

Package ‘MADAM’

February 14, 2012

Type Package

Title This package provides some basic methods for meta-analysis

Version 1.1

Date 2010-08-16

Author Karl Kugler, Laurin Mueller, Matthias Wieser

Maintainer Karl Kugler <karl@eigenlab.net>

Description In this package we provide some basic utilities for performing meta-analysis. We included methods for dealing with missing values, or measurements with a variance of zero. For developing or testing meta-analysis methods a mock up data model is integrated. Additionally did we implement a method for performing significance based methods proposed by Fisher and Rhodes. For visualizing the results three methods were added. Some functions were taken from the GeneMeta and RankProd packages in order to implement parallel versions of the existing methods. Please refer to the original authors if you make use of these implementations.

License LGPL

URL <http://madam.r-forge.r-project.org>

LazyLoad yes

Depends R (>= 2.10), Biobase (>= 2.4), qvalue, RankProd, RColorBrewer, GeneMeta, gplots, gene-filter, impute

Imports RColorBrewer, GeneMeta, gplots, genefilter, impute

Suggests snow

Repository CRAN

Date/Publication 2011-04-11 15:24:24

R topics documented:

MADAM-package	2
calculateRankProduct	3
calculateRankSum	4
corrNA	5
corrNAGroupwise	7
corrVar	8
default.zero.subst	9
doEnsemble	10
doES	11
doRhodesFDR	12
doRP	13
fisherMethod	14
fishersum	15
generateRandomMADData	16
multiTtest	17
plotFDR	18
plotHeatMap	19
plotMAVolcano	20
zScoreFDRClust	21
zScoresClust	23
Index	26

MADAM-package

*MADAM - Meta-Analytical Data Aggregation Methods Toolbox***Description**

The MADAM package was developed in order to help reseachers and scientist working with meta-analysis. MADAM provides a set of method for generating test data, performing meta-analysis and visualizing the results.

Some functions were taken from the GeneMeta and RankProd packages in order to implement parallel versions of the existing methods. Please refer to the original authors if you make use of these implementations.

Details

Package:	MADAM
Type:	Package
Version:	1.1
Date:	2010-06-29
License:	LGPL
LazyLoad:	yes
URL:	http://madam.r-forge.r-project.org

Author(s)

Karl Kugler, Matthias Wieser Maintainer: Karl Kugler <karl@eigenlab.net>

calculateRankProduct *Calculate the rank product and asses the significance*

Description

With this function the rank product for each feature (row) in data is calculated first. Then, the significance is calculated by permutating the ranks B times and comparing the derived random rank products with the original one. Since this might turn out to be computationally demanding, a snow cluster object has to be provided.

Usage

```
calculateRankProduct(data, B = 1000, cluster = NULL)
```

Arguments

data	A matrix containing rank information from several studies. Features are stored in rows, and the single studies in columns.
B	The number of permutations used to assess the significance of the rank product.
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

A data.frame object with 4 columns, where the rownames are the rownames from the input matrix data.

RP	The rank product
p.value	The significances of RP
q.value	The p-values corrected for the FDR
rank	The rank of a feature, derived by ranking the q.values. Ties are ranked randomly.

Note

This function is based on the implementation of the RankProd package Version 2.20.0 by Hong F et al. Any modification made by MADAM developers are only for enabling cluster computing.

Author(s)

Karl Kugler, karl@eigenlab.net

References

RankProd: a bioconductor package for detecting differentially expressed genes in meta-analysis. Hong F, Breitling R, McEntee CW, Wittner BS, Nemhauser JL, Chory J. *Bioinformatics*. 2006 Nov 15;22(22):2825-7. Epub 2006 Sep 18.

See Also

[calculateRankSum](#)

Examples

```
set.seed(123)
M <- sapply(1:5, sample, x=1:10, size=10)
res <- calculateRankProduct(data=M, B = 100, cluster = NULL)
res
```

calculateRankSum	<i>Calculate the rank sum and assess the significance</i>
------------------	---

Description

With this function the rank sum for each feature (row) in data is calculated first. Then, the significance is calculated by permutating the ranks B times and comparing the derived random rank products with the original one (similar to the rank product). Since this might turn out to be computationally demanding, a snow cluster object has to be provided.

The basic idea is similar to the one of the rank product.

Usage

```
calculateRankSum(data, B = 10000, cluster = NULL)
```

Arguments

data	A matrix containing rank information from several studies. Features are stored in the rows, and the single studies in the columns.
B	The number of permutations used to assess the significance of the rank sum.
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

A data.frame object with 4 columns, where the rownames are the rownames from the input matrix data.

RS	The rank sum
p.value	The significances of RS
q.value	The p-values corrected for the FDR
rank	The rank of a feature, derived by ranking the q.values. Ties are ranked randomly.

Author(s)

Karl Kugler, karl@eigenlab.net

References

RankProd: a bioconductor package for detecting differentially expressed genes in meta-analysis. Hong F, Breitling R, McEntee CW, Wittner BS, Nemhauser JL, Chory J. *Bioinformatics*. 2006 Nov 15;22(22):2825-7. Epub 2006 Sep 18.

See Also

[calculateRankProduct](#)

Examples

```
set.seed(123)
M <- sapply(1:5, sample, x=1:10, size=10)
res <- calculateRankSum(data=M, B = 100, cluster = NULL)
res
```

 corrNA

Functions to impute missing values

Description

Missing values might cause problems in the analysis pipeline. Within MADAM a simple imputation method was integrated. This function bases on calculating mean values and sampling variance robust values around it.

A more sophisticated method using k nearest neighbors as provided by the **impute** package, which is wrapped to by this function as well.

Usage

```
corrNA(es, method = "madam", cl, cl.val = c(0, 1), conservative = TRUE,
na.thres = 0.5, na.abs.thres = 2, exclude = TRUE)
```

Arguments

es	A list of ExpressionSet objects, all sharing the same feature names.
method	Selection of the used imputation method. Current options are "madam" and "knn". If using "madam" the MADAM based method is used, by selecting "knn" the knn method from the impute package is used.
cl	A list of factor variables, labelling the class for each of the samples in es.
cl.val	Level values of cl (default: 0,1)).
conservative	~ A conservative substitution at least a percentage of na.thres features needs to be present in each group to impute missing values with the

na.thres	The percentage threshold of values that have at least to be present in order to impute missing values using the MADAM method.
na.abs.thres	An absolute number of entries needed to impute missing values, if mean should not be calculated the conservative (percentage) way.
exclude	If set to 'TRUE', features still containing missing values after imputation are removed from all studies.

Value

The method returns the list of ExpressionSets that was provided, but now with missing values inserted and failed features excluded, if this option was set.

Author(s)

Karl Kugler, karl@eigenlab.net

References

Hastie, T., Tibshirani, R., Sherlock, G., Eisen, M., Brown, P. and Botstein, D., Imputing Missing Data for Gene Expression Arrays, Stanford University Statistics Department Technical report (1999), <http://www-stat.stanford.edu/~hastie/Papers/missing.pdf>

Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein and Russ B. Altman, Missing value estimation methods for DNA microarrays BIOINFORMATICS Vol. 17 no. 6, 2001 Pages 520-525

See Also

[corrNA](#)

Examples

```
set.seed(666)
A <- generateRandomMADData(g = 20, perc.sig = 0.1, i = 3)
c1 <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
#generate some bad measurements
exprs(A[[1]])[1,1] <- NA
exprs(A[[1]])[4,2:4] <- NA
exprs(A[[2]])[3,4:6] <- NA
sapply(A, function(a){sum(is.na(exprs(a)))})
A <- corrNA(A, method="madam", c1)
sapply(A, function(a){sum(is.na(exprs(a)))})
```

corrNAGroupwise	<i>Function to filter features with missing values in a groupwise manner</i>
-----------------	--

Description

Filtering for features with too many missing values is an important task. If feature information is not uniformly distributed between the experimental groups it may lead to problems with later analysis. An overall threshold for feature information may be reached for the whole study but not within each group.

This method checks for a certain ratio of numbers to present in every experimental group. By default bad features are filtered out (`drop=TRUE`), optionally the information about which features are bad can be returned.

Usage

```
corrNAGroupwise(es = NULL, cl = NA, Th = 0.5, drop=TRUE)
```

Arguments

<code>es</code>	A single ExpressionSet object or a list of ExpressionSet objects
<code>cl</code>	In case <code>es</code> is a single ExpressionSet object a factor with as many entries as <code>es</code> has samples, or in case <code>es</code> is a list, a list of the same length containing factors
<code>Th</code>	A percentage of at least <code>Th</code> measurements has to be present in each feature and group for a feature to be valid
<code>drop</code>	By default bad features will be removed from the corresponding ExpressionSet, if <code>drop</code> is set to <code>FALSE</code> they will be kept and a list indicating which features are bad will be returned instead.

Value

By default a list with ExpressionSet object will be returned. They are identical to the input `es`, but bad features are removed. If `drop` is set to `FALSE` a list will be returned providing bad feature information for each ExpressionSet object. This means that the indices of those features that do not fulfill the quality criterion are being listed.

Author(s)

Karl Kugler, karl@eigenlab.net

See Also

[corrVar](#), [corrNA](#)

Examples

```

set.seed(666)
es <- generateRandomMADData(g=10, i=3)

#generate some missingness
exprs(es[[1]])[1,1:5] <- NA

#prepare group information
es.g <- lapply(es, function(x){x$group})

#standard call with list
corrNAGroupwise(es=es, cl=es.g)[[1]]

#standard call with single object
corrNAGroupwise(es=es[[1]], cl=es.g[[1]])[[1]]

#do not drop bad features
corrNAGroupwise(es=es[[1]], cl=es.g[[1]], drop=FALSE)

```

corrVar

Function to replace variances of zero

Description

In very rare cases measurements may show a variance of zero for some features. This might be induced by either technical or data manipulation means. Since such non-variance may cause problems in the later analysis, we integrated a method for dealing with this. First, we calculate the variance for all features within one group, and then use the mean m of it for adding some noise n to the features with zero variance: $n \sim N(0,m)$ for each group separately.

Usage

```

corrVar(es, cl, cl.val = c(0, 1), mean = 0, sd = 0,
        na.rm = TRUE)

```

Arguments

es	A list of ExpressionSet objects, all sharing the same feature names.
cl	A list of factor variables, labeling the class for each of the samples in es.
cl.val	Level values of cl (default: 0,1)..
mean	The mean used for N (default is set to zero)
sd	The standard deviation of n. It will be either calculated for each group, when setting it to zero (default), or can be given to the function.
na.rm	If set to 'TRUE' (default), missing values are excluded from calculating the variances.

Value

The method returns the list of ExpressionSets that was provided, but now with variances of zero replaced by the new ones.

Author(s)

Karl Kugler, karl@eigenlab.net

See Also

[corrVar](#)

Examples

```
set.seed(666)
A <- generateRandomMADData(g = 20, perc.sig = 0.1, i = 3)
c1 <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
#generate some variance of 0
exprs(A[[1]])[1,] <- 5
min(rowSds(exprs(A[[1]])))
es <- corrVar(A, c1)
min(rowSds(exprs(es[[1]])))
```

default.zero.subst *Default zero significance replacement*

Description

Since for Fisher's formula, significances of zero would cause a problem ($\ln(0) == \text{Inf}$), all such p-values are replaced with this number.

Usage

```
default.zero.subst
```

Format

The format is: num 1e-06

Author(s)

Karl Kugler, karl@eigenlab.net

doEnsemble

*Method to perform an ensemble approach for meta-analysis.***Description**

This function allows performing a complete meta-analysis for a set of studies. Including an effect size, a significance and a ranking approach. Additionally an exploratory ensemble approach may be performed. The user may choose to plot some figures and report the complete results into a directory.

Usage

```
doEnsemble(A, cl, cl.val = c(0, 1), feature.names = NULL, do.FM = TRUE,
perm.ES = 1000, perm.RP = 1000, perm.ENS.RP = 1000, perm.ENS.RS = 1000,
do.ENS.FM = TRUE, cluster = NULL, useREM = TRUE, results.dir = getwd(),
write.all = FALSE, plot.fdr = FALSE, plot.ranks = FALSE)
```

Arguments

A	A list of ExpressionSet objects, all sharing the same feature names.
cl	A list of factor variables, labeling the class for each of the samples in A.
cl.val	Level values of cl (default: 0,1)).
feature.names	names of features common to all sets. If NULL, featureNames(A[[1]]) is used.
do.FM	Doing meta analysis based on significance (Fisher method).
perm.ES	Number of permutations for calculating FDR for ES Choi.
perm.RP	Number of permutations for rank product approach.
perm.ENS.RP	Number of permutations for Ensemble rank product approach.
perm.ENS.RS	Number of permutations for Ensemble rank sum approach.
do.ENS.FM	Doing Ensemble Fisher-Method.
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.
useREM	boolean parameter indicating whether to use ES REM model (default is TRUE)
results.dir	Directory where results are written as csv. The script will create a subfolde with date and time, where everything will be stored.
write.all	If TRUE all results are written as a csv in a subfolder of results.dir.
plot.fdr	Plot FDR figures for MA & Ensemble methods.
plot.ranks	Plot heatmaps for single MA methods.

Value

The method returns the list of objects with all the results for both up- and downregulation. Wording for upregulation in the second group is g2up, and for downregulation in the second group is referred to as g2down. Additionally the target directory for all resulting output files is reported.

Note

When performing an ensemble approach with meta-analysis data coming from the same set of studies, the result has to be considered to be biased towards the alternative hypothesis due to correlations within the data. The result has then to be considered for exploratory use only!

Author(s)

Karl Kugler, karl@eigenlab.net

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
c1 <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
res <- doEnsemble(A, c1, feature.names=NULL, do.FM= TRUE,
  perm.ES=10, perm.RP=10, perm.ENS.RP= 10, perm.ENS.RS= 10,
  do.ENS.FM= TRUE, cluster= NULL, useREM= TRUE,
  results.dir=getwd(), write.all=FALSE, plot.fdr= FALSE, plot.ranks=FALSE)
names(res)
```

doES

*Method to wrap parameters for Effect Size Meta Analysis.***Description**

Function hiding call of ES method by Choi implemented in GeneMeta package.

Usage

```
doES(A, c1, c1.val = c(0, 1), useREM = TRUE, nperm = 5000, cluster = NULL)
```

Arguments

A	List o ExpressionSets.
c1	List classes to be tested for first class. (should be in the first sample of the first study)
c1.val	Level values of c1 (default: 0,1)).
useREM	Boolean indicating whether to use REM ES model.
nperm	Number of permutations to calculate FDR (default is 5000).
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

list with information on up- and down regulation of features

Note

This function is based on the implementation of the GeneMeta package Version 1.20.0 by Choi et al. Any modification made by MADAM developers are only for enabling cluster computing.

Note

Since the current implementation of MADAM only wraps to the method within the **GeneMeta** package, it cannot be run in a parallel manner.

Author(s)

Karl Kugler, karl@eigenlab.net

References

Choi et al, Combining multiple microarray studies and modeling interstudy variation. *Bioinformatics*, 2003, i84-i90.

See Also

[zScoreFDR](#)

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
res <- doES(A, cl, useREM = TRUE, nperm = 10, cluster = NULL)
res
```

doRhodesFDR

Function for performing classical Rhodes MA method by deriving significance from p-values and permutating labels.

Description

Perform method as proposed by Rhodes.

Usage

```
doRhodesFDR(data, B = 10000, zero.subst = default.zero.subst, cluster = NULL)
```

Arguments

data	Data frame or matrix containing p-values from various methods.
B	Number of permutations.
zero.subst	Value to replace zeros with.
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

A data.frame containing significances and ranks

Author(s)

Karl Kugler, karl@eigenlab.net

References

Rhodes DR, Barrette TR, Rubin MA, Ghosh D, Chinnaiyan AM. Meta-analysis of microarrays: interstudy validation of gene expression profiles reveals pathway dysregulation in prostate cancer. *Cancer Res.* 2002 Aug 1;62(15):4427-33.

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
pval <- multiTtest(A, cl, alternative= "two.sided")
rownames(pval) <- rownames(exprs(A[[1]]))
res <- doRhodesFDR(data=pval, B = 10, zero.subst = default.zero.subst, cluster = NULL)
res
```

doRP

Function for calling RankProduct meta-analysis method by Breitling.

Description

Class to wrap parameters for Rank Product Meta-Analysis by Breitling.

Usage

```
doRP(A, cl, cl.val = c(0, 1), nperm = 1000, gene.names = NULL, cluster = NULL)
```

Arguments

A	List of ExpressionSets.
cl	List classes to be tested for first class.
cl.val	Level values of cl (default: 0,1). First sample of each study has to be of first type.
nperm	Number of permutations to calculate FDR (default is 1000)
gene.names	If not provided within the list of ExpressionSets, feature names can be set here.
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

List of significances and ranks for both up- and downregulation.

Note

Since the current implementation of MADAM only wraps to the method within the **RankProd** package, it cannot be run in a parallel manner.

Author(s)

Karl Kugler, karl@eigenlab.net

References

Breitling, R., Armengaud, P., Amtmann, A., and Herzyk, P. Rank Products: A simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments, *_FEBS Letter_*, 57383-92, 2004

See Also

[calculateRankProduct](#), [calculateRankSum](#), [RPadvance](#)

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
res <- doRP(A, cl=cl, nperm = 10, cluster = NULL)
res
```

fisherMethod

Perform classical Fisher Method for combining p-values in a parallel manner.

Description

Function for performing classical Fisher method by deriving significance from chi square distribution.

Usage

```
fisherMethod(data, zero.subst = default.zero.subst, cluster = NULL)
```

Arguments

<code>data</code>	Data.frame with features in rows and studies in columns.
<code>zero.subst</code>	Since p-values of 0 cause problems, these are substituted by a very small number (default is <code>default.zero.subst</code>).
<code>cluster</code>	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

list of significances and ranks

Author(s)

Karl Kugler, karl@eigenlab.net

References

Fisher RA Statistical Methods for Research Workers. Oliver & Boyd, Edinburgh 1925.

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
pval <- multiTtest(A, cl, alternative= "two.sided")
rownames(pval) <- rownames(exprs(A[[1]]))
res <- fisherMethod(data=pval)
res
```

fishersum

Function to calculate Fisher's sum.

Description

This function calculates the sum, that is used for Fisher's method of combining p-values.

Usage

```
fishersum(P)
```

Arguments

`P` Vector of p-values.

Value

Returns the sum of $-2 \cdot \log$ of the vector of p-values.

Author(s)

Karl Kugler, karl@eigenlab.net

References

Fisher RA Statistical Methods for Research Workers. Oliver & Boyd, Edinburgh 1925.

Examples

```
set.seed(123)
P1 <- c(1,1,1,1)
P2 <- c(0.05, 0.05, 0.05, 0.05)
S1 <- fishersum(P1)
S2 <- fishersum(P2)
1-pchisq(S1, df=4*2)
1-pchisq(S2, df=4*2)
```

generateRandomMADData *Calculate the rank product and asses the significance*

Description

With this function the rank product for each feature (row) in data is calculated first. Then, the significance is calculated by permutating the ranks B times and comparing the derived random rank products with the original one. Since this might turn out to be computationally demanding, a snow cluster object has to be provided.

Usage

```
generateRandomMADData(g = 5000, perc.sig = 0.1, i = 2,
k = 0, err.span = c(0.1, 0.2))
```

Arguments

<code>g</code>	Number of features per study.
<code>perc.sig</code>	Percentage of overall significant features.
<code>i</code>	Number of studies.
<code>k</code>	Number of samples for each of the two groups/study. If set to 0, numbers are sampled randomly
<code>err.span</code>	Span for var in error.

Details

In order to validate our ensemble approach, two data models were merged and adapted for our purposes. This model driven approach enabled the testing and validating of our work on three levels: First, on the single study level, second on the information derived from the meta-analysis, and third we were able to inspect our results on the ensemble approach level. The basic assumption in this model is, that in a set of G features, every feature has a baseline concentration, being the same for all studies. This baseline concentration may differ from study to study and from sample to sample.

If no study size information is provided they are sampled as: between 8 and 16 with probability 0.6, with probability 0.1 between 5 and 14 and with probability 0.3 between 15 and 30.

Value

A list of ExpressionSet Objects with features names "ART". Artificially down- or upregulated features are marked with "_D" or "_U"

Author(s)

Karl Kugler, karl@eigenlab.net

References

Hong F, Breitling R: A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments. *Bioinformatics* 2008, 24(3):374-382

Kong X, Mas V, Archer KJ: A non-parametric meta-analysis approach for combining independent microarray datasets: application using two microarray datasets pertaining to chronic allograft nephropathy. *BMC Genomics* 2008, 9:98

Examples

```
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = 0)
A[[1]]
exprs(A[[1]])
```

multiTtest

Function to perform t-tests on a list of ExpressionSets.

Description

This function helps in executing a t-test (as implemented in the **stats** package) on a list of ExpressionSet objects, which is the main data type used in almost all MADAM based functions.

Usage

```
multiTtest(es, cl, cl.val = c(0, 1), alternative = "two.sided", cluster = NULL)
```

Arguments

<code>es</code>	List of ExpressionSets.
<code>cl</code>	List of factors to discriminate between the two groups.
<code>cl.val</code>	Level values of <code>cl</code> (default: 0,1)).
<code>alternative</code>	Alternative hypothesis, either greater, less or two.sided (default) - greater: group 1 > group 2 - less: group 1 < group 2
<code>cluster</code>	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Value

Data.frame with id's as rownames and p-values for each set as columns.

Author(s)

Karl Kugler, karl@eigenlab.net

See Also

stats::t.test

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
pval <- multiTtest(A, cl, alternative= "two.sided", cluster = NULL)
rownames(pval) <- rownames(exprs(A[[1]]))
res <- fisherMethod(data=pval)
res
```

plotFDR

Plot results significance against the rank of the significance.

Description

For plotting the significances versus the ranks, the significances are first ranked. If ties occur, the first observed significance will get the lowest rank.

This helps in assessing the distribution of significances for meta analysis. Three types of studies can be annotated: single studies, meta-analysis of studies, and ensemble methods that combine meta-analysis.

Usage

```
plotFDR(x, title = "", types = factor(rep("s", ncol(x))))
```

Arguments

x	A matrix with the rownames being features and colnames being studies/methods. Entries are significances.
title	Title for image.
types	Three level factor indicating if a x comes from a normal method or a ensemble approach, difference will be in color and line type allowed levels are "s" single study, "m" meta analysis, and "e" ensemble approach.

Value

a plot

Author(s)

Karl Kugler, karl@eigenlab.net

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
pval <- multiTtest(A, cl, alternative= "two.sided")
rownames(pval) <- rownames(exprs(A[[1]]))
res <- fisherMethod(data=pval, cluster = NULL)
PP <- data.frame(pval, res$p.value)
colnames(PP) <- c(paste("study", 1:3), "fisher")
plotFDR(PP, types=c(rep("s", 3), "m"))
```

plotHeatMap

Function for plotting ranklist heatmap.

Description

This helps in plotting a heatmap of ranks from a set of studies. Typically the ranks are derived from the significances or a test statistic, but may of course represent effect sizes as well.

Usage

```
plotHeatMap(rankList, title = "", top.col = NULL, side.col = NULL,
col = rainbow(nrow(rankList)))
```

Arguments

rankList	Data.frame containing ranks, rows representing the features and columns the studies or methods...
title	Image title
top.col	Vector of colors used for plotting column headers (default= NULL)
side.col	Vector indicating how to print row side colors (default= NULL)
col	Colors to be used for plotting ranks (default: rainbow(nrow(rankList)))

Details

A ranklist heatmap is plotted using the function heatmap.2 from the gplots package. Clustering is done using the Euclidian distance.

Value

a plot

Author(s)

Karl Kugler, karl@eigenlab.net

See Also

gplots::heatmap.2

Examples

```
set.seed(123)
M <- sapply(1:5, sample, x=1:100, size=100)
plotHeatMap(M)
plotHeatMap(M, title = "", top.col = NULL, side.col = NULL,
col = rainbow(nrow(M)))
```

plotMAVolcano

Function for MA-volcano plot

Description

Function to plot a volcano plot using information from Effect Size based MA and significance based MA.

Usage

```
plotMAVolcano(x, title = "", col = "black", pch = ".", points = NULL,
points.col = "red", points.pch = 1)
```

Arguments

x	A matrix with two columns (first column containing effect size, second the significance).
title	Title of plot.
col	Color for volcano plot.
pch	Character type for volcano plot.
points	A matrix with two columns: specifying if certain points are to be drawn separately.
points.col	Character type for plotting points.
points.pch	color for plotting points

Value

Plots a MA-Volcano plot.

Author(s)

Karl Kugler, karl@eigenlab.net

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
res <- doES(A, cl, useREM = TRUE, nperm = 10, cluster = NULL)
plotMAVolcano(data.frame(res$g2up$MUvals, res$g2up$FDR))
```

zScoreFDRClust

Tool for Meta-analysis of gene expression data.

Description

A meta-analysis function for computing FDR with optional cluster.

Usage

```
zScoreFDRClust(esets, classes, useREM=TRUE,
               nperm=1000, CombineExp=1:length(esets), cluster = NULL)
```

Arguments

esets	A 'list' of 'ExpressionSet's, one expression set per experiment. All experiments must have the same variables(genes).
classes	A 'list' of class memberships, one per experiment. Each 'list' can only contain 2 levels.
useREM	A 'logical' value indicating whether or not to use a REM, 'TRUE', or a FEM, 'FALSE', for combining the z scores.
nperm	number of permutations to calculate the FDR
CombineExp	'vector' of integer- which experiments should be combined-default:all experiments
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Details

The function 'zScores' implements the approach of Choi et al. for for a set of 'ExpressionSet's. The function 'zScoreFDR' computes a FDR for each gene, both for each single experiment and for the combined experiment. The FDR is calculated as described in Choi et al. Up to now ties in the zscores are not taken into account in the calculation. The function might produce incorrect results in that case. The function also computes zScores, both for the combines experiment and for each single experiment.

'Clust'-functions are the same functions, extended for making cluster-computation possible and optional

Value

A 'matrix' with one row for each probe(set) and the following columns:

zSco_Ex_: For each single experiment the standardized mean difference, 'Effect_Ex_', divided by the estimated standard deviation, the square root of the 'EffectVar_Ex_' column.

MUvals: The combined standardized mean difference (using a FEM or REM)

MUsds: The standard deviation of the 'MUvals'.

zSco: The z statistic - the 'MUvals' divided by their standard deviations, 'MUsds'.

Qvals: Cochran's Q statistic for each gene.

df: The degree of freedom for the Chi-square distribution. This is equal to the number of combined experiments minus one.

Qpvalues: The probability that a Chi-square random variable, with 'df' degrees of freedom) has a higher value than the value from the Q statistic.

Chisq: The probability that a Chi-square random variate (with 1 degree of freedom) has a higher value than the value of zSco^2.

Effect_Ex_: The standardized mean difference for each single experiment.

EffectVar_Ex_: The variance of the standardized mean difference for each single experiment. Note that the three column names that end in an underscore are replicated, once for each experiment that is being analyzed.

Author(s)

M. Ruschhaupt Clustering option added by Matthias Wieser, UMIT

References

Choi et al, Combining multiple microarray studies and modeling interstudy variation. *Bioinformatics*, 2003, i84-i90.

See Also

[zScoreFDR](#)

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
res <- zScoreFDRClust(A, cl, useREM = TRUE, nperm = 10,
                    CombineExp=1:length(A), cluster = NULL)
res
```

zScoresClust

Tool for Meta-analysis of gene expression data.

Description

A meta-analysis function for computing zScores with optional cluster.

Usage

```
zScoresClust(esets, classes, useREM=TRUE,
             CombineExp=1:length(esets), cluster = NULL)
```

Arguments

esets	A 'list' of 'ExpressionSet's, one expression set per experiment. All experiments must have the same variables(genes).
classes	A 'list' of class memberships, one per experiment. Each 'list' can only contain 2 levels.
useREM	A 'logical' value indicating whether or not to use a REM, 'TRUE', or a FEM, 'FALSE', for combining the z scores.
CombineExp	'vector' of integer- which experiments should be combined-default:all experiments
cluster	A snow cluster object. If this package is used without parallel computing facilities, computing time may be much higher.

Details

The function 'zScores' implements the approach of Choi et al. for for a set of 'ExpressionSet's. The function 'zScoreFDR' computes a FDR for each gene, both for each single experiment and for the combined experiment. The FDR is calculated as described in Choi et al. Up to now ties in the zscores are not taken into account in the calculation. The function might produce incorrect results in that case. The function also computes zScores, both for the combines experiment and for each single experiment.

'Clust'-functions are the same functions, extended for making cluster-computation possible and optional

Value

A 'matrix' with one row for each probe(set) and the following columns:

zSco_Ex_: For each single experiment the standardized mean difference, 'Effect_Ex_', divided by the estimated standard deviation, the square root of the 'EffectVar_Ex_' column.

MUvals: The combined standardized mean difference (using a FEM or REM)

MUsds: The standard deviation of the 'MUvals'.

zSco: The z statistic - the 'MUvals' divided by their standard deviations, 'MUsds'.

Qvals: Cochran's Q statistic for each gene.

df: The degree of freedom for the Chi-square distribution. This is equal to the number of combined experiments minus one.

Qpvalues: The probability that a Chi-square random variable, with 'df' degrees of freedom) has a higher value than the value from the Q statistic.

Chisq: The probability that a Chi-square random variate (with 1 degree of freedom) has a higher value than the value of $zSco^2$.

Effect_Ex_: The standardized mean difference for each single experiment.

EffectVar_Ex_: The variance of the standardized mean difference for each single experiment. Note that the three column names that end in an underscore are replicated, once for each experiment that is being analyzed.

Author(s)

M. Ruschhaupt Clustering option added by Matthias Wieser, UMIT

References

Choi et al, Combining multiple microarray studies and modeling interstudy variation. Bioinformatics, 2003, i84-i90.

See Also

[zScores](#)

Examples

```
set.seed(123)
A <- generateRandomMADData(g = 100, perc.sig = 0.1, i = 3, k = rep(5, 6))
cl <- lapply(A, function(a){factor(as.numeric(a$group)-1)})
res <- zScoresClust(A, cl, useREM = TRUE,
                   CombineExp=1:length(A), cluster = NULL)
res
```

Index

- *Topic **NA**
 - corrNAGroupwise, [7](#)
 - *Topic **datagen**
 - generateRandomMADData, [16](#)
 - *Topic **datasets**
 - default.zero.subst, [9](#)
 - *Topic **hplot**
 - plotFDR, [18](#)
 - plotHeatMap, [19](#)
 - plotMAVolcano, [20](#)
 - *Topic **manip**
 - corrNA, [5](#)
 - corrNAGroupwise, [7](#)
 - corrVar, [8](#)
 - doEnsemble, [10](#)
 - *Topic **package**
 - MADAM-package, [2](#)
 - *Topic **univar**
 - calculateRankProduct, [3](#)
 - calculateRankSum, [4](#)
 - doES, [11](#)
 - doRhodesFDR, [12](#)
 - doRP, [13](#)
 - fisherMethod, [14](#)
 - fishersum, [15](#)
 - multiTtest, [17](#)
 - zScoreFDRClust, [21](#)
 - zScoresClust, [23](#)
- calculateRankProduct, [3](#), [5](#), [14](#)
calculateRankSum, [4](#), [4](#), [14](#)
corrNA, [5](#), [6](#), [7](#)
corrNAGroupwise, [7](#)
corrVar, [7](#), [8](#), [9](#)
- default.zero.subst, [9](#)
doEnsemble, [10](#)
doES, [11](#)
doRhodesFDR, [12](#)
doRP, [13](#)
- fisherMethod, [14](#)
fishersum, [15](#)
- generateRandomMADData, [16](#)
- MADAM (MADAM-package), [2](#)
MADAM-package, [2](#)
multiTtest, [17](#)
- plotFDR, [18](#)
plotHeatMap, [19](#)
plotMAVolcano, [20](#)
- RPadvance, [14](#)
- zScoreFDR, [12](#), [23](#)
zScoreFDRClust, [21](#)
zScores, [24](#)
zScoresClust, [23](#)