

Package ‘RankProd’

October 9, 2013

Version 2.32.0

Date 2011-12-13

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

Author Fangxin Hong <fxhong@jimmy.harvard.edu> and Ben Wittner <Wittner.Ben@mgh.harvard.edu> with contribution from Rainer Breitling <r.breitling@bio.gla.ac.uk>, Colin Smith <colin@colin-smith.org>, and Florian Battke <battke@informatik.uni-tuebingen.de>

Maintainer Fangxin Hong <fxhong@jimmy.harvard.edu>

Depends R (>= 1.9.0)

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

R topics documented:

arab	2
golub	2
lymphoma	3
plotGene	4
plotRP	6
RP	7
RPadvance	9
RSadvace	11
topGene	14

arab *Genomic Response to Brassinosteroid in Arabidopsis*

Description

This data is from Affy ATH1 array experiments of genomic response to brassinosteroid in Arabidopsis conducted at two laboratories. The data set contains random selected 500 genes and 10 samples, 6 from lab 1 and 4 from lab 2. Data was 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 indicating the treatment class, 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 indicating 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 *sub set 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). Original data set contain 3051 genes

Usage

data(golub)

Value

golub	matrix of gene expression levels for the 38 tumor mRNA samples, rows correspond to genes (500 genes) and columns to mRNA samples.
golub.cl	numeric vector indicating the tumor class, 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	<i>Subset of the Intensity data for 8 cDNA slides with CLL and DLBL samples from the Alizadeh et al. paper in Nature 2000</i>
----------	---

Description

8 cDNA chips from Alizadeh lymphoma paper

Usage

data(lymphoma)

Format

lymphoma is an `exprSet` containing the data from 8 chips from the lymphoma data set 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.

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.

plotGene

Graphical Display of The expression levels

Description

Plot the expression values and print statistical results for each individual gene based on a user query

Usage

```
plotGene(gene.to.plot,x,gene.names=NULL, data, cl,origin,
         logged=TRUE,logbase=2)
```

Arguments

<code>gene.to.plot</code>	Name of the gene to be plotted
<code>x</code>	the value returned by function <code>RP</code> , <code>RPadvance</code> or <code>RSadvance</code>
<code>gene.names</code>	Names of all genes in the data set. If "NULL", rownames of the data will be used
<code>data</code>	the same as that used in <code>RP</code> or <code>RPadvance</code>
<code>cl</code>	the same as that used in <code>RP</code> or <code>RPadvance</code>

origin	a vector containing the origin labels of the sample. e.g. for the data sets generated at multiple laboratories, the label is the same for samples within one lab and different for samples from different labs. The same as that used in RPadvance
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

Value

A graphical display of the expression levels of the input gene. The estimated statistics for differential expression will be printed on the plot as well as output in the screen. The statistics include: F.C.: fold-change under each dataset if multiple datasets are used AveFC: average fold-change across all datasets pfp(pval): estimated percentage of false prediction (p-value) for differential expression under each of the two tests: up-regulation in class 2 compared with class 1 and down-regulation in class 2 compared with class 1

Author(s)

Fangxin Hong <fhong@salk.edu>

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)
```

Description

Plot a graphical display of the estimated pfp vs number of identified genes

Usage

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

Arguments

x	the value returned by function RP, RPadvance or RSadvance
cutoff	threshold in pfp 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 specified, a horizontal line will be plotted on the graphic to indicate the position of the cutoff point, and all genes identified will be marked 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>

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)

```

RP

*Rank Product Analysis of Microarray***Description**

Perform rank product method to identify differentially expressed genes. It is possible to do either a one-class or two-class analysis.

Usage

```

RP(data,c1,num.perm=100,logged=TRUE,
    na.rm=FALSE,gene.names=NULL,plot=FALSE, rand=NULL, huge=FALSE)

```

Arguments

<code>data</code>	the data set that should be analyzed. Every row of this data set must correspond to a gene.
<code>c1</code>	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.
<code>num.perm</code>	number of permutations used in the calculation of the null density. Default is 'num.perm=100'.
<code>logged</code>	if "TRUE", data has been logged, otherwise set it to "FALSE"
<code>na.rm</code>	if 'FALSE' (default), the NA value will not be used in computing rank. If 'TRUE', the missing values will be replaced by the gene-wise mean of the non-missing values. Gene with all values missing will be assigned "NA"
<code>gene.names</code>	if "NULL", no gene name will be assigned to the estimated percentage of false positive predictions (pfp).
<code>plot</code>	If "TRUE", plot the estimated pfp versus the rank of each gene.
<code>rand</code>	if specified, the random number generator will be put in a reproducible state using the rand value as seed.
<code>huge</code>	If "TRUE", use an alternative method for computation. Using considerably less memory, this allows the Rank Product to be calculated for larger input data (see details). However, the result will not contain the Orirank value. For input with n rows, $m=m_1+m_2$ columns for two classes, and k permutations, memory requirements are $2n$ with <code>huge=TRUE</code> instead of $n*k+n*m_1*m_2$.

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.

pdfp	estimated percentage of false positive predictions (pdfp) up to the position of each gene under two identification each
pval	estimated pvalue for each gene being up- and down-regulated
RPs	Original rank-product of each genes for two identification each
RPrank	rank of the rank product of each genes
Orirank	original rank in each comparison, which is used to construct rank product. Not present if huge=TRUE is used.
AveFC	fold change of average expression under class 1 over that under class 2. log-fold change if data is in log scaled, original fold change if data is unlogged.

Note

Percentage of false prediction (pdfp), 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 separate steps, thus two pdfps 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)

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

[topGene RPadvance plotRP](#)

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)

```

RPadvance

Advanced Rank Product Analysis of Microarray

Description

Advance rank product method to identify differentially expressed genes. It is possible to combine data from different studies, e.g. data sets generated at different laboratories.

Usage

```

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

```

Arguments

data	the data set that should be analyzed. Every row of this data set 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 group data, the label for each sample should be 1.
origin	a vector containing the origin labels of the sample. e.g. for the data sets generated at multiple laboratories, the label is the same for samples within one lab and different for samples from different labs.
num.perm	number of permutations used in the calculation of the null density. Default is 'B=100'.
logged	if "TRUE", data has been logged, otherwise set it to "FALSE"
na.rm	if 'FALSE' (default), the NA value will not be used in computing rank. If 'TRUE', the missing values will be replaced by the genewise mean of the non-missing values. Gene with all value missing will be assigned "NA"

gene.names	if "NULL", no gene name will be attached to the estimated percentage of false prediction (pfp).
plot	If "TRUE", plot the estimated pfp verse the rank of each gene
rand	if specified, the random number generator will be put in a reproducible state.
huge	If "TRUE", use an alternative method for computation. Using considerably less memory, this allows the Rank Product to be calculated for larger input data (see details). However, the result will not contain the Orirank value. For input with n rows, $m=m1+m2$ columns for two classes, and k permutations, memory requirements are $2n$ with huge=TRUE instead of $n*k+n*m1*m2$.

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 identification each
pval	estimated pvalue for each gene being up- and down-regulated
RPs	Original rank-product of each genes for two identification each
RPrank	rank of the rank products of each gene in ascending order
Orirank	original ranks in each comparison, which is used to compute rank product. Not present if huge=TRUE is used.
AveFC	fold change of average expression under class 1 over that under class 2, if multiple origin, than averaged across all origin. log-fold change if data is in log scaled, original fold change if data is unlogged.
all.FC	fold change of class 1/class 2 under each origin. log-fold change if data is in log scaled

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 separate 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 different origins.

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

[topGene RP plotRP 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 <- 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
#independently 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$fpf)
head(RP.adv.out$RPs)
head(RP.adv.out$AveFC)
```

Description

Advance rank sum method to identify differentially expressed genes. It is possible to combine data from different studies, e.g. data sets generated at different laboratories.

Usage

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

Arguments

<code>data</code>	the data set that should be analyzed. Every row of this data set must correspond to a gene.
<code>cl</code>	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 group data, the label for each sample should be 1.
<code>origin</code>	a vector containing the origin labels of the sample. e.g. for the data sets generated at multiple laboratories, the label is the same for samples within one lab and different for samples from different labs.
<code>num.perm</code>	number of permutations used in the calculation of the null density. Default is 'B=100'.
<code>logged</code>	if "TRUE", data has been logged, otherwise set it to "FALSE"
<code>na.rm</code>	if 'FALSE' (default), the NA value will not be used in computing rank. If 'TRUE', the missing values will be replaced by the genewise mean of the non-missing values. Gene with all value missing will be assigned "NA"
<code>gene.names</code>	if "NULL", no gene name will be attached to the estimated percentage of false prediction (pfp).
<code>plot</code>	If "TRUE", plot the estimated pfp versus the rank of each gene
<code>rand</code>	if specified, the random number generator will be put in a reproducible state.

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.

<code>pfp</code>	estimated percentage of false positive predictions (pfp) up to the position of each gene under two identification each
<code>pval</code>	estimated pvalue for each gene being up- and down-regulated
<code>RSs</code>	Original rank-sum (average rank) of each genes
<code>RSrank</code>	rank of the rank sum of each gene in ascending order
<code>Orirank</code>	original ranks in each comparison, which is used to compute rank sum
<code>AveFC</code>	fold change of average expression under class 1 over that under class 2, if multiple origin, then averaged across all origin. log-fold change if data is in log scaled, original fold change if data is unlogged.
<code>all.FC</code>	fold change of class 1/class 2 under each origin. log-fold change if data is in log scaled

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 separate 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-organ studies. It is similar to function 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)

Fangxin Hong <fhong@salk.edu>

See Also

[topGene RP plotRP RPadvance](#)

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 *Output Significant Genes*

Description

Identify differentially 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 the function RP, RP.advance or Rsum.advance
cutoff	threshold in pfp 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 the default setting. "pval" used p-value which is less stringent than pfp
num.gene	number of candidate genes of interests, if cutoff is provided, this will be ignored
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

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 list 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

*Topic **datasets**

arab, [2](#)
golub, [2](#)
lymphoma, [3](#)

*Topic **htest**

plotGene, [4](#)
plotRP, [6](#)
RP, [7](#)
RPadvance, [9](#)
RSadvance, [11](#)
topGene, [14](#)

arab, [2](#)

exprSet, [3](#)

golub, [2](#)

lym.exp (lymphoma), [3](#)
lymphoma, [3](#)

plotGene, [4](#)
plotRP, [6](#), [8](#), [11](#), [13](#), [15](#)

RP, [5](#), [6](#), [7](#), [11](#), [13](#), [15](#)
RPadvance, [5](#), [6](#), [8](#), [9](#), [13](#), [15](#)
RSadvance, [5](#), [6](#), [11](#), [11](#), [15](#)

topGene, [5](#), [6](#), [8](#), [11](#), [13](#), [14](#)