SpeCond

October 25, 2011

`SpeCond` *The condition-specific detection function*

Description

`SpeCond` performs a full condition-specific detection from an expression matrix

Usage

```
SpeCond(expressionMatrix, param.detection = NULL,
multitest.correction.method = "BY", prefix.file = "A",
print.hist.pv = FALSE, fit1 = NULL, fit2 = NULL,
specificOutlierStep1 = NULL, condition.factor=NULL,
condition.method=c("mean","max"))
```

Arguments

`expressionMatrix`
: an ExpressionSet object or a matrix of expression values (in log2); columns are the conditions, rows are genes (or probe sets)

`param.detection`
: the parameter matrix for the detection must contain the values for "lambda", "beta", "per", "md", "mlk" ,"rsd' and "pv" for the two steps of the procedure. Can be obtained by `getDefaultParameter` or `createParameterMatrix`

`multitest.correction.method`
: the multitest correction method. The default is "BY", for the possible values see `p.adjust`

`prefix.file`
: a prefix added to the histogram file (if produced). It will be used to link to the result html pages generated by other functions using the result object of this function (if no other prefix value is implemented). The default is "A". It is useful to change the prefix when you perform a new analysis with different parameters as you may want to compare the results

`print.hist.pv`
: a logical (TRUE/FALSE) value indicating whether a histogram of (non-adjusted) p-values is to be printed; the default is FALSE Optional parameters:

`fit1`
: the result of `fitPrior` containing the parameter of the mixture normal model of the expression data. If NULL `fitPrior` function will be called

`fit2` the result of `fitNoPriorWithExclusion` containing the parameter of the mixture normal model of the expression data ignoring the outliers detected in the first step of the procedure. If NULL, `fitNoPriorWithExclusion` function will be called

`specificOutlierStep1` the list of outliers detected by the first step procedure, result of the `getSpecificOtuliersStep1` function. If NULL, `getSpecifcOutliersSpep1` will be called

`condition.factor` this argument can be used if expressionMatrix is an ExpressionSet object; a factor object of length equal to the number of columns (samples) of the ExpressionSet object specifying which sample(s) belong to which condition (condition.factor levels); can be extracted from the phenoData

`condition.method` this argument can be used if expressionMatrix is an ExpressionSet object; the method (mean or max) to summarise the samples by conditions (defined by the condition.factor vector)

Details

SpeCond uses the `Mclust` function to obtain the mixture of normal distributions uses by the detection procedure.

If `expressionMatrix` is an ExpressionSet object it is necessary to obtain an expression value matrix. This is obtain by the `getMatrixFromExpressionSet` function. This take into consideration if `condition.factor` is not NULL the transformation of the expression values for the several samples of each condition to one expression values for each condition for each gene.

If `print.hist.pv` is TRUE the histogramme of the non-adjusted p-values is plotted. It is a way to check the normal distribution fitting. If the histogramme is relatively flat the normal distribution(s) fits properly the data.

Value

An object of class `sp_list`

`prefix.file` the prefix used for this analysis. It will be used by default in the function `getFullHtmlSpeCondResult` and `getGeneHtmlPage`

`fit1` the fitting parameters used by the detection in the first step of the procedure: the result of the `fitPrior` function called internally or the same as the fit1 argument if not NULL

`fit2` the fitting parameters used by the detection in the second step of the procedure: the result of the `fitNoPriorWithExclusion` function called internally or the same as the fit2 argument if not NULL

`specificOutliersStep1` the condition(s) for which the expression value of the gene is detected as outlier in the first step of the procedure. If NULL, no expression value has been detected in the first step. The second fitting ignores these expression values

`specificResult` a list of 7 attributes containing al the specific results. This object result of `getSpecificResult`

Author(s)

Florence Cavalli, florence@ebi.ac.uk

References

C.Fraley and A. E. Raftery, Model-based clustering, discriminant analysis, and density estimation, *Journal of the American Statistical Association, Vol. 97, pages 611-631 (2002).*

C. Fraley and A. E. Raftery, MCLUST Version 3 for R: Normal Mixture Modeling and Model-based Clustering, Technical Report No. 504, Department of Statistics, University of Washington, September 2006.

See Also

`Mclust`, `fitPrior`, `fitNoPriorwithExclusion`, `getSpecificOutliersStep1`, `getSpecificResul`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
dim(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample, param.detection=NULL,
multitest.correction.method="BY", prefix.file="E", print.hist.pv=TRUE,
fit1=NULL, fit2=NULL, specificOutlierStep1=NULL)
```

`createParameterMatrix`

Create or Modify the SpeCond argument parameters

Description

`createParameterMatrix` creates and/or modifies the param.detection matrix used as argument in the SpeCond function. If parm.detection is NULL the param.detection matrix used is the one containing the default parameter values, as obtained by `getDefaultParameter`. The remaining arguments enable to change the values of the param.detection matrix.

Usage

```
createParameterMatrix(param.detection = NULL, beta.1 = NULL, beta.2 = NULL,
lambda.1 = NULL, lambda.2 = NULL, per.1 = NULL, per.2 = NULL, md.1 = NULL,
md.2 = NULL, mlk.1 = NULL, mlk.2 = NULL, rsd.1 = NULL, rsd.2 = NULL,
pv.1 = NULL, pv.2 = NULL)
```

Arguments

`param.detection`	a matrix of 2 rows and 7 columns as the result of `getDefaultParameter`
`beta.1`	Influences the prior applied during the determination of the variance of the normal distributions. It is necessary in the first fitting step to allow the model to capture isolated outliers.
`beta.2`	The normal use of SpeCond does not prior on Step2: must be set to 0
`lambda.1`	Influences the choice of models by affecting the selection of one, two or three normal distributions, thus introducing some weight on the effect of number of parameters to be defined. The default is 1, the model uses the BIC value taking into account the log-likelihood value.

`lambda.2`	Same as `lambda.1` for the second step of the SpeCond function
`per.1`	percentage threshold: this is the percentage of conditions that can be detected as specific. As per increases a larger number of expression values per genes can be identified as specific. The default is 0.1
`per.2`	percentage threshold: This is the final percentage of condition that can be detected as specific. As per increases a larger number of expression values per genes can be identified as specific. The default is 0.3
`md.1`	median difference: this is the minimum value between the median values of two mixture components that is allowed to identify one of them as representing outliers, i.e. possibly not part of the null distribution. This corresponds to a biological fact; specific expression that corresponds to noise should not be detected as specific
`md.2`	Same as `md.1` for the second step of the SpeCond function. For consistency should be equal to `md.1`
`mlk.1`	minimum log-likelihood: enables the identification of clusters of conditions that are well separated from the others in the model. If the gene mlk value>mlk, the mixture component can be detected as outlier (i.e. not part of the null distribution)
`mlk.2`	same as `mlk.2` for the second step of the SpeCond function
`rsd.1`	minimum of standard deviation ratio: enables the identification of clusters of conditions that are extremely spread out compared to the distribution clustering of most expression values. If the gene rsd values< rsd the mixture component can be detected as outlier (i.e. not part of the null distribution)
`rsd.2`	same as `rsd.1` for the second step of the SpeCond function
`pv.1`	p-value threshold to detect a condition as specific for a given gene
`pv.2`	same as `pv.1` for the second step of the SpeCond function, for consistency should be equal to pv.1

Value

param.detection: a matrix of 2 row and 7 columns. The rows "Step1 "and "Step2" correspond respectively to the first and second set of parameters for the SpeCond function. The parameters (columns) are: lambda, beta, per, md, mlk, rsd. See the `createParameterMatrix` documentation for more details about the parameters.

Warning

The SpeCond code is based on: beta.2=0 md.1=md.2 per.1<=per.2 pv.1=pv.2

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`getDefaultParameter`

Examples

```
##Get the default parameters and changing the mlk.1 value to 10:
param.detection2=createParameterMatrix(mlk.1=10)
param.detection2
## Modify param.detection2 with mlk.1 value to 15 and rsd.2 value to 0.2
param.detection2B=createParameterMatrix(param.detection=param.detection2,
mlk.1=10, rsd.2=0.2)
param.detection2B
```

---

expSetSpeCondExample

An ExpressionSet example object used in the SpeCond package

Description

expSetSpeCondExample is an ExpressionSet example object used as an example for the SpeCond package. This ExpressionSet only contains an expression matrix and the phenoData. This object has only the purpose of illustrating how SpeCond can be used with an ExpressionSet input object.

Usage

```
data(expSetSpeCondExample)
```

Format

The format is: Formal class 'ExpressionSet' [package "Biobase"] with 7 slots ..@ phenoData :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots .. .. ..@ varMetadata :'data.frame': 2 obs. of 1 variable: .. ..$ labelDescription: chr [1:2] "Tissue names" "Experience number" .. .. ..@ data :'data.frame': 64 obs. of 2 variables: .. ..$ Tissue: Factor w/ 32 levels "Adrenal_cortex",..: 23 23 5 5 1 1 16 16 32 32 ... .- attr(*, "names")= chr [1:64] "S_1" "S_2" "S_3" "S_4" ... .$ Exp : Factor w/ 2 levels "Exp1","Exp2": 1 2 1 2 1 2 1 2 1 2 ... .- attr(*, "names")= chr [1:64] "S_1" "S_2" "S_3" "S_4" ... .@ dimLabels : chr [1:2] "sampleNames" "sampleColumns" .. .. ..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slots .. ..@ .Data:List of 1 .. ..$: int [1:3] 1 1 0

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

getMatrixFromExpressionSet

Examples

```
data(expSetSpeCondExample)
```

expressionSpeCondExample

The expression matrix example used in the SpeCond package

Description

`expressionSpeCondExample` is expression value matrix (log2) used as an example for the SpeCond package. This is a subset of a new analysis of the Su et al, 2008 data. The columns are human tissues, the rows are probeset IDs.

Usage

```
data(expressionSpeCondExample)
```

Format

A matrix of 220 rows and 32 columns

Author(s)

Florence Cavalli, florence@ebi.ac.uk

Source

Su et al, PNAS, 2004, 'A gene atlas of the mouse and human protein-encoding transcriptomes'

Examples

```
data(expressionSpeCondExample)
```

fitNoPriorWithExclusion

Fit the expression values profile with a mixture of normal components

Description

`firPrior` performs a clustering of expression values for each gene profile using the mclust function ignoring the outliers (detected by the first step of the SpeCond prcedure) present in the SpecificOutlierStep1 argument . This results to a mixture of normal distribution components (from 1 to 3 components) fitting the expression values.

Usage

```
fitNoPriorWithExclusion(expressionMatrix, specificOutlierStep1 = FALSE,
param.detection = NULL, lambda = 1, beta = 0)
```

Arguments

expressionMatrix	the expression value matrix, genes*conditions
specificOutlierStep1	the list of outliers detected by the first step procedure, result of the getSpecificOutliersStep1 function or an attritube of the SpeCond result object. These outliers won't be taken into account for the mixture normal modelling performed by this function
param.detection	the matrix of parameters as obtained by getDefaultParameter or createParamterMatrix It must contain positive values for "lambda" and "beta". If NULL, the function getDefaultParameter will be used
lambda	positive value, it influences the choice of models by affecting the selection of one, two or three normal distributions, thus introducing some weight on the effect of number of parameters to be defined. The default is 1, the model uses the BIC value taking into account the log-likelihood value
beta	Should be equal to 0; prior is put on the variance determination of the normal distribution

Value

fit2	list of the gene as first attributes, for each gene a list of three attributes:
G	number of normal components fitting the data
NorMixParam	the parameters of each normal component: proportion, mean and standard deviation for the gene
classification	the normal component id in which the expression values of the gene are attributed

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

fitPrior, SpeCond

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the SpeCond analysis step by step
param.detection=getDefaultParameter()
param.detection

fit1=fitPrior(expressionSpeCondExample, param.detection=param.detection)

specificOutlierStep1=getSpecificOutliersStep1(expressionSpeCondExample,
 fit=fit1$fit1, param.detection, multitest.correction.method="BY",
prefix.file="run1_Step1", print.hist.pv=FALSE)

fit2=fitNoPriorWithExclusion(expressionSpeCondExample,
specificOutlierStep1=specificOutlierStep1,
```

```
param.detection=param.detection)

##then use getSpecificResult()
```

`fitPrior` *Fit the expression value profiles with a mixture of normal components*

Description

`firPrior` performs a clustering of expression values for each gene profile using the mclust function. This results to a mixture of normal distribution components (from 1 to 3 components) fitting the expression values.

Usage

```
fitPrior(expressionMatrix, param.detection = NULL, lambda = 1,
beta = 6, evaluation.lambda.beta = FALSE)
```

Arguments

`expressionMatrix`	the expression value matrix, genes*conditions
`param.detection`	the matrix of parameters as obtained by `getDefaultParameter` or `createParamterMatrix` It must contain positive values for "lambda" and "beta". If NULL, the function `getDefaultParameter` will be used
`lambda`	positive value, it influences the choice of models by affecting the selection of one, two or three normal distributions, thus introducing some weight on the effect of number of parameters to be defined. The default is 1, the model uses the BIC value taking into account the log-likelihood value
`beta`	positive value, it influences the prior applied during the determination of the variance of the normal distributions. It is important for this fitting since it allows the model to capture isolated outliers. The default value is 6
`evaluation.lambda.beta`	if TRUE, an extra attribute will be return indicating for how many gene the lambda and beta parameters change the number of normal component chosen to fit the expression values

Details

If `evaluation.lambda.beta` is TRUE an additional attribute `G.lambda.beta.effect` is returned. It is a matrix presenting the number of time the values of G (number of normal components for a particular gene) has changed between `lambda`=0 and the `lambda.1` value and between `beta`=0 and the `beta.1` value.

Value

`fit1`	list of the gene as first attributes, for each gene a list of three attributes:
`G`	number of normal components fitting the data
`NorMixParam`	the parameters of each normal component: proportion, mean and standard deviation for the gene

`classification`
: the normal component id in which the expression values of the gene are attributed

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`fitNoPriorWithExclusion`, `SpeCond`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the SpeCond analysis step by step
param.detection=getDefaultParameter()
param.detection

fit1=fitPrior(expressionSpeCondExample, param.detection=param.detection)

##then use getSpecificOutliersStep1(), fitNoPriorWithExclusion() and
## getSpecificResult()
```

---

getDefaultParameter

Get the default parameter to use SpeCond function

Description

This function returns the matrix param.detection default argument for the SpeCond function

Usage

```
getDefaultParameter()
```

Value

param.detection: a matrix of 2 row and 7 columns. The rows "Step1 "and "Step2" correspond respectively to the first and second set of parameters for the SpeCond function. The parameters (columns) are: lambda, beta, per, md, mlk, rsd. See the createParameterMatrix documentation for more details about the parameters.

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`createParameterMatrix`

Examples

```
param.detection=getDefaultParameter()
param.detection
```

`getFullHtmlSpeCondResult` *Visualisation function of the SpeCond analysis results*

Description

`getFullHtmlSpeCondResult` generates a full result html page.

Usage

```
getFullHtmlSpeCondResult(SpeCondResult=NULL, L.specific.result = NULL,
param.detection = NULL, page.name = "SpeCond_result",
page.title = "Condition-specific analysis results", prefix.file = NULL,
outdir="General_Result", sort.condition = "all",
gene.page.info=NULL, heatmap.profile = TRUE, heatmap.expression = FALSE,
heatmap.unique.profile = FALSE, expressionMatrix = NULL)
```

Arguments

`SpeCondResult`
: the `sp_list` class object result of the `SpeCond` functions

`L.specific.result`
: List of results present in the `sp_list` class specificResult object, see `SpeCond` or `getSpecificResult` functions

`param.detection`
: The parameter matrix used by the SpeCond detection procedure

`page.name`
: The name of the result html page. The default is "SpeCond_result"

`page.title`
: The title of the result html page. The default is "Condition-specific analysis results"

`prefix.file`
: a prefix added to the generated file(s) and the `outdir` directory name to linked them to the full result html page, by default is NULL, the `prefix.file` attribute of the `SpeCondResult` is used. It is useful to change the prefix when you create a new result page. As you may want to get results with different parameter sets and plots so using a different `SpeCondResult` or `L.specific.result` objects

`outdir`
: the name of the directory in which the generated files will be created. The default is "General_result"

`sort.condition`
: If the condition must sorted in the barplot presented the number of specific genes by condition. Can table the values: positive", "negative", "all": the conditions are sorted respectively by the number of specific genes detected as up-regulated, down-regulated or both

`gene.page.info`
: the result of the `getGeneHtmlPage` function. Enables the creation of links between this full result page and the single result pages created by the previous function. The default is "NULL"; no links are created

`heatmap.profile`
: TRUE/FALSE, to print or not a heatmap showing the specific profile of the genes. The default is FALSE

`heatmap.expression`
: TRUE/FALSE, to print or not a heatmap showing the expression of the genes. The default is FALSE

`heatmap.unique.profile`
: TRUE/FALSE, to print or not a heatmap showing the unique specific profile. The default is FALSE

`expressionMatrix`
: Must not be NULL if heatmap.expression=TRUE, must be the same as the input expression matrix. The default is NULL

Details

Either `SpeCondResult` or `L.specific.result` can be specified to use this function. If you use `L.specific.result` you ahve tp define `prefix.file`.

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`getGeneHtmlPage`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample, param.detection=NULL,
        multitest.correction.method="BY", prefix.file="E", print.hist.pv=TRUE,
                        fit1=NULL, fit2=NULL, specificOutlierStep1=NULL)

specificResult=generalResult$specificResult

##Produce the general html page results
getFullHtmlSpeCondResult(SpeCondResult=generalResult, param.detection=
  specificResult$param.detection, page.name="Example_SpeCond_results",
  page.title="Tissue specific results", sort.condition="all", heatmap.profile=TRUE,
  heatmap.expression=FALSE, heatmap.unique.profile=FALSE,
  expressionMatrix=expressionSpeCondExample)

##Produce the Gene html page results for the first 20 genes using the specificResult obje
## these pages to the table result in the general html page
specificResult=generalResult$specificResult
genePageInfo=getGeneHtmlPage(expressionSpeCondExample, specificResult, name.index.html=
  "index_example_SpeCond_Results.html",outdir="Single_result_pages_example",
   gene.html.ids=c(1:20))
```

```
##Produce the general html page results
getFullHtmlSpeCondResult(L.specific.result=specificResult$L.specific.result,
  param.detection=specificResult$param.detection, page.name="Example_SpeCond_results2",
  page.title="Tissue specific results", prefix.file="S2", sort.condition="all",
  heatmap.profile=TRUE, heatmap.expression=FALSE, heatmap.unique.profile
  =FALSE, expressionMatrix=Mexp, gene.page.info=genePageInfo)
```

`getGeneHtmlPage` *Visualise for each gene the condition-specific detection result from*

Description

`getGeneHtmlPage` generates html results pages for a set of genes as well as an index page. The index allows to navigate between the gene result pages.

Usage

```
getGeneHtmlPage(expressionMatrix, specificResult,
name.index.html = "index.html", prefix.file = NULL,
outdir="Single_result_pages", gene.html = NULL,
gene.html.ids = c(1:10))
```

Arguments

`expressionMatrix`	the matrix of expression values initially used
`specificResult`	the `sp_list` class object result of the `getSpecificProbeset` function
`name.index.html`	the name of the html index, by default is index.html
`prefix.file`	a prefix added to the generated file(s) and `outdir` directory name to linked to the index file. The default is NULL, the `prefix.file` attribute of `specificResult` is used
`outdir`	the name of the directory in which the generated files will be created. The default is "Single_result_pages"
`gene.html`	a vector of gene names for which you want to create html pages, same as the row names of the expressionMatrix object. The default is NULL (the values of the *gene.html.ids* argument will be used)
`gene.html.ids`	a vector of integer corresponding to the row numbers in the expressionMatrix object of the genes for which you want to create html pages. The default is the 10 first rows (or the number of row of the expressionMatrix if inferior to 10)

Details

The main file `name.index.html` is created in the current directory. The result page(s) to which it points are created in the `outdir` directory. If both `gene.html` and `gene.html.ids` are set to NULL, the gene html pages for every gene in the expressionMatrix object will be generated It is useful to change the prefix when you create a new index as well as changing the `name.index.html` value. As you may want to get index with the same genes but different parameters set and plots so using a different specificResult object. It is possible to use `gene.html` or `gene.html.ids` to select a list of gene.

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`getFullHtmlSpeCondResult`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample, param.detection=NULL,
 multitest.correction.method="BY", prefix.file="E", print.hist.pv=TRUE, fit1=NULL,
 fit2=NULL, specificOutlierStep1=NULL)
specificResult=generalResult$specificResult
##Produce the Gene html page results for the first 20 genes using the specificResult
##object
genePageInfo=getGeneHtmlPage(expressionSpeCondExample, specificResult,
 name.index.html="index_example_SpeCond_Results.html", outdir=
 "Single_result_pages_dir", gene.html.ids=c(1:20))
```

`getMatrixFromExpressionSet` *Obtain the expression matrix from an ExpressionSet object*

Description

`getMatrixFromExpressionSet` method returns an matrix of expression values from an ExpressionSet object. It takes into consideration the need of summarizing the samples values by conditions to perform the SpeCond analysis

Usage

```
getMatrixFromExpressionSet(expSet, condition.factor = NULL,
                        condition.method = c("mean", "median","max"))
```

Arguments

`expSet`	an ExpressionSet object
`condition.factor`	a factor object of length equal to the number of columns (samples) of the ExpressionSet object specifying which sample(s) belong to which condition (condition.factor levels); can be extracted from the phenoData
`condition.method`	the method (mean, median or max) to summarise the samples by conditions (defined by the condition.factor vector)

Details

For each level of the condition.factor, the expression values of the ExpressionSet object are computed using the condition.method method. If there is only one sample for a condition the expression value is not changed if condition.factor is NULL, the expression matrix of the ExpressionSet object will simply be extracted using exprs()

Value

A matrix of expression values of size (number of row in the ExpressionSet * number of level of the condition.factor)

Author(s)

Florence Cavalli, florence@ebi.ac.uk

References

Biobase

See Also

`SpeCond`

Examples

```
library(SpeCond)
data(expSetSpeCondExample)
expSetSpeCondExample
f_Tissues=factor(paste("Tissue_",rep(1:32,each=2),sep=""))
f_Tissues
Mexp=getMatrixFromExpressionSet(expSetSpeCondExample,
  condition.factor=f_Tissues,condition.method="mean")
## or
Mexp=getMatrixFromExpressionSet(expSetSpeCondExample,
  condition.factor=expSetSpeCondExample$Tissue,condition.method="mean")
```

`getProfile` *Create the condition-specific profile of specific matrix result from*

Description

`getProfile` converts a matrix of 0,1,-1 values in a matrix of one columns. Each row is transformed to a character chain of the values separated by comma.

Usage

```
getProfile(M.specific)
```

Arguments

`M.specific`	Is a matrix result present in the SpeCond object result: generalResult\$specificResult\$L.specific.resul

Value

`M.specific.profile`
: a matrix of number of row as the M.specific matrix x 2 columns. The first column "profile" is the profile: character chain of the values in M.specific separated by commas. The second column of the 2 columns: "sum.row" is the number of condition in which the genes is specific (up or down regulated)

`M.specific.profile.unique`
: a matrix of number of unique profile * number of conditions. The columns order is the same as M.specific

`M.specific.profile.table`
: a matrix of number of unique profile *2. The columns are: profile, nb.gene. The first column is the profile: character chain of the unique rows in M.specific separated by commas. The second column is the number of genes (rows) from M.specific which have this profile

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`SpeCond`, `writeSpeCondResult`, `writeUniqueProfileSpecifcResult`, `writeGeneResult`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
dim(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample, param.detection=NULL,
  multitest.correction.method="BY", prefix.file="E", print.hist.pv=TRUE, fit1=NULL,
  fit2=NULL, specificOutlierStep1=NULL)

##get the profiles for each gene
L.specific.result.profile=getProfile(generalResult$specificResult$L.specific.result
                                                          $M.specific)
##or
specificResult=generalResult$specificResult
L.specific.result.profile=getProfile(specificResult$L.specific.result$M.specific)
```

`getSpecificOutliersStep1` *Detect the condition-specific as outliers in for the first step on the*

Description

Perform the first detection step of the SpeCond procedure. Use the fitting of the gene expression value with a mixture of normal distribution results and a set of rules to detect the outliers. It returns the outliers detected as specifically expressed for each gene.

Usage

```
getSpecificOutliersStep1(expressionMatrix, fit1 = NULL,
param.detection = NULL, multitest.correction.method = "BY",
prefix.file = NULL, print.hist.pv = FALSE)
```

Arguments

expressionMatrix	the gene expression matrix (genes * conditions)
fit1	the result of `fitPrior` containing the parameter of the mixture normal model of the expression data
param.detection	the parameter for the detection, a vector with the names ("per","md","mlk","rsd","pv") or the first row of the matrix obtained by `getDefaultParameter` or `createParameterMatr`
multitest.correction.method	the multitest correction method. The default is "BY", for the possible values see `p.adjust`
prefix.file	a prefix added to the generated file. The default is NULL but has to be set. It is useful to change the prefix when you perform a new analysis. As you may want to compare the results with different parameters set.
print.hist.pv	to print in a pdf file the (non-adjusted) p-value histogram

Details

Frist essential method to obtain the matrix of expression value from your ExpressionSet to apply the SpeCond procedure step by step using the following function `fitPrior`, `fitNoPriorwithExclusion`, `getSpecificOutliersStep1`, `getSpecificResult`. The returned matrix will be the expressionMatrix argument of the above function

Value

A list of size the number of rows (genes) in the expressionMatrix. If the gene has outlier expression, the column number of this outlier is stored, NULL if not.

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`fitPrior`, `SpeCond`, `getSpecificResult`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the SpeCond analysis step by step
param.detection=getDefaultParameter()
param.detection

fit1=fitPrior(expressionSpeCondExample, param.detection=param.detection)
```

```
specificOutlierStep1=getSpecificOutliersStep1(expressionSpeCondExample,
 fit=fit1$fit1, param.detection, multitest.correction.method="BY",
 prefix.file="run1_Step1", print.hist.pv=FALSE)

##then use fitNoPriorWithExclusion() and getSpecificResult()
```

`getSpecificResult` *Detect the condition-specific genes for the second step on the SpeCond*

Description

Perform the second detection step of the SpeCond procedure. Use the second fitting (without prior and ignoring the outliers detected in the first step) of the gene expression value with a mixture of normal distribution results and a set of rules to detect the outliers. It returns the outliers detected as specifically expressed for each gene.

Usage

```
getSpecificResult(expressionMatrix, fit2 = NULL, param.detection = NULL,
specificOutlierStep1 = NULL, multitest.correction.method = "BY",
prefix.file = NULL, print.hist.pv = FALSE)
```

Arguments

`expressionMatrix`
: the gene expression matrix (genes * conditions)

`fit2`
: The result of `fitNoPriorWithExclusion` containing the parameter of the mixture normal model of the expression data ignoring the outliers detected in the first step of the procedure

`param.detection`
: the parameter for the detection, a vector with the names ("per","md","mlk","rsd","pv") or the second row of the matrix obtained by `getDefaultParameter` or `createParameterMatrix`

`specificOutlierStep1`
: the list of outliers detected by the first step procedure, result of the `getSpecificOtuliersStep1` function

`multitest.correction.method`
: the multitest correction method. The default is "BY", for the possible values see `p.adjust`

`prefix.file`
: a prefix added to the generated file. The default is NULL but as to be set. It is useful to change the prefix when you perform a new analysis. As you may want to compare the results with different parameters set

`print.hist.pv`
: a logical (TRUE/FALSE) whether to print in a pdf file the (non-adjusted) p-value histogram; the default is FALSE

Value

An object of class `sp_list`

`prefix.file` the prefix used for this analysis. It will be used by default in the function `getGeneHtmlPage`

`fit` the fitting parameters used by the detection i.e. the argument fit2

`param.detection` the parameters used for the detection i.e. the argument parm.detection

`L.specific.result` Full detection results (It will be used by the `getFullHtmlSpeCondResult`). This list contains 7 attributes:

`M.specific.all` matrix of 0: not selective, 1: selective up-regulated, -1: selective down-regulated; same dimensions as the input expression values matrix

`M.specific` same as M.specific.all but reduced to the specific genes. NULL if no gene has been detected as specific

`M.specific.sum.row` Number of conditions in which the gene is specific

`M.specific.sum.column` Number of specific genes by conditions

`L.pv` list of all genes with a matrix of conditions and the corresponding p-values (if the gene is specific)

`specific` vector of size the number of genes with "Not specific" or "Specific" according to the specificity of the gene

`L.condition.specific.id` list of the specific genes with a vector of column numbers (condition ids), for which the gene is specific

`L.null` a list of vectors of 1 and 0 representing the null distribution. The length of the vector for each gene corresponds to the number of normal distributions fitting the gene expression value. The list is sorted as the gene order in the input expression matrix

`L.mlk` a list of vectors containing the min log-likelihood computed between normal distribution components. NULL if the mixture model of the gene is composed of only one component or if the proportion of all components is superior to the per.2 parameter

`L.rsd` a list of vectors containing the standard deviation ratio computed between normal distribution components. NULL if the mixture model of the gene is composed of only one component

`identic.row.ids` row number(s) from the initial input matrix which contain identical values for all conditions. These rows are not considered in the analysis

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`fitNoPriorwithExclusion`, `SpeCond`, `getSpecificResult`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the SpeCond analysis step by step
param.detection=getDefaultParameter()
param.detection

fit1=fitPrior(expressionSpeCondExample, param.detection=param.detection)

specificOutlierStep1=getSpecificOutliersStep1(expressionSpeCondExample,
 fit=fit1$fit1, param.detection, multitest.correction.method="BY",
 prefix.file="run1_Step1", print.hist.pv=FALSE)

fit2=fitNoPriorWithExclusion(expressionSpeCondExample,
       specificOutlierStep1=specificOutlierStep1, param.detection=param.detection)

specificResult=getSpecificResult(expressionSpeCondExample, fit=fit2,
  specificOutlierStep1=specificOutlierStep1, param.detection,
  multitest.correction.method="BY", prefix.file="run1_Step2",
  print.hist.pv=FALSE)
```

---

simulatedSpeCondData

An example of simulated expression matrix used in the SpeCond package

---

Description

simulatedSpeCondData is a expression value matrix used as an example for the SpeCond package. The expression values were randomly generated from three different normal distributions.

Usage

```
data(simulatedSpeCondData)
```

Format

A matrix of 600 rows and 30 columns

Details

The default expression values for each probeset is randomly generated from a normal distribution of mean=7 and sd=0.6. The probesets 1 to 100 have specific expression values for the conditions 10, 20 and 30 coming from a normal distribution of mean=11, sd=0.5. The probesets 200 to 300 have specific expression values for the conditions 9, 18 and 27 coming from a normal distribution of mean=13, sd=0.4. This data set is used to show the ipmportance and the effect of the paramters in the SpeCond detection. See the SpCond vignette for more detailsy

Examples

```
data(simulatedSpeCondData)
```

`writeGeneResult` *Write a condition-specific analysis result text file*

Description

`writeGeneResult` produces a text file containing the list of gene, if they have been detected as tissue-specific or not (S/N), for how many tissues in total, how many tissue as up-regulated, how many tissue as down-regulated, in which tissues for up-regulated and down-regulated.

Usage

```
writeGeneResult(L.specific.result, file.name.result.gene =
 "gene_summary_result.txt", gene.names = NULL)
```

Arguments

`L.specific.result`	the `L.specific.result` list of the included in the result of the main `SpeCond` function: generalResult$specificResult$L.specific.result
`file.name.result.gene`	the name of the produced file containing the list of specific genes an thier specific detection
`gene.names`	vector of gene's names to select a suset of genes. The default is NULL, all genes from the input matrix in `SpeCond` function are used

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`SpeCond`,`getProfile`,`writeSpeCondResult`,`writeUniqueProfileSpecificResult`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample,
  param.detection=NULL, multitest.correction.method="BY", prefix.file="E",
  print.hist.pv=TRUE, fit1=NULL, fit2=NULL, specificOutlierStep1=NULL)
specificResult=generalResult$specificResult

##write the result file
writeGeneResult(specificResult$L.specific.result, file.name.result.gene=
  "Example_gene_summary_result.txt", gene.names=
   rownames(expressionSpeCondExample)[1:10])
```

writeSpeCondResult *Write in text files the main result of the SpeCond function*

Description

writeSpeCondResult produces three text files: - The table of the gene detected as specific and in which condition they are specific (0: no specific, 1: specific up-regulated, -1:specific down-regulated). The default name is file.name.profile="specific_profile.txt". - The list of the specific genes. The default name is: "list_specific_probeset.txt". - The table of the unique specific profiles detected. The default name is: "specific_unique_profile.txt".

Usage

```
writeSpeCondResult(L.specific.result, file.name.profile =
 "specific_profile.txt", file.specific.gene = "list_specific_gene.txt",
 file.name.unique.profile = "specific_unique_profile.txt")
```

Arguments

L.specific.result
: The L.specific.result list of the included in the result of the main SpeCond function: generalResult$specificResult$L.specific.result

file.name.profile
: The name of the produced file containing the gene's profiles

file.specific.gene
: The name of the produced file containing the list of the specific genes

file.name.unique.profile
: The name of the produced file containing the unique gene's profiles

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

SpeCond, getProfile, writeUniqueProfileSpecifcResult, writeGeneResult

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample, param.detection=NULL,
 multitest.correction.method="BY", prefix.file="E", print.hist.pv=TRUE, fit1=NULL,
 fit2=NULL, specificOutlierStep1=NULL)
 specificResult=generalResult$specificResult

##write the SpeCond results files
 writeSpeCondResult(specificResult$L.specific.result,file.name.profile=
 "Example_specific_profile.txt", file.specific.gene="Example_list_specific_gene.txt",
  file.name.unique.profile="Example_specific_unique_profile.txt")
```

`writeUniqueProfileSpecificResult` *Write the specific profiles from the SpeCond analysis*

Description

Produces a text file with the unique specific profiles among the conditions detected by the SpeCond analysis.

Usage

```
writeUniqueProfileSpecificResult(L.specific.result, file.name.unique.profile =
 "specific.unique_profile.txt", full.list.gene = FALSE)
```

Arguments

`L.specific.result`	the `L.specific.result` list of the included in the result of the main `SpeCond` function: generalResult$specificResult$L.specific.result
`file.name.unique.profile`	the name of the produced file containing the gene's profiles
`full.list.gene`	If TRUE, the last column correspond to the gene's names which have the profile described in the row

Author(s)

Florence Cavalli, florence@ebi.ac.uk

See Also

`SpeCond`, `getProfile`, `writeSpeCondResult`, `writeGeneResult`

Examples

```
library(SpeCond)
data(expressionSpeCondExample)
##Perform the condition specific detection analysis with SpeCond()
generalResult=SpeCond(expressionSpeCondExample,
  param.detection=NULL, multitest.correction.method="BY", prefix.file="E",
  print.hist.pv=TRUE, fit1=NULL, fit2=NULL, specificOutlierStep1=NULL)
specificResult=generalResult$specificResult

##write the result file
writeUniqueProfileSpecificResult(L.specific.result=specificResult$L.specific.result,
  file.name.unique.profile="Example_specific_unique_profile.txt", full.list.gene=FALSE)
```

Index