Package 'RankProd'

July 3, 2025

Version 3.35.0

Date 2016-09-16

Title Rank Product method for identifying differentially expressed genes with application in meta-analysis

Author Francesco Del Carratore

<francesco.delcarratore@manchester.ac.uk>,
Andris Jankevics <andris.jankevics@gmail.com>
Fangxin Hong <fxhong@jimmy.harvard.edu>,
Ben Wittner <Wittner.Ben@mgh.harvard.edu>,
Rainer Breitling <rainer.breitling@manchester.ac.uk>,
and Florian Battke <battke@informatik.uni-tuebingen.de>

Maintainer Francesco Del Carratore

<francescodc87@gmail.com>

Depends R (>= 3.2.1), stats, methods, Rmpfr, gmp

Imports graphics

Description Non-parametric method for identifying differentially expressed (up- or down- regulated) genes based on the estimated percentage of false predictions (pfp). The method can combine data sets from different origins (meta-analysis) to increase the power of the identification.

License file LICENSE

License_restricts_use yes

biocViews DifferentialExpression, StatisticalMethod, Software, ResearchField, Metabolomics, Lipidomics, Proteomics, SystemsBiology, GeneExpression, Microarray, GeneSignaling

git_url https://git.bioconductor.org/packages/RankProd

git_branch devel

git_last_commit 3628151

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-07-02

Contents

`Apples` *Metabolomics data on spiked apples*

Description

A dataset of LC-MS features, obtained from twenty apples. The last ten apples are spiked with known compounds. This set provides a test case for biomarker selection methods: the task is to retrive the true biomarker variables. The raw LC-MS data hava been converted to CDF format and processed with XCMS to obtain the basepeaks.

Usage

```
data(Apples)
```

Value

The format is a list of four elements:

`mz`	the m/z values of the features (rounded)
`rt`	the retention times of the features
`apples.data`	a matrix containing the intensities in the individual samples
`apples.data.vsn`	a matrix containing the intesities after variance stabilization and normalization performed with the vsn package
`Biom`	the indices of the "true" biomarkers
`apples.cl`	numeric vector encoding which samples are part of the spiked class (code 1) and which ones are controls (code 0)

Author(s)

Francesco Del Carratore

References

P. Franceschi, D. Masuero, U. Vrhovsek, F. Mattivi and R. Wehrens: A benchmark spike-in data set for biomarker identification in metabolomics. *J. Chemom.* 26, 16-24 (2012)

R. Wehrens, P. Franceschi, U. Vrhovsek and F. Mattivi. Stability-based biomarker selection. *Analytica Chimica Acta* (2011), 705, 15-23. http://dx.doi.org/10.1016/j.aca.2011.01.039

Examples

```
data(Apples)
## show features identified in all apples
plot(rt, mz,
     xlab = "Retention time (s)", ylab = "m/z",
     main = "Spiked apples - subset")
```

---

arab *Genomic Response to Brassinosteroid in Arabidopsis*

Description

These data are from Affy ATH1 array experiments of genomic response to brassinosteroid in Arabidopsis conducted by two laoratories. The data set contains 500 random selected genes and 10 samples, 6 from lab 1 and 4 from lab 2. Data were pre-processed by RMA

Usage

```
data(arab)
```

Value

`arab`	matrix of gene expression levels of 500 genes from 10 samples, rows correspond to genes and columns to mRNA samples.
`arab.cl`	numeric vector encoding the treatment classes, 5 brassinosteroid-treated cases (code 1) and 5 control cases (code 0)
`arab.gnames`	character vector containing the AffyID of the 500 genes for the expression matrix `arab`
`arab.origin`	numeric vector encoding the origin of the samples, 6 samples from lab 1 (code 1) and 4 samples from lab 2 (code 2)

References

Nemhauser JL, Mockler TC, Chory J. Interdependency of brassinosteroid and auxin signaling in Arabidopsis. *PLoS Biol.* 2004 21460.
Microarray data from AtGenExpress (http://arabidopsis.org/info/expression/ATGenExpress.jsp)

golub *A subset of the Gene expression dataset from Golub et al. (1999)*

Description

Gene expression data (500 genes and 38 tumor mRNA samples) from the leukemia microarray study of Golub et al. (1999). The original dataset contains 3051 genes

Usage

```
data(golub)
```

Value

golub	matrix of gene expression levels for the 38 tumor mRNA samples. Rows correspond to genes and columns to mRNA samples.
golub.cl	numeric vector encoding the tumor classes, 27 acute lymphoblastic leukemia (ALL) cases (code 0) and 11 acute myeloid leukemia (AML) cases (code 1).
golub.gnames	a matrix containing the names of the 500 genes for the expression matrix golub. The three columns correspond to the gene index, ID, and Name, respectively.

Source

Golub et al. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, *Science*, Vol. 286:531-537.
http://www-genome.wi.mit.edu/MPR/ .

References

S. Dudoit, J. Fridlyand, and T. P. Speed (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, Vol. 97, No. 457, p. 77–87.

lymphoma *Subset of the Intensity data for 8 cDNA slides with CLL and DLBL samples from the Alizadeh et al. paper in Nature 2000*

Description

8 cDNA chips from Alizadeh lymphoma paper

Usage

```
data(lymphoma)
```

Format

lymphoma is an exprSet containing the data of 8 chips from the lymphoma dataset by Alizadeh et al. (see references). Each chip represents two samples: on color channel 1 (CH1, Cy3, green) the common reference sample, and on color channel 2 (CH2, Cy5, red) the various disease samples. See pData(lymphoma). The 9216x16 matrix exprs(lymphoma) contains the background-subtracted spot intensities (CH1I-CH1B and CH2I-CH2B, respectively).

Details

The chip intensity files were downloaded from the Stanford microarray database. Starting from the link below, this was done by following the links *Published Data -> Alizadeh AA, et al. (2000) Nature 403(6769):503-11 -> Data in SMD -> Display Data*, and selecting the following 8 slides:

lc7b019
lc7b047
lc7b048
lc7b056
lc7b057
lc7b058
lc7b069
lc7b070

Then, the script makedata.R from the scripts subdirectory of this package was run to generate the R data object.

Value

lym.exp	8 cDNA chips from Alizadeh lymphoma paper
lynx	Is a time series with numbers of annual numbers of lynx trapping in Canada from 1821-1934. Taken from Brockwell & Davis (1991), this appears to be the series considered by Campbell & Walker(1977)

Source

http://genome-www5.stanford.edu/MicroArray/SMD

References

A. Alizadeh et al., Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403(6769):503-11, Feb 3, 2000.

---

plotRP *Graphical Display of the Rank Product/Sum analysis*

Description

Plot a graph of the estimated pfp vs the number of identified genes

Usage

```
plotRP(x, cutoff=NULL)
```

Arguments

x	the value returned by function RP, RPadvance, RSadvance, RankProducts or RP.advance
cutoff	The pfp threshold value used to select genes

Value

A graphical display of the estimated pfp vs number of identified genes, which is also the gene rank of its original rank product/sum across all comparison. If cutoff is sepcified, a horizontal line will be plotted on the graphic to indicate the positon of the cutoff point, and all genes identified will be marked in red.

Two plots will be displayed, one for the identification of up-regulated genes in class 2, one for the identification of down-regulated genes in class 2

Author(s)

Fangxin Hong, <fhong@salk.edu>
Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk>

See Also

topGene RP RPadvance RSadvance

Examples

```
# Load the data of Golub et al. (1999). data(golub)
#contains a 3051x38 gene expression
# matrix called golub, a vector of length called golub.cl
#that consists of the 38 class labels,
# and a matrix called golub.gnames whose third column contains the gene names.
data(golub)

#use a subset of data as example, apply the rank product method
subset <- c(1:4,28:30)
#Setting rand=123, to make the results reproducible,
#identify genes that are up-regulated in class 2
#(class label =1)
RP.out <- RP(golub[,subset],golub.cl[subset], rand=123)

#plot the results
plotRP(RP.out,cutoff=0.05)
```

RandProd-internal *Internal RankProd functions*

Description

Internal RankProd functions.

Details

These functions are not meant to be called directly by the user.

Author(s)

Fangxin Hong, <fhong@salk.edu> \ Rainer Breitling, <rainer.breitling@manchester.ac.uk> \ Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk> \ Andris Janckevics, <andris.jankevics@gmail.com>

`RankProducts` *Rank Product/Rank Sum Analysis*

Description

The function performs the Rank Product (or Rank Sum) method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis.

Usage

```
RankProducts(data, cl, logged = TRUE, na.rm = TRUE, gene.names = NULL,
plot = FALSE, rand = NULL, calculateProduct = TRUE, MinNumOfValidPairs = NA,
RandomPairs = NA, huge = FALSE, fast = TRUE, tail.time = 0.05)
```

Arguments

`data`	the data set that should be analyzed. Every row of this dataset must correspond to a gene
`cl`	a vector containing the class labels of the samples. In the two class unpaired case, the label of a sample is either 0 (e.g., control group) or 1 (e.g., case group). For one class data, the label for each sample should be 1
`logged`	if "TRUE" data have been previously log transformed. Otherwise it should be set as "FALSE"
`na.rm`	if "FALSE", the NA value will not be used in computing rank. If "TRUE" (default), the missing values will be replaced by the genewise median of the non-missing values. Gene with a number of missing values greater than "MinNumOfValidPairs" are still not considered in the analysis
`gene.names`	if "NULL", no gene name will be attached to the outputs, otherwise it contains the vector of gene names
`plot`	if "TRUE", plot the estimated pfp vs the rank of each gene
`rand`	if specified, the random number generator will be put in a reproducible state
`calculateProduct`	if calculateProduct="TRUE" (default) the rank product method is performed. Otherwise the rank sum method is performed
`MinNumOfValidPairs`	a parameter that indicates the minimum number of NAs accepted per each gene. If it is set to NA (default) the half of the number of replicates is used
`RandomPairs`	number of random pairs generated in the function, if set to NA (default), the odd integer closer to the square of the number of replicates is used
`huge`	if "TRUE" not all the outputs are evaluated in order to save space

fast	if "FALSE" the exact p-values for the Rank Sum are evaluated for any size of the dataset. Otherwise (default), if the size of the dataset is too big, only the p-values that can be computed in "tail.time" minutes (starting from the tail) are evaluated with the exact method. The others are estimated with the Gaussian approximation. If calculateProduct="TRUE" this parameter is ignored
tail.time	the time (default 0.05 min) dedicated to evaluate the exact p-values for the Rank Sum. If calculateProduct="TRUE" this parameter is ignored

Value

A summary of the results obtained by the Rank Product (or Rank Sum) method.

pfp	Estimated percentage of false positive predictions (pfp), both considering up-regulated an downregulated genes
pval	Estimated pvalues per each gene being up- and down-regulated
RPs/RSs	The rank-product/rank-sum statistics evaluated per each gene
RPrank/RSrank	rank of the Rank Product (or Rank Sum) of each gene in ascending order
Orirank	Ranks obtained when considering each possible pairing. In this version of the package, this is not used to compute Rank Product (or Rank Sum), but it is kept for backward compatibility
AveFC	Fold change of average expressions (class1/class2). log fold-change if data has been log transformed, original fold change otherwise
allrank1	Fold change of class 1/class 2 under each origin. log fold-change if data has been log transformed, original fold change otherwise
allrank2	Fold change of class 2/class 1 under each origin. log fold-change if data has been log transformed, original fold change otherwise
nrep	Total number of replicates
groups	Vector of labels (as cl)
RandomPairs_ranks	a matrix containing the ranks evaluated for each RandomPair

Author(s)

Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk>
Andris Janckevics, <andris.jankevics@gmail.com>

References

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

See Also

topGene RP RPadvance plotRP RP.advance RSadvance

Examples

```
# Load the data of Golub et al. (1999). data(golub)
# contains a 3051x38 gene expression
# matrix called golub, a vector of length called golub.cl
# that consists of the 38 class labels,
# and a matrix called golub.gnames whose third column
# contains the gene names.
data(golub)


#use a subset of data as example, apply the rank
#product method
subset <- c(1:4,28:30)
#Setting rand=123, to make the results reproducible,

RP.out <- RankProducts(golub[,subset],golub.cl[subset],rand=123)

# class 2: label =1, class 1: label = 0
#pfp for identifying genes that are up-regulated in class 2
#pfp for identifying genes that are down-regulated in class 2
head(RP.out$pfp)
#Rank Sum
RS.out <-RankProducts(golub[,subset],golub.cl[subset],rand=123,
    calculateProduct=FALSE)
head(RS.out$pfp)
```

RP *Rank Product Analysis*

Description

The function performs the Rank Product method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis. This function has been kept only to guarantee backward compatibility, in fact the same results can be obtained by RankProducts.

Usage

```
RP(data, cl, num.perm = 100, logged = TRUE, na.rm = TRUE, gene.names = NULL,
plot = FALSE, rand = NULL, huge = FALSE)
```

Arguments

data	the function performs the Rank Product (or Rank Sum) method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis
cl	a vector containing the class labels of the samples. In the two class unpaired case, the label of a sample is either 0 (e.g., control group) or 1 (e.g., case group). For one class data, the label for each sample should be 1
num.perm	in this version of the package, this parameter is not used any more, but it is kept for backward compatibility
logged	if "TRUE" data have been previously log transformed. Otherwise it should be set as "FALSE"

na.rm	if "FALSE", the NA value will not be used in computing rank. If "TRUE" (default), the missing values will be replaced by the genewise median of the non-missing values. Gene with a number of missing values greater than 50% are still not considered in the analysis
gene.names	if "NULL", no gene name will be attached to the outputs, otherwise it contains the vector of gene names
plot	if "TRUE", plot the estimated pfp vs the rank of each gene
rand	if specified, the random number generator will be put in a reproducible state
huge	if "TRUE" not all the outputs are evaluated in order to save space

Value

A summary of the results obtained by the Rank Product method.

pfp	estimated percentage of false positive predictions (pfp), both considering upregulated an downregulated genes
pval	estimated pvalues per each gene being up- and down-regulated
RPs	the Rank Product statistics evaluated per each gene
RPrank	rank of the Rank Product of each gene in ascending order
Orirank	ranks obtained when considering each possible pairing. In this version of the package, this is not used to compute Rank Product (or Rank Sum), but it is kept for backward compatibility
AveFC	fold changes of average expressions (class1/class2). log fold-change if data has been log transformed, original fold change otherwise
allrank1	fold change of class 1/class 2 under each origin. log fold-change if data has been log transformed, original fold change otherwise
allrank2	fold change of class 2/class 1 under each origin. log fold-change if data has been log transformed, original fold change otherwise
nrep	total number of replicates
groups	vector of labels (as cl)
RandomPairs_ranks	a matrix containing the ranks evaluated for each RandomPair

Note

Percentage of false prediction (pfp), in theory, is equivalent of false discovery rate (FDR), and it is possible to be large than 1.

The function looks for up- and down- regulated genes in two seperate steps, thus two pfps and pvalues are computed and used to identify gene that belong to each group.

This function is suitable to deal with data from a single origin, e.g. single experiment. If the data has different origin, e.g. generated at different laboratories, please refer RP.advance.

Author(s)

Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk>
Andris Janckevics, <andris.jankevics@gmail.com>

References

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

See Also

topGene RSadvance RPadvance plotRP RP.advance RankProducts

Examples

```
# Load the data of Golub et al. (1999). data(golub)
# contains a 3051x38 gene expression
# matrix called golub, a vector of length called golub.cl
# that consists of the 38 class labels,
# and a matrix called golub.gnames whose third column
# contains the gene names.
data(golub)


#use a subset of data as example, apply the rank
#product method
subset <- c(1:4,28:30)
#Setting rand=123, to make the results reproducible,

RP.out <- RP(golub[,subset],golub.cl[subset],rand=123)

# class 2: label =1, class 1: label = 0
#pfp for identifying genes that are up-regulated in class 2
#pfp for identifying genes that are down-regulated in class 2
head(RP.out$pfp)
```

RP.advance *Advanced Rank Product/Rank Sum Analysis*

Description

The function performs the Rank Product (or Rank Sum) method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis. It is also possible to combine data from different studies (e.g. datasets generated by different laboratories)

Usage

```
RP.advance(data, cl, origin, logged = TRUE, na.rm = TRUE, gene.names = NULL,
plot = FALSE, rand = NULL, calculateProduct = TRUE, MinNumOfValidPairs = NA,
RandomPairs = NA, huge = FALSE, fast = TRUE, tail.time = 0.05)
```

Arguments

data	the data set that should be analyzed. Every row of this dataset must correspond to a gene

cl	a vector containing the class labels of the samples. In the two class unpaired case, the label of a sample is either 0 (e.g., control group) or 1 (e.g., case group). For one class data, the label for each sample should be 1
origin	a vector containing the origin labels of the samples. The label is the same for samples within one lab and different for samples from different labs.
logged	if "TRUE" data have been previously log transformed. Otherwise it should be set as "FALSE"
na.rm	if "FALSE", the NA value will not be used in computing rank. If "TRUE" (default), the missing values will be replaced by the genewise median of the non-missing values. Gene with a number of missing values greater than "MinNumOfValidPairs" are still not considered in the analysis
gene.names	if "NULL", no gene name will be attached to the outputs, otherwise it contains the vector of gene names
plot	if "TRUE", plot the estimated pfp vs the rank of each gene
rand	if specified, the random number generator will be put in a reproducible state
calculateProduct	if calculateProduct="TRUE" (default) the rank product method is performed. Otherwise the rank sum method is performed
MinNumOfValidPairs	a parameter that indicates the minimum number of NAs accepted per each gene. If it is set to NA (default) the half of the number of replicates is used
RandomPairs	number of random pairs generated in the function, if set to NA (default), the odd integer closer to the square of the number of replicates is used
huge	if "TRUE" not all the outputs are evaluated in order to save space
fast	if "FALSE" the exact p-values for the Rank Sum are evaluated for any size of the dataset. Otherwise (default), if the size of the dataset is too big, only the p-values that can be computed in "tail.time" minutes (starting from the tail) are evaluated with the exact method. The others are estimated with the Gaussian approximation. If calculateProduct="TRUE" this parameter is ignored
tail.time	the time (default 0.05 min) dedicated to evaluate the exact p-values for the Rank Sum.If calculateProduct="TRUE" this parameter is ignored.

Value

A summary of the results obtained by the Rank Product (or Rank Sum) method.

pfp	estimated percentage of false positive predictions (pfp), both considering upregulated an downregulated genes
pval	estimated pvalues per each gene being up- and down-regulated
RPs/RSs	the Rank Product (or Rank Sum) statistics evaluated per each gene
RPrank/RSrank	rank of the Rank Product (or Rank Sum) of each gene in ascending order
Orirank	ranks obtained when considering each possible pairing. In this version of the package, this is not used to compute Rank Product (or Rank Sum), but it is kept for backward compatibility
AveFC	fold changes of average expressions (class1/class2). log fold-change if data has been log transformed, original fold change otherwise
allrank1	fold change of class 1/class 2 under each origin. log fold-change if data has been log transformed, original fold change otherwise

allrank2	fold change of class 2/class 1 under each origin. log fold-change if data has been log transformed, original fold change otherwise
nrep	total number of replicates
groups	vector of labels (as cl)
RandomPairs_ranks	a matrix containing the ranks evaluated for each RandomPair

Author(s)

Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk>
Andris Janckevics, <andris.jankevics@gmail.com>

References

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

See Also

topGene RP RPadvance plotRP RankProducts RSadvance

Examples

```
# Load the data of Golub et al. (1999). data(golub)
# contains a 3051x38 gene expression
# matrix called golub, a vector of length called golub.cl
# that consists of the 38 class labels,
# and a matrix called golub.gnames whose third column
# contains the gene names.
data(golub)

##For data with single origin
subset <- c(1:4,28:30)
origin <- rep(1,7)
#identify genes
RP.out <- RP.advance(golub[,subset],golub.cl[subset],
            origin,plot=FALSE,rand=123)

#For data from multiple origins

# Load the data arab in the package, which contains
# the expression of 22,081 genes
# of control and treatment group from the experiments
# indenpently conducted at two
#laboratories.
data(arab)
arab.origin #1 1 1 1 1 1 2 2 2 2
arab.cl #0 0 0 1 1 1 0 0 1 1
RP.adv.out <- RP.advance(arab,arab.cl,arab.origin,
                gene.names=arab.gnames,logged=TRUE,rand=123)

attributes(RP.adv.out)
head(RP.adv.out$pfp)
head(RP.adv.out$RPs)
```

```
head(RP.adv.out$AveFC)



#Suppose we want to check the consistence of the data
#sets generated in two different
#labs. For example, we would look for genes that were \
# measured to be up-regulated in
#class 2 at lab 1, but down-regulated in class 2 at lab 2.\
data(arab)
arab.cl2 <- arab.cl

arab.cl2[arab.cl==0 &arab.origin==2] <- 1

arab.cl2[arab.cl==1 &arab.origin==2] <- 0

arab.cl2
##[1] 0 0 0 1 1 1 1 1 0 0


#look for genes differentially expressed
#between hypothetical class 1 and 2
arab.sub=arab[1:500,] ##using subset for fast computation
arab.gnames.sub=arab.gnames[1:500]
Rsum.adv.out <- RP.advance(arab.sub,arab.cl2,arab.origin,calculateProduct
                =FALSE,logged=TRUE,gene.names=arab.gnames.sub,rand=123)

attributes(Rsum.adv.out)
```

RPadvance *Advanced Rank Product Analysis*

Description

The function performs the Rank Product method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis. It is also possible to combine data from different studies (e.g. datasets generated by different laboratories. This function has been kept only to guarantee backward compatibility, in fact the same results can be obtained by RankProducts.

Usage

```
RPadvance(data, cl, origin, num.perm = 100, logged = TRUE, na.rm = TRUE,
gene.names = NULL, plot = FALSE, rand = NULL, huge = FALSE)
```

Arguments

data	the data set that should be analyzed. Every row of this dataset must correspond to a gene
cl	a vector containing the class labels of the samples. In the two class unpaired case, the label of a sample is either 0 (e.g., control group) or 1 (e.g., case group). For one class data, the label for each sample should be 1
origin	a vector containing the origin labels of the samples. The label is the same for samples within one lab and different for samples from different labs.

num.perm	in this version of the package, this parameter is not used any more, but it is kept for backward compatibility
logged	if "TRUE" data have been previously log transformed. Otherwise it should be set as "FALSE"
na.rm	if "FALSE", the NA value will not be used in computing rank. If "TRUE" (default), the missing values will be replaced by the genewise median of the non-missing values. Gene with a number of missing values greater than 50% are still not considered in the analysis
gene.names	if "NULL", no gene name will be attached to the outputs, otherwise it contains the vector of gene names
plot	if "TRUE", plot the estimated pfp vs the rank of each gene
rand	if specified, the random number generator will be put in a reproducible state
huge	if "TRUE" not all the outputs are evaluated in order to save space

Value

A summary of the results obtained by the Rank Product method.

pfp	estimated percentage of false positive predictions (pfp), both considering upregulated an downregulated genes
pval	estimated pvalues per each gene being up- and down-regulated
RPs	the Rank Product statistics evaluated per each gene
RPrank	rank of the Rank Product of each gene in ascending order
Orirank	ranks obtained when considering each possible pairing. In this version of the package, this is not used to compute Rank Product (or Rank Sum), but it is kept for backward compatibility
AveFC	fold changes of average expressions (class1/class2). log fold-change if data has been log transformed, original fold change otherwise
allrank1	fold change of class 1/class 2 under each origin. log fold-change if data has been log transformed, original fold change otherwise
allrank2	fold change of class 2/class 1 under each origin. log fold-change if data has been log transformed, original fold change otherwise
nrep	total number of replicates
groups	vector of labels (as cl)

Note

Percentage of false prediction (pfp), in theory, is equivalent of false discovery rate (FDR), and it is possible to be large than 1.

The function looks for up- and down- regulated genes in two seperate steps, thus two pfps are computed and used to identify gene that belong to each group. The function is able to replace function RP in the same library. it is a more general version, as it is able to handle data from differnt origins.

Author(s)

Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk>
Andris Janckevics, <andris.jankevics@gmail.com>

References

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

See Also

topGene RP RSadvance plotRP RP.advance RankProducts

Examples

```
# Load the data of Golub et al. (1999). data(golub)
# contains a 3051x38 gene expression
# matrix called golub, a vector of length called golub.cl
# that consists of the 38 class labels,
# and a matrix called golub.gnames whose third column
# contains the gene names.
data(golub)

##For data with single origin
subset <- c(1:4,28:30)
origin <- rep(1,7)
#identify genes
RP.out <- RPadvance(golub[,subset],golub.cl[subset],
        origin,plot=FALSE,rand=123)

#For data from multiple origins

#Load the data arab in the package, which contains
# the expression of 22,081 genes
# of control and treatment group from the experiments
#indenpently conducted at two
#laboratories.
data(arab)
arab.origin #1 1 1 1 1 1 2 2 2 2
arab.cl #0 0 0 1 1 1 0 0 1 1
RP.adv.out <- RPadvance(arab,arab.cl,arab.origin,
                num.perm=100,gene.names=arab.gnames,logged=TRUE,rand=123)

attributes(RP.adv.out)
head(RP.adv.out$pfp)
head(RP.adv.out$RPs)
head(RP.adv.out$AveFC)
```

RSadvance *Advanced Rank Sum Analysis*

Description

The function performs the Rank Sum method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis. It is also possible to combine data from different studies (e.g. datasets generated by different laboratories. This function has been kept only to guarantee backward compatibility, in fact the same results can be obtained by RankProducts.

Usage

```
RSadvance(data, cl, origin, num.perm = 100, logged = TRUE, na.rm = TRUE,
gene.names = NULL, plot = FALSE, rand = NULL, huge = FALSE, fast = TRUE,
tail.time = 0.05)
```

Arguments

data	the data set that should be analyzed. Every row of this dataset must correspond to a gene
cl	a vector containing the class labels of the samples. In the two class unpaired case, the label of a sample is either 0 (e.g., control group) or 1 (e.g., case group). For one class data, the label for each sample should be 1
origin	a vector containing the origin labels of the samples. The label is the same for samples within one lab and different for samples from different labs.
num.perm	in this version of the package, this parameter is not used any more, but it is kept for backward compatibility
logged	if "TRUE" data have been previously log transformed. Otherwise it should be set as "FALSE"
na.rm	if "FALSE", the NA value will not be used in computing rank. If "TRUE" (default), the missing values will be replaced by the genewise median of the non-missing values. Gene with a number of missing values greater than 50% are still not considered in the analysis
gene.names	if "NULL", no gene name will be attached to the outputs, otherwise it contains the vector of gene names
plot	if "TRUE", plot the estimated pfp vs the rank of each gene
rand	if specified, the random number generator will be put in a reproducible state
huge	if "TRUE" not all the outputs are evaluated in order to save space
fast	if "FALSE" the exact p-values for the Rank Sum are evaluated for any size of the dataset. Otherwise (default), if the size of the dataset is too big, only the p-values that can be computed in "tail.time" minutes (starting from the tail) are evaluated with the exact method. The others are estimated with the Gaussian approximation. If calculateProduct="TRUE" this parameter is ignored
tail.time	the time (default 0.05 min) dedicated to evaluate the exact p-values for the Rank Sum. If calculateProduct="TRUE" this parameter is ignored

Value

A result of identifying differentially expressed genes between two classes. The identification consists of two parts, the identification of up-regulated and down-regulated genes in class 2 compared to class 1, respectively.

pfp	Estimated percentage of false positive predictions (pfp) up to the position of each gene under two identificaiton each
pval	Estimated pvalues for each gene being up- and down-regulated
RSs	Rank-sum (average rank) of each genes
RSrank	Rank of the rank sum of each gene in ascending order
Orirank	Ranks in each possible pairing, in this version of the function this is not used to compute rank sum. It is here only for backward compatibility

AveFC	Fold change of average expression under class 1 over that under class 2, if multiple origin, than avraged across all origin. Log-fold change if data is in log scaled, original fold change if data is unlogged
allrank1	Fold change of class 1/class 2 under each origin. Log-fold change if data is in log scaled
allrank2	Fold change of class 2/class 1 under each origin. Log-fold change if data is in log scaled
nrep	Total number of replicates considering all the different origins
groups	Vector of labels (as cl).

Note

Percentage of false prediction (pfp), in theory, is equivalent of false discovery rate (FDR), and it is possible to be large than 1.

The function looks for up- and down- regulated genes in two seperate steps, thus two pfps are computed and used to identify gene that belong to each group.

The function is able to deal with single or multiple-orgin studies. It is similar to funcion RP.advance expect a rank sum is computed instead of rank product. This method is more sensitive to individual rank values, while rank product is more robust to outliers (refer RankProd vignette for details)

Author(s)

Francesco Del Carratore, <francesco.delcarratore@postgrad.manchester.ac.uk>
Andris Janckevics, <andris.jankevics@gmail.com>

References

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

See Also

topGene RP RPadvance plotRP RP.advance RankProducts

Examples

```
#Suppose we want to check the consistence of the data
#sets generated in two different
#labs. For example, we would look for genes that were \
# measured to be up-regulated in
#class 2 at lab 1, but down-regulated in class 2 at lab 2.\
data(arab)
arab.cl2 <- arab.cl

arab.cl2[arab.cl==0 &arab.origin==2] <- 1

arab.cl2[arab.cl==1 &arab.origin==2] <- 0

arab.cl2
##[1] 0 0 0 1 1 1 1 1 0 0
```

```
#look for genes differentially expressed
#between hypothetical class 1 and 2
arab.sub=arab[1:500,] ##using subset for fast computation
arab.gnames.sub=arab.gnames[1:500]
Rsum.adv.out <- RSadvance(arab.sub,arab.cl2,arab.origin,
                          num.perm=100,logged=TRUE,
                          gene.names=arab.gnames.sub,rand=123)

attributes(Rsum.adv.out)
```

---

topGene *topGene*

Description

Indentify diiffentially expressed genes using rank product method

Usage

```
topGene(x,cutoff=NULL,method="pfp",num.gene=NULL,logged=TRUE,
                    logbase=2,gene.names=NULL)
```

Arguments

x	the value returned by function RP, RPadvance, RSadvance, RankProducts or RP.advance
cutoff	The pfp threshold value used to select genes
method	if cutoff is provided, the method needs to be selected to identify genes. "pfp" uses percentage of false prediction, which is a default setting. "pval" uses p-values which is less stringent than pfp
logged	if "TRUE", data has been logged, otherwise set it to "FALSE"
logbase	base used when taking log, used to restore the fold change.The default value is 2, this will be ignored if logged=FALSE
gene.names	if "NULL", no gene name will be attached to the output table
num.gene	number of candidates genes of interests, if cutoff is provided, this will be ignored

Value

Two tables of identified genes with gene.index: index of gene in the original data set RP/Rsum: Computed rank product/sum for each gene FC:(class1/class2): Expression Fold change of class 1/ class 2. pfp: estimated pfp for each gene if the gene is used as cutoff point P.value: estimated p-value for each gene Table 1 list genes that are up-regulated under class 2, Table 1 ist genes that are down-regulated under class 2

Author(s)

Fangxin Hong <fhong@salk.edu>

References

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

See Also

plotRP RP RPadvance RSadvance

Examples

```
# Load the data of Golub et al. (1999). data(golub)
# contains a 3051x38 gene expression
# matrix called golub, a vector of length called golub.cl
# that consists of the 38 class labels,
# and a matrix called golub.gnames whose third column
# contains the gene names.
data(golub)

#use a subset of data as example, apply the rank
#product method
subset <- c(1:4,28:30)
#Setting rand=123, to make the results reproducible,

#identify genes
RP.out <- RP(golub[,subset],golub.cl[subset],rand=123)

#get two lists of differentially expressed genes
#by setting FDR (false discivery rate) =0.05

table=topGene(RP.out,cutoff=0.05,method="pfp",logged=TRUE,logbase=2,
               gene.names=golub.gnames[,3])
table$Table1
table$Table2

#using pvalue<0.05
topGene(RP.out,cutoff=0.05,method="pval",logged=TRUE,logbase=2,
        gene.names=golub.gnames[,3])

#by selecting top 10 genes
topGene(RP.out,num.gene=10,gene.names=golub.gnames[,3])
```

Index