/mcl/src/mcl-latest.tar.gz
# Uncompress and install mcl
tar xvfz mcl-latest.tar.gz
cd mcl-xx-xxx
./configure
```

```
make
sudo make install
# You should get mcl in your path
mcl -h
```

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

Van Dongen S. (2000) A cluster algorithm for graphs. National Research Institute for Mathematics and Computer Science in the 1386-3681.

Open source clustering software. De Hoon MJ, Imoto S, Nolan J, Miyano S. Bioinformatics. 2004 Jun 12;20(9):1453-4.

Cluster analysis and display of genome-wide expression patterns. Eisen MB, Spellman PT, Brown PO, Botstein D. Proc Natl Acad Sci U S A. 1998 Dec 8;95(25):14863-8.

See Also

DBFMCL,heatmapFromCDT,plotGeneExpProfiles,getSignatures,getExpressionMatrix

Examples

```
## Not run:
## with an artificial dataset

m <- matrix(rnorm(80000), nc=20)
m[1:100,1:10] <- m[1:100,1:10] + 4
m[101:200,11:20] <- m[101:200,11:20] + 3
m[201:300,5:15] <- m[201:300,5:15] + -2

res <- createSignatures4TB(data = m, name="artificial", distance.method = "pearson", medi

plotGeneExpProfiles(res)
allsign <- heatmapFromCDT("artificial.dataMods.cdt")

plotGeneExpProfiles(res, signature=1)
heatmapFromCDT("artificial.dataMods.cdt", signature=1)

## with a real dataset
library(ALL)
data(ALL)
exp <- createSignatures4TB(data = ALL , name="ALLdataset", distance.method = "pearson", m
plotGeneExpProfiles(exp, signatures=1)
plotGeneExpProfiles(res)
allsign <- heatmapFromCDT("ALLdataset.dataMods.cdt")
si1 <- heatmapFromCDT("ALLdataset.dataMods.cdt", signature=1)

## End(Not run)
```

`doNormalScore` *Normal score transformation of a matrix.*

Description

This function performs normal score transformation of a matrix. The normal score transformation ranks each column from lowest to the highest values and matches these ranks to equivalent ranks from a normal distribution.

Usage

```
doNormalScore(sdata,set.seed = 123)
```

Arguments

`sdata`	a matrix.
`set.seed`	specify seeds for random number generator.

Value

A matrix.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

Examples

```
m <- matrix(rnorm(1000),nc=4)
m[,1] <- m[,1] + 4
boxplot(as.data.frame(m))
m <- doNormalScore(m)
boxplot(as.data.frame(m))
```

`doRankTransformation` *Rank transformation of a matrix.*

Description

This function performs rank transformation of a matrix (each column is transformed so that each values is replaced by its corresponding ranks.).

Usage

```
doRankTransformation(data = NULL)
```

Arguments

data	a matrix

Value

A matrix.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

Examples

```
m <- matrix(rnorm(1000),nc=4)
m <- doRankTransformation(m)
head(m)
```

---

getData4DBFMCL *Fetch an expression matrix from a file or an ExpressionSet object*

Description

This function retrieves the expression matrix from a file or an `ExpressionSet` object. This is an internal function called by createSignatures4TB that should not be used directly.

Usage

```
getData4DBFMCL(data = NULL, filename = NULL, path = ".")
```

Arguments

data	a matrix
filename	a character string representing the file name.
path	A character string representing the data directory.

Value

This function return a list which contains a matrix and a name. If filename is not equal to NULL, this name corresponds to the prefix of the given filename, in the other case name = NULL.

Warnings

Convert data.frame, expressionSet or tab-delimited file to matrix class object. The input data must contain an expression matrix with gene as rows and samples as columns. Note that space characters inside gene names are not allowed (as they are not supported by the mcl command-line program).

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

`createSignatures4TB`

---

getExpressionMatrix

A function to fetch an expression matrix from the TranscriptomeBrowser

---

Description

This function takes a signature ID as input and retrieves the corresponding expression matrix from the TranscriptomeBrowser database (`http://tagc.univ-mrs.fr/tbrowser`).

Usage

```
getExpressionMatrix(signatureID = NULL, verbose = TRUE, save = FALSE)
```

Arguments

`signatureID`	a TBrowserDB signature ID (e.g., "0AC1A39A5"), which can be obtained using the `getSignatures` function.
`verbose`	if set to TRUE the function runs verbosely.
`save`	if set to TRUE data are stored onto disk.

Value

A `data.frame` containing the expression matrix of the requested signature. The first two columns store probe and gene informations and additional columns contain expression values. for corresponding samples (as columns)

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

Examples

```
## Not run:
## what are the TBrowserDB signatures that contain both XBP1, GATA3 and ESR1 ?
library(RColorBrewer)
res <- getSignatures(field="gene", value="XBP1 & GATA3 & ESR1")
res

# Retrieve expression matrix for the second signature.
em <- getExpressionMatrix(signatureID = "3DE64836D")

#Getting gene names and sample informations
geneNames <- paste(em[,1],em[,2],sep="||")
em <- as.matrix(em[,-c(1,2)])
samplesInfo <- getTBInfo(field="samples", value = "3DE64836D")
ind <- match(colnames(em), samplesInfo[,1])
colnames(em) <- samplesInfo[ind,2]

# Heatmap
col <- colorRampPalette(brewer.pal(10, "RdBu"))(256)
split <- strsplit(colnames(em)," (", fixed=TRUE)
pheno <- unlist(lapply(split,"[",1))
pheno <- as.factor(pheno)
levels(pheno) <- 1:5
cc <- rainbow(5, start=0, end=.3)
cc <- cc[pheno]
heatmap(em, Rowv=FALSE, Colv=FALSE, col=col, ColSideColors=cc, labRow=geneNames,cexRow=0.

## End(Not run)
```

`getSignatures` *A function to retrieve transcriptional signature IDs from the*

Description

This is one of the main function of the **RTools4TB** package. It allows direct access to TBrowserDB (http://tagc.univ-mrs.fr/tbrowser). The `getSignatures` function can be used to retrieve transcriptional signatures (i)derived from a given experiment or microarray platform, (ii)containing a user-defined list of genes or probes (using or not a boolean query) or (iii)enriched in genes sharing a common annotation term (user must provide a q-value).

See "Details" section for more information about the syntax.

Usage

```
getSignatures(field=c("gene", "probe", "platform", "experiment", "annotation"),
```

Arguments

field	The request type. Should be one of: "gene", "probe", "platform", "experiment", "annotation".
value	Depends on the field argument: if field is set to "gene" value must contain HUGO IDs (*e.g.*, "CD4 CD3E CD3D"). Logical operators are supported (*e.g.*, "CD4 & CD3E & CD3D", see "details" section). if field is set to "probe" value must contain a list of probe IDs (*e.g.*, Affymetrix probe IDs). Logical operators are supported. if field is set to "platform" value must contain one platform ID (*e.g.*, "GPL96"). if field is set to "experiment" value must contain one experiment ID (*e.g.*, "GSE2004"). if field is set to "annotation" value must contain a list of annotation terms separated by logical operators(*e.g.*, "breast cancer" or "18q11.2|18q12.1|18q21.1|18q22-q23").
qValue	an integer (10E-"qValue"). Default to 0. This q-value is used to select signatures associated with a given annotation term (see examples section). Used only when field = "annotation".
nbMin	an integer. Used only when value corresponds to a gene list without logical operators (see details). Only signatures containing at least nbMin genes out of the list will be retrieved (see details section).
verbose	if set to TRUE the function runs verbosely.
save	if set to TRUE data are stored onto disk.

Details

The "value" argument to getSignatures may contain logical operators (see help section on TBrowser web site for more informations, http://tagc.univ-mrs.fr/tbrowser)

& : AND | : OR ! : NOT , (used in conjonction with &)

However, when field = "gene" or field = "probe", user can perform a request using a list of item separated by blanks (without logical operators). These blanks are interpreted as the OR logical operators. In this case, all signatures containing at least one gene of the list will be returned. To select more informative signatures we suggest to use the nbMin argument that will select signatures containing at least nbMin genes out of the list.

Moreover, user may include logical operators in the request. Indeed, this is a convenient way to create relevant queries. Suppose your field of interest is related to T-cell activation. You could be interested in retrieving all TS that contain the CD4 gene as they should contain additional T cell markers. Comparing these TS should help you to define a set of frequent CD4 neighbors (very likely related to TCR signaling cascade). Thereby, your request should be:

```
res <- getSignatures(field="gene", value="CD4")
```

This gene is found in 371 TS (with the current database release), and obtaining associated gene lists would be time consuming and would not emphasize on what you are really expecting. Indeed, the CD4 marker is also expressed by macrophages. Another solution would be to search for TS containing two T-cell markers (CD4 and CD3E for instance) and to exclude (using the NOT operator) those containing the CD14 marker (a macrophages marker). The syntax should be the following:

```
res <- getSignatures(field="gene", value="CD4 & CD3E & !CD14")
```

In the same way you could try to exclude TS containing B-cells by discarding those containing the CD19 of IGHM marker. The resulting query would be the following:

```
res <- getSignatures(field="gene", value="CD4 & CD3E & !(CD19 | IGHM)")
```

Value

This function will return a vector containing the names of the transcriptional signatures that satisfy the constraints. Additional informations about these signatures (GEO platform ID, GEO experiment ID, Organism, number of probes, number of genes, number of biological samples) can be obtained using the `getTBInfo` function (`field = "signatureID"`).

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

Other functions which allow to query the TBrowser database: `getTBInfo`, `getExpressionMatrix`

Examples

```
## Not run:
# retrieving transcriptional signatures containing PCNA, CDC2 and CDC6.
res <- getSignatures(field="gene", value="PCNA & CDC2 & CDC6")

# retrieving transcriptional signatures contain at least two genes out of the following l
res <- getSignatures(field="gene", value="PCNA CDC2 CDC6", nbMin=2)

# retrieving transcriptional signatures related to GSE2004
gse2004TS <- getSignatures(field="experiment", value="GSE2004")

# retrieving transcriptional signatures related to the platform GPL96
gpl96TS <- getSignatures(field="platform", value="GPL96")

# retrieving transcriptional signatures enriched in gene related to the keyword ""HSA0411
data(annotationList)
attach(annotationList)
table(TableName)
annotationList[Keyword=="HSA04110:CELL CYCLE",]
ccTS20 <- getSignatures(field="annotation", value="HSA04110:CELL CYCLE", qValue=20)


# retrieving transcriptional signatures enriched in gene located in 8q region.
query <- paste(grep("^8q", Keyword, val = T), collapse = "|")
query
cc <- getSignatures(field = "annotation", value = query, qValue = 10)


## End(Not run)
```

`getTBInfo` *Get information about an experiment or a platform.*

Description

This function fetch informations from the TBrowserDB for a microarray experiment, a microarray platform or a transcriptional signature.

Usage

```
getTBInfo(field = c("platform", "experiment", "signature", "samples"), value = N
```

Arguments

`field`	The type of information to retrieve. Should be one of "platform", "experiment" or "signature".
`value`	A platform ID (e.g., GPL96), experiment ID (e.g., GSE2004) or signatureID (e.g., "0AC1A39A5") depending on the "field" argument.
`verbose`	If set to TRUE the fonction displays informations on the screen.
`save`	if set to TRUE data are stored onto disk.

Value

The output will differ depending one the `field` argument.

if `field = "platform"`, the function will display plateform informations (e.g.: name, organism, manufacturer, nb. probes, nb. genes, Title and Description).

if `field = "experiment"`, the function will display informations about the experiment (e.g.: name, organism, PMID, nb. samples, title and summary).

if `field = "signature"`, the function will display informations about the experiment (e.g.: signatureID, platform, experiment, organism, nb.probes, nb.genes, nb.samples).

if `field = "samples"`, the function will display informations about all the samples from an signatureID (e.g.: sampleID, descriptions).

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

Other function which allow to query the TBrowser database: `getSignatures`, `getExpressionMatrix`

Examples

```
## Not run:
# retrieving information related to GSE2004
gse2004Info <- getTBInfo(field="experiment", value="GSE2004")

# retrieving information related to GPL96
gpl96Info <- getTBInfo(field="platform", value="GPL96")

# retrieving information related to a signature
signInfo <- getTBInfo(field="signature", value="0AC1A39A5")

# retrieving samples information related to a signature
samplesInfo <- getTBInfo(field="samples", value="0AC1A39A5")

## End(Not run)
```

`heatmapFromCDT` *Reads a "cdt" file containing a set of transcriptional signatures and*

Description

This function can be called to display the results of the createSignatures4TB function.

Usage

```
heatmapFromCDT(cdt.filename, signature = NULL, fac = NULL)
```

Arguments

`cdt.filename`	a character string representing the "cdt" file name.
`signature`	a vector. By default (`signature = NULL`), all signatures are displayed.
`fac`	a factor. By default `fac = NULL`, no phenodata are displayed. If fac corresponds to a factor, the several levels of fac are displayed by a distinct color.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

`createSignatures4TB`, `plotGeneExpProfiles`

Examples

```
## Not run:
library(ALL)
data(ALL)
res <- createSignatures4TB(data=ALL, name="ALLdataset", median.center=TRUE, distance.meth
all <- heatmapFromCDT("ALLdataset.dataMods.cdt")
sig6 <- heatmapFromCDT("ALLdataset.dataMods.cdt", signature=6)
rownames(sig6)

## End(Not run)
```

---

matplotProfiles *An internal plotting function based on the matplot function.*

Description

An internal function based on the matplot function.

Usage

```
matplotProfiles(data, imgName, saveHTML = FALSE, X11 = TRUE, verbose = FALSE)
```

Arguments

data	a matrix
imgName	a prefix for image name.
saveHTML	a logical indicating if results should be stored in a HTML file.
X11	if set to TRUE a new graphic windows is generated.
verbose	if set to TRUE the function runs verbosely.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

See Also

Used in the plotGeneExpProfiles function.

`plotGeneExpProfiles` *Visualization of transcriptional signature profiles.*

Description

This function is used to visualize expression profiles of transcriptional signatures stored in a DBFMCLresult object or in a tab-delimited file.

Usage

```
plotGeneExpProfiles(data = NULL, filename = NULL, path = ".", signatures = NULL,
```

Arguments

`data`	a DBFMCLresult class object or a matrix.
`filename`	a character string representing the file name (if data are loaded from a flat file).
`path`	a character string representing the data directory.
`signatures`	a vector that indicates which signatures to plot.
`saveHTML`	if set to TRUE a HTML file is created.
`filename.out`	a character string representing the HTML file name (if `saveHTML=TRUE`).
`X11`	if set to TRUE a new graphic windows is generated.
`verbose`	if set to TRUE the function runs verbosely.

Details

Mean expression profile is highlighted in green.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

`matplotProfiles`

Examples

```
## Not run:
# Please check that the directory is writable
# before using the following code
library(ALL)
data(ALL)
ALLnorm <- doNormalScore(exprs(ALL))
res <- DBFMCL(data=ALLnorm, name="ALLout")
plotGeneExpProfiles(res)
plotGeneExpProfiles(res, signatures=1:2)
plotGeneExpProfiles(res, signatures=1:2, saveHTML=TRUE)

## End(Not run)
```

---

writeDBFMCLresult *Writes the results of DBFMCL function onto disk.*

Description

Writes the results of DBFMCL function onto disk. The output file contains an expression matrix in which transcriptional signatures are separed by blank lines.

Usage

```
writeDBFMCLresult(object, filename.out = NULL, path = ".", verbose = TRUE)
```

Arguments

`object`	a `DBFMCLresults` class object.
`filename.out`	a character string representing the file name where the data will be stored.
`path`	a character string representing the data directory where the data will be stored.
`verbose`	if set to TRUE the function runs verbosely.

Value

A tab-delimited file.

Author(s)

Bergon A., Lopez F., Textoris J., Granjeaud S. and Puthier D.

References

Lopez F.,Textoris J., Bergon A., Didier G., Remy E., Granjeaud S., Imbert J. , Nguyen C. and Puthier D. TranscriptomeBrowser: a powerful and flexible toolbox to explore productively the transcriptional landscape of the Gene Expression Omnibus database. PLoSONE, 2008;3(12):e4001.

See Also

`DBFMCLresult-class`, `DBFMCL`

Examples

```
## Not run:
library(ALL)
data(ALL)
ALLnorm <- doNormalScore(exprs(ALL))
res <- DBFMCL(data=ALLnorm, name="ALLout")
plotGeneExpProfiles(res)
writeDBFMCLresult(res, "ALL.sign.txt")

## End(Not run)
```

Index