Package 'TCC'

November 12, 2026

Type Package

Title TCC: Differential expression analysis for tag count data with robust normalization strategies

Version 1.52.0

Author Jianqiang Sun, Tomoaki Nishiyama, Kentaro Shimizu, and Koji Kadota

Maintainer Jianqiang Sun <sun@bitdessin.dev>, Tomoaki Nishiyama <tomoakin@staff.kanazawa-u.ac.jp>

Description This package provides a series of functions for performing differential expression analysis from RNA-seq count data using robust normalization strategy (called DEGES). The basic idea of DEGES is that potential differentially expressed genes or transcripts (DEGs) among compared samples should be removed before data normalization to obtain a well-ranked gene list where true DEGs are top-ranked and non-DEGs are bottom ranked. This can be done by performing a multi-step normalization strategy (called DEGES for DEG elimination strategy). A major characteristic of TCC is to provide the robust normalization methods for several kinds of count data (two-group with or without replicates, multi-group/multi-factor, and so on) by virtue of the use of combinations of functions in depended packages.

Depends R (>= 3.0), methods, DESeq2, edgeR, ROC

Suggests RUnit, BiocGenerics

License GPL-2

Copyright Authors listed above

biocViews ImmunoOncology, Sequencing, DifferentialExpression, RNASeq

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

git_branch RELEASE_3_23

git_last_commit 6c302b8

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

arab . . . 2
calcAUCValue . . . 3
calcNormFactors . . . 4
clusterSample . . . 7
estimateDE . . . 8
filterLowCountGenes . . . 10
getNormalizedData . . . 11
getResult . . . 12
hypoData . . . 13
hypoData_mg . . . 14
hypoData_ts . . . 15
makeFCMatrix . . . 16
nakai . . . 17
plot . . . 17
plotFCPseudocolor . . . 19
ROKU . . . 20
simulateReadCounts . . . 22
TCC . . . 25
TCC-class . . . 26
WAD . . . 27

Index **29**

`arab` *Arabidopsis RNA-Seq data set*

Description

This dataset was imported from NBPSeq package and the following explanation is verbatim copy of their explanation:

An RNA-Seq dataset from a pilot study of the defense response of *Arabidopsis* to infection by bacteria. We performed RNA-Seq experiments on three independent biological samples from each of the two treatment groups. The matrix contains the frequencies of RNA-Seq reads mapped to genes in a reference database. Rows correspond to genes and columns correspond to independent biological samples.

Usage

```
data(arab)
```

Format

A 26222 by 6 matrix of RNA-Seq read frequencies.

Details

This dataset was imported from NBPSeq package and the following explanation is verbatim copy of their explanation:

We challenged leaves of *Arabidopsis* with the defense-eliciting Δ*hrcC* mutant of *Pseudomonas syringae* pathovar *tomato* DC3000. We also infiltrated leaves of *Arabidopsis* with 10mM MgCl2 as a mock inoculation. RNA was isolated 7 hours after inoculation, enriched for mRNA and prepared for RNA-Seq. We sequenced one replicate per channel on the Illumina Genome Analyzer (http://www.illumina.com). The length of the RNA-Seq reads can vary in length depending on user preference and the sequencing instrument. The dataset used here are derived from a 36-cycle sequencing reaction, that we trimmed to 25mers. We used an in-house computational pipeline to process, align, and assign RNA-Seq reads to genes according to a reference database we developed for *Arabidopsis*.

References

Di Y, Schafer DW, Cumbie JS, and Chang JH (2011): "The NBP Negative Binomial Model for Assessing Differential Gene Expression from RNA-Seq", Statistical Applications in Genetics and Molecular Biology, 10 (1).

Examples

```
data(arab)
```

`calcAUCValue` *Calculate AUC value from a TCC-class object*

Description

This function calculates AUC (Area under the ROC curve) value from a TCC-class object for simulation study.

Usage

```
calcAUCValue(tcc, t = 1)
```

Arguments

`tcc`	TCC-class object having values in both `stat$rank` and `simulation$trueDEG` fields.
`t`	numeric value (between 0 and 1) specifying the FPR (i.e., the x-axis of ROC curve). AUC value is calculated from 0 to `t`. The default is 1.

Details

This function is generally used after the estimateDE function that estimates p-values (and the derivatives such as the q-values and the ranks) for individual genes based on the statistical model for differential expression (DE) analysis. In case of the simulation analysis, we know which genes are DEGs or non-DEGs in advance and the information is stored in the simulation$trueDEG field of the TCC-class object tcc (i.e., tcc$simulation$trueDEG). The calcAUCValue function calculates the AUC value between the ranked gene list obtained by the estimateDE function and the truth obtained by the simulateReadCounts function. A well-ranked gene list should have a high AUC value (i.e., high sensitivity and specificity).

Value

numeric scalar.

Examples

```
# Analyzing a simulation data for comparing two groups
# (G1 vs. G2) with biological replicates.
# the first 200 genes are DEGs, where 180 are up-regulated in G1.
# The DE analysis is performed by an exact test in edgeR coupled
# with the DEGES/edgeR normalization factors.
tcc <- simulateReadCounts(Ngene = 1000, PDEG = 0.2,
                          DEG.assign = c(0.9, 0.1),
                          DEG.foldchange = c(4, 4),
                          replicates = c(3, 3))
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
tcc <- estimateDE(tcc, test.method = "edger", FDR = 0.1)
calcAUCValue(tcc)
```

---

calcNormFactors *Calculate normalization factors*

Description

This function calculates normalization factors using a specified multi-step normalization method from a TCC-class object. The procedure can generally be described as the $STEP1 - (STEP2 - STEP3)_n$ pipeline.

Usage

```
## S4 method for signature 'TCC'
calcNormFactors(tcc, norm.method = NULL, test.method = NULL,
                iteration = TRUE,  FDR = NULL, floorPDEG = NULL,
                increment = FALSE, ...)
```

Arguments

tcc	TCC-class object.
norm.method	character specifying a normalization method used in both the $STEP1$ and $STEP3$. Possible values are "tmm" for the TMM normalization method implemented in the edgeR package, "edger" (same as "tmm"), and "deseq2" for the method implemented in the DESeq2 package. The default is "tmm".
test.method	character specifying a method for identifying differentially expressed genes (DEGs) used in $STEP2$: one of "edger", "deseq2", "bayseq", "voom" and "wad". See the "Details" filed in estimateDE for detail. The default is "edger".
iteration	logical or numeric value specifying the number of iteration (n) in the proposed normalization pipeline: the $STEP1-(STEP2-STEP3)_n$ pipeline. If FALSE or 0 is specified, the normalization pipeline is performed only by the method in $STEP1$. If TRUE or 1 is specified, the three-step normalization pipeline is performed. Integers higher than 1 indicate the number of iteration in the pipeline.
FDR	numeric value (between 0 and 1) specifying the threshold for determining potential DEGs after $STEP2$.
floorPDEG	numeric value (between 0 and 1) specifying the minimum value to be eliminated as potential DEGs before performing $STEP3$.
increment	logical value. if increment = TRUE, the DEGES pipeline will perform again from the current iterated result.
...	arguments to identify potential DEGs at $STEP2$. See the "Arguments" field in estimateDE for details.

Details

The calcNormFactors function is the main function in the TCC package. Since this pipeline employs the DEG identification method at $STEP2$, our multi-step strategy can eliminate the negative effect of potential DEGs before the second normalization at $STEP3$. To fully utilize the DEG elimination strategy (DEGES), we strongly recommend not to use iteration = 0 or iteration = FALSE. This function internally calls functions implemented in other R packages according to the specified value.

- norm.method = "tmm"
  The calcNormFactors function implemented in edgeR is used for obtaining the TMM normalization factors at both $STEP1$ and $STEP3$.
- norm.method = "deseq2"
  The estimateSizeFactors function implemented in DESeq2 is used for obetaining the size factors at both $STEP1$ and $STEP3$. The size factors are internally converted to normalization factors that are comparable to the TMM normalization factors.

Value

After performing the calcNormFactors function, the calculated normalization factors are populated in the norm.factors field (i.e., tcc$norm.factors). Parameters used for DEGES normalization (e.g., potential DEGs identified in $STEP2$, execution times for the identification, etc.) are stored in the DEGES field (i.e., tcc$DEGES) as follows:

`iteration` the iteration number n for the $STEP1 - (STEP2 - STEP3)_n$ pipeline.

`pipeline` the DEGES normalization pipeline.

`threshold` it stores (i) the type of threshold (`threshold$type`), (ii) the threshold value (`threshold$input`), and (iii) the percentage of potential DEGs actually used (`threshold$PDEG`). These values depend on whether the percentage of DEGs identified in $STEP2$ is higher or lower to the value indicated by `floorPDEG`. Consider, for example, the execution of `calcNormFactors` function with "`FDR = 0.1` and `floorPDEG = 0.05`". If the percentage of DEGs identified in $STEP2$ satisfying `FDR = 0.1` was 0.14 (i.e., higher than the `floorPDEG` of 0.05), the values in the threshold fields will be `threshold$type = "FDR"`, `threshold$input = 0.1`, and `threshold$PDEG = 0.14`. If the percentage (= 0.03) was lower than the predefined `floorPDEG` value of 0.05, the values in the threshold fields will be `threshold$type = "floorPDEG"`, `threshold$input = 0.05`, and `threshold$PDEG = 0.05`.

`potDEG` numeric binary vector (0 for non-DEG or 1 for DEG) after the evaluation of the percentage of DEGs identified in $STEP2$ with the predefined `floorPDEG` value. If the percentage (e.g., 2%) is lower than the `floorPDEG` value (e.g., 17%), 17% of elements become 1 as DEG.

`prePotDEG` numeric binary vector (0 for non-DEG or 1 for DEG) before the evaluation of the percentage of DEGs identified in $STEP2$ with the predefined `floorPDEG` value. Regardless of the `floorPDEG` value, the percentage of elements with 1 is always the same as that of DEGs identified in $STEP2$.

`execution.time` computation time required for normalization.

Examples

```
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)

# Calculating normalization factors using the DEGES/edgeR method
# (the TMM-edgeR-TMM pipeline).
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
tcc$norm.factors

# Calculating normalization factors using the iterative DEGES/edgeR method
# (iDEGES/edgeR) with n = 3.
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 3, FDR = 0.1, floorPDEG = 0.05)
tcc$norm.factors
```

clusterSample *Perform hierarchical clustering for samples from expression data*

Description

This function performs hierarchical clustering for samples (tissues or columns) from expression data.

Usage

```
clusterSample(data, dist.method = "spearman", hclust.method = "average",
              unique.pattern = TRUE)
```

Arguments

data	numeric matrix or data frame containing expression data (count data or microarray data), where each row indicates the gene (or transcript or probeset ID), each column indicates the sample (or library), and each cell indicates the expression value (i.e., number of counts or signal intensity) of the gene in the sample.
dist.method	character string specifying a type for correlation coefficient ("spearman" or "pearson") used as distance. The default is "spearman". The hierarchical clustering is performed using the distance (i.e., 1 - "spearman" correlation coefficient, by default).
hclust.method	character string specifying an agglomeration method used in hclust function: "ward", "single", "complete", "average", "mcquitty", "median" or "centroid". The default is "average".
unique.pattern	logical. If FALSE, the input expression data are directly applied for clustering. If TRUE (default), the input data only having unique expression patterns are applied.)

Value

An object of class hclust which describes the tree produced by the clustering process. See hclust for details.

Examples

```
# Perform sample clustering with default options.
data(hypoData)
hc <- clusterSample(hypoData)
plot(hc)

# Obtain the same result using the 'unique.pattern = FALSE' option.
data(hypoData)
keep <- as.logical(rowSums(hypoData) > 0)
data <- unique(hypoData[keep, ])
hc <- clusterSample(data, unique.pattern = FALSE)
plot(hc)
```

estimateDE *Estimate degrees of differential expression (DE) for individual genes*

Description

This function calculates p-values (or the related statistics) for identifying differentially expressed genes (DEGs) from a TCC-class object. estimateDE internally calls a specified method implemented in other R packages.

Usage

```
estimateDE(tcc, test.method, FDR, paired,
           full, reduced,                  # for DESeq2
           design, contrast,               # for edgeR, DESeq2, voom
           coef,                           # for edgeR, voom
           group, cl,                      # for baySeq
           samplesize,                     # for baySeq, SAMseq
           logged, floor,                  # for WAD
           ...
)
```

Arguments

tcc	TCC-class object.
test.method	character string specifying a method for identifying DEGs: one of "edger", "deseq2", "bayseq", "voom", and "wad". See the "Details" field for detail. The default is "edger".
FDR	numeric value (between 0 and 1) specifying the threshold for determining DEGs.
paired	logical. If TRUE, the input data are regarded as (two-group) paired samples. If FALSE, the input data are regarded as unpaired samples. The default is FALSE.
full	a formula for creating full model described in DESeq2. The right hand side can involve any column of tcc$group is used as the model frame. See the nbinomLRT function in DESeq2 for details.
reduced	a formula for creating reduced model described in DESeq2. The right hand side can involve any column of tcc$group is used as the model frame. See the nbinomLRT function in DESeq2 for details.
design	the argument is used in edgeR, voom (limma) and DESeq2. For edgeR and voom, it should be the numeric matrix giving the design matrix for the generalized linear model. See the glmFit function in edgeR or the lmFit function in limma for details. For DESeq2, it should be a formula specifying the design of the experiment. See the DESeqDataSet function in DESeq2 for details.
contrast	the argument is used in edgeR and DESeq2. For edgeR, numeric vector specifying a contrast of the linear model coefficients to be tested equal to zero. See the glmLRT function in edgeR for details. For DESeq2, the argument is same to contrast which used in DESeq2 package to retrive the results from Wald test. See the results function in DESeq2 for details.

coef	integer or character vector indicating which coefficients of the linear model are to be tested equal to zero. See the glmLRT function in edgeR for details.
group	numeric or character string identifying the columns in the tcc$group for analysis. See the group argument of topCounts function in baySeq for details.
cl	snow object when using multi processors if test.method = "bayseq" is specified. See the getPriors.NB function in baySeq for details.
samplesize	integer specifying the sample size for estimating the prior parameters if test.method = "bayseq" (defaults to 10000).
logged	logical. If TRUE, the input data are regarded as log2-transformed. If FALSE, the log2-transformation is performed after the floor setting. The default is logged = FALSE. Ignored if test.method is not "wad".
floor	numeric scalar (> 0) specifying the floor value for taking logarithm. The default is floor = 1, indicating that values less than 1 are replaced by 1. Ignored if logged = TRUE. Ignored if test.method is not "wad".
...	further paramenters.

Details

estimaetDE function is generally used after performing the calcNormFactors function that calculates normalization factors. estimateDE constructs a statistical model for differential expression (DE) analysis with the calculated normalization factors and returns the p-values (or the derivatives). The individual functions in other packages are internally called according to the specified test.method parameter.

- test.method = "edger"
  There are two approaches (i.e., exact test and GLM) to identify DEGs in edgeR. The two approches are implmented in TCC. As a default, the exact test approach is used for two-group data, and GLM approach is used for multi-group or multi-factor data. However, if design and the one of coef or contrast are given, the GLM approach will be used for two-group data.
  If the exact test approach is used, estimateCommonDisp, estimateTagwiseDisp, and exactTest are internally called.
  If the GLM approach is used, estimateGLMCommonDisp, estimateGLMTrendedDisp, estimateGLMTagwiseDisp, glmFit, and glmLRT are internally called.
- test.method = "deseq2"
  estimateDispersions, and nbinomWaldTest are internally called for identifying DEGs. However, if full and reduced are given, the nbinomLRT will be used.
- test.method = "bayseq"
  getPriors.NB and getLikelihoods in baySeq are internally called for identifying DEGs. If paired = TRUE, getPriors and getLikelihoods in baySeq are used.
- test.method = "voom"
  voom, lmFit, and eBayes in limma are internally called for identifying DEGs.
- test.method = "wad"
  The WAD implemented in TCC is used for identifying DEGs. Since WAD outputs test statistics instead of p-values, the tcc$stat$p.value and tcc$stat$q.value are NA. Alternatively, the test statistics are stored in tcc$stat$testStat field.

Value

A TCC-class object containing following fields:

stat$p.value	numeric vector of p-values.
stat$q.value	numeric vector of q-values calculated based on the p-values using the p.adjust function with default parameter settings.
stat$testStat	numeric vector of test statistics if "wad" is specified.
stat$rank	gene rank in order of the p-values or test statistics.
estimatedDEG	numeric vector consisting of 0 or 1 depending on whether each gene is classified as non-DEG or DEG. The threshold for classifying DEGs or non-DEGs is preliminarily given as the FDR argument.

Examples

```
# Analyzing a simulation data for comparing two groups
# (G1 vs. G2) with biological replicates
# The DE analysis is performed by an exact test in edgeR coupled
# with the DEGES/edgeR normalization factors.
# For retrieving the summaries of DE results, we recommend to use
# the getResult function.
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
tcc <- estimateDE(tcc, test.method = "edger", FDR = 0.1)
head(tcc$stat$p.value)
head(tcc$stat$q.value)
head(tcc$estimatedDEG)
result <- getResult(tcc)
```

---

filterLowCountGenes *Filter genes from a TCC-class object*

Description

This function takes a TCC object and returns a new TCC object without genes having low count tags across samples. The threshold is configurable with low.count parameter.

Usage

```
filterLowCountGenes(tcc, low.count = 0)
```

Arguments

tcc	TCC-class object.
low.count	numeric value (>= 0) specifying the threshold for filtering genes. The higher value indicates the more numbers of genes to be filtered out.

Value

TCC-class object consisting of genes whose total counts across samples is higher than the `low.count` value.

Examples

```
# Filtering genes with zero counts across samples (default) from
# a hypothetical count dataset that originally has 1,000 genes.
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
dim(tcc$count)
tcc <- filterLowCountGenes(tcc)
dim(tcc$count)

# Filtering genes with 10 counts across samples from hypoData.
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
dim(tcc$count)
tcc <- filterLowCountGenes(tcc, 10)
dim(tcc$count)
```

---

getNormalizedData *Obtain normalized count data*

Description

This function generates normalized count data from both original count data and calculated normalization factors.

Usage

```
getNormalizedData(tcc)
```

Arguments

`tcc`	TCC-class object.

Details

This function is generally used after the `calcNormFactors` function that calculates normalization factors. The normalized data is calculated using both the original count data stored in the `count` field and the normalization factors stored in the `norm.factors` field in the TCC-class object.

Value

A numeric matrix containing normalized count data.

Examples

```
# Note that the hypoData has non-DEGs at 201-1000th rows.
nonDEG <- 201:1000
data(hypoData)
summary(hypoData[nonDEG, ])
group <- c(1, 1, 1, 2, 2, 2)

# Obtaining normalized count data after performing the
# DEGES/edgeR normalization method, i.e., DEGES/edgeR-normalized data.
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
normalized.count <- getNormalizedData(tcc)
summary(normalized.count[nonDEG, ])

# Obtaining normalized count data after performing the TMM normalization
# method (Robinson and Oshlack, 2010), i.e., TMM-normalized data.
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", iteration = 0)
normalized.count <- getNormalizedData(tcc)
summary(normalized.count[nonDEG, ])
```

`getResult` *Obtain the summaries of results after the differential expression analysis*

Description

This function is generally used after the `estimateDE` function. It retrieves the summaries of differential expression (DE) results from TCC-class object. The retrieved information includes p-values, q-values, coordinates of M-A plot (i.e., M and A values), and so on.

Usage

```
getResult(tcc, sort = FALSE, ...)
```

Arguments

`tcc`	TCC-class object
`sort`	logical. If `TRUE`, the retrieved results are sorted in order of the `stat$rank` field in the TCC-class object. If `FALSE`, the results are retrieved by the original order.
`...`	further arguments for calculating the coordinates of M-A plot. See `plot` for details.

Value

A data frame object containing following fields:

gene_id	character vector indicating the id of the count unit, usually gene.
a.value	numeric vector of average expression level on log2 scale (i.e., A-value) for each gene across the compared two groups. It corresponds to the x coordinate in the M-A plot.
m.value	numeric vector of fold-change on $\log_2$ scale (i.e., M-value) for each gene between the two groups compared. It corresponds to the y coordinate in the M-A plot.
p.value	numeric vector of p-value.
q.value	numeric vector of q-value calculated based on the p-value using the p.adjust function with default parameter settings.
rank	numeric vector of gene rank in order of the p-values.
estimatedDEG	numeric vector consisting of 0 or 1 depending on whether each gene is classified as non-DEG or DEG. The threshold for classifying DEGs or non-DEGs is preliminarily given when performing estimateDE.

Examples

```
# Obtaining DE results by an exact test in edgeR coupled with
# the DEGES/edgeR normalization factors.
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
tcc <- estimateDE(tcc, test.method = "edger", FDR = 0.1)
result <- getResult(tcc, sort = TRUE)
head(result)
```

---

hypoData *A simulation dataset for comparing two-group tag count data, focusing on RNA-seq*

Description

A simulation dataset, consisting of 1,000 rows (or genes) and 6 columns (or independent biological samples).

Usage

```
data(hypoData)
```

Format

hypoData is a matrix of dimension 1,000 times 6.

Details

This package typically start the differential expression analysis with a count table matrix such as hypoData where each row indicates the gene (or transcript), each column indicates the sample (or library), and each cell indicates the number of counts to the gene in the sample. The first three columns are produced from biological replicates of, for example, Group 1 and the remaining columns are from Group 2; i.e., G1_rep1, G1_rep2, G1_rep3 vs. G2_rep1, G2_rep2, G2_rep3. This data is generated by the simulateReadCounts function with default parameter settings. The first 200 genes are differentially expressed in the two groups. Of these, the first 180 genes are expressed at a higher level in Group 1 (G1) and the remaining 20 genes are expressed at a higher level in G2. Accordingly, the 201-1000th genes are not differentially expressed (non-DEGs). The levels of differential expression (DE) are four-fold in both groups.

Examples

```
# The 'hypoData' is generated by following commands.
tcc <- simulateReadCounts(Ngene = 1000, PDEG = 0.2,
                          DEG.assign = c(0.9, 0.1),
                          DEG.foldchange = c(4, 4),
                          replicates = c(3, 3))
hypoData <- tcc$count
```

---

hypoData_mg *A simulation dataset for comparing three-group tag count data, focusing on RNA-seq*

Description

A simulation dataset, consisting of 1,000 rows (or genes) and 9 columns (or independent biological samples)

Usage

```
data(hypoData_mg)
```

Format

hypoData_mg is a matrix of dimension 1,000 times 9.

Details

The hypoData_mg, a matrix object, is a simulation dataset which consists of 1,000 rows (genes) and 9 columns (samples). Each cell of matrix indicates the number of counts to the gene in the sample. The first three columns are produced from biological replicates of, for example, Group 1, the next three columns are from Group2 and the remaining columns are from Group 3; i.e., G1_rep1, G1_rep2, G1_rep3 vs. G2_rep1, G2_rep2, G2_rep3 vs. G3_rep1, G3_rep2, G3_rep3. This data is generated by the simulateReadCounts function with the following parameters (see Examples). The first 200 genes are differentially expressed among the three groups. Of these, the first 140 genes are expressed at a higher level only in Group 1 (G1), the next 40 genes are expressed at a higher level only in G2 and the last 20 genes are expressed at a higher level only in G3. Accordingly, the 201-1000th genes are not differentially expressed (non-DEGs). The levels of differential expression (DE) are four-fold in each group.

Examples

```
# The 'hypoData_mg' is generated by following commands.
tcc <- simulateReadCounts(Ngene = 1000, PDEG = 0.2,
                          DEG.assign = c(0.7, 0.2, 0.1),
                          DEG.foldchange = c(4, 4, 4),
                          replicates = c(3, 3, 3))
hypoData_mg <- tcc$count
```

---

hypoData_ts *A sample microarray data for detecting tissue-specific patterns.*

Description

A hypothetical micoarray data consisting of eight rows (genes) and ten columns (tissues). The expression patterns are quite similar to those in figure 1 in Kadota et al., 2006.

Usage

```
data(hypoData_ts)
```

Format

hypoData_ts is a matrix of dimension eight times ten.

Details

The hypoData_ts is designed for explaining the performance of ROKU that identify tissue-specific expression patterns. The hypoData_ts contains a total of eight genes having various expression patterns across ten tissues: (1) 'up-type' genes selectively over-expressed in a small number of tissues but unexpressed ("gene1"), slightly expressed ("gene3"), and moderately expressed ("gene4"), (2) 'down-type' genes selectively under-expressed ("gene5"), and (3) 'mixed-type' genes selectively over- and under-expressed in some tissues ("gene6"). The other genes are not tissue-specific genes (i.e., "gene2", "gene7", and "gene8").

References

Kadota K, Ye J, Nakai Y, Terada T, Shimizu K: ROKU: a novel method for identification of tissue-specific genes. BMC Bioinformatics 2006, 7: 294.

Examples

```
data(hypoData_ts)
```

`makeFCMatrix` *Generate the fold change matrix for simulating count data*

Description

Generate the fold change matrix for simulating count data under the Gammma distribution

Usage

```
makeFCMatrix(Ngene = 10000, PDEG = 0.20, DEG.assign = NULL,
             replicates = NULL, fc.params = NULL)
```

Arguments

`Ngene`	numeric scalar specifying the number of genes.
`PDEG`	numeric scalar specifying the proportion of differentially expressed genes (DEGs).
`DEG.assign`	numeric vector specifying the proportion of DEGs up- or down-regulated in individual groups to be compared. The number of elements should be the same as that of `replicates` if `replicates` is specified. The indication of `replicates` means a single-factor experimental design.
`replicates`	numeric vector indicating the numbers of (biological) replicates for individual groups compared. Ignored if `group` is specified.
`fc.params`	foldchanges of DEGs are randomly sampled from $f+\Gamma(a,b)$ where a is a shape and b is a scale of Gamma distribution. The default values are $f = 1.2$, $a = 2$, and $b = 0.5$

Details

`makeFCMatrix` function is a function for generating the foldchanges of DEGs. The foldchanges are randomly sampled from $f + \Gamma(a, b)$ where a is a shape and b is a scale of Gamma distribution.

Value

matrix

Examples

```
fc.matrix <- makeFCMatrix()
```

nakai *DNA microarray data set*

Description

This is a log2-transformed two-group microarray (Affymetrix GeneChip) data consisting of 31,099 probesets. A total of eight samples were taken from the rat liver: the first four samples are fed and the last fours are 24-hour fasted. The original data can be obtained from NCBI Gene Expression Omnibus (GEO) with GSE7623. This is a subset.

Usage

```
data(nakai)
```

Format

nakai is a matrix of 31,099 rows (probesets) and 8 columns (samples or tissues).

References

Nakai Y, Hashida H., Kadota K, Minami M, Shimizu K, Matsumoto I, Kato H, Abe K., Up-regulation of genes related to the ubiquitin-proteasome system in the brown adipose tissue of 24-h-fasted rats. Bioscience Biotechnology and Biochemistry 2008, 72(1):139-148.

Examples

```
data(nakai)
```

plot *Plot a log fold-change versus log average expression (so-called M-A plot)*

Description

This function generates a scatter plot of log fold-change (i.e., $M = \log_2 G2 - \log_2 G1$ on the y-axis between Groups 1 vs. 2) versus log average expression (i.e., $A = (\log_2 G1 + \log_2 G2)/2$ on the x-axis) using normalized count data.

Usage

```
## S3 method for class 'TCC'
plot(x, FDR = NULL, median.lines = FALSE, floor = 0,
     group = NULL, col = NULL, col.tag = NULL, normalize = TRUE, ...)
```

Arguments

x	TCC-class object.
FDR	numeric scalar specifying a false discovery rate (FDR) threshold for determining differentially expressed genes (DEGs)
median.lines	logical. If TRUE, horizontal lines specifying the median M values for non-DEGs (black) and DEGs (red) are drawn.
floor	numeric scalar specifying a threshold for adjusting low count data.
group	numeric vector consists two elements for specifying what two groups should be drawn when data contains more than three groups.
col	vector specifying plotting color.
col.tag	numeric vector spacifying the index of col for coloring the points of the genes.
normalize	logical. If FALSE, the coordinates of M-A plot are calculated from the raw data.
...	further graphical arguments, see plot.default.

Details

This function generates roughly three different M-A plots depending on the conditions for TCC-class objects. When the function is performed just after the new method, all the genes (points) are treated as non-DEGs (the default is black; see Example 1). The simulateReadCounts function followed by the plot function can classify the genes as *true* non-DEGs (black), *true* DEGs. (see Example 2). The estimateDE function followed by the plot function generates *estimated* DEGs (magenta) and the remaining *estimated* non-DEGs (black).

Genes with normalized counts of 0 in any one group cannot be plotted on the M-A plot because those M and A values cannot be calculated (as $\log 0$ is undefined). Those points are plotted at the left side of the M-A plot, depending on the minimum A (i.e., log average expression) value. The x coordinate of those points is the minimum A value minus one. The y coordinate is calculated as if the zero count was the minimum observed non zero count in each group.

Value

A scatter plot to the current graphic device.

Examples

```
# Example 1.
# M-A plotting just after constructing the TCC class object from
# hypoData. In this case, the plot is generated from hypoData
# that has been scaled in such a way that the library sizes of
# each sample are the same as the mean library size of the
# original hypoData. Note that all points are in black. This is
# because the information about DEG or non-DEG for each gene is
# not indicated.
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
plot(tcc)
```

```
normalized.count <- getNormalizedData(tcc)
colSums(normalized.count)
colSums(hypoData)
mean(colSums(hypoData))


# Example 2.
# M-A plotting of DEGES/edgeR-normalized simulation data.
# It can be seen that the median M value for non-DEGs approaches
# zero. Note that non-DEGs are in black, DEGs are in red.
tcc <- simulateReadCounts()
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
plot(tcc, median.lines = TRUE)


# Example 3.
# M-A plotting of DEGES/edgeR-normalized hypoData after performing
# DE analysis.
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
tcc <- estimateDE(tcc, test.method = "edger", FDR = 0.1)
plot(tcc)

# Changing the FDR threshold
plot(tcc, FDR = 0.7)
```

`plotFCPseudocolor` *Create a pseudo-color image of simulation data*

Description

This function creates a pseudo-color image of simulation data regarding the number of differentially expressed genes (DEGs) and the breakdowns for individual groups from a TCC-class object.

Usage

```
plotFCPseudocolor(tcc, main, xlab, ylab)
```

Arguments

`tcc`	TCC-class object.
`main`	character string indicating the plotting title.
`xlab`	character string indicating the *x*-label title.
`ylab`	character string indicating the *y*-label title.

Details

This function should be used after the simulateReadCounts function that generates simulation data with arbitrary defined conditions. The largest log fold-change (FC) values are in magenta and no-changes are in white.

Examples

```
# Generating a simulation data for comparing two groups
# (G1 vs. G2) with biological replicates.
# the first 200 genes are DEGs, where 180 are up in G1.
tcc <- simulateReadCounts(Ngene = 1000, PDEG = 0.2,
                          DEG.assign = c(0.9, 0.1),
                          DEG.foldchange = c(4, 4),
                          replicates = c(3, 3))
plotFCPseudocolor(tcc)

# Generating a simulation data for comparing three groups
# (G1 vs. G2 vs. G3) with biological replicates.
# the first 300 genes are DEGs, where the 70%, 20%, and 10% are
# up-regulated in G1, G2, G3, respectively. The levels of DE are
# 3-, 10, and 6-fold in individual groups.
tcc <- simulateReadCounts(Ngene = 1000, PDEG = 0.3,
                          DEG.assign = c(0.7, 0.2, 0.1),
                          DEG.foldchange = c(3, 10, 6),
                          replicates = c(3, 3, 3))
plotFCPseudocolor(tcc)
```

ROKU *detect tissue-specific (or tissue-selective) patterns from microarray data with many kinds of samples*

Description

ROKU is a method for detecting tissue-specific (or tissue-selective) patterns from gene expression data for many tissues (or samples). ROKU (i) ranks genes according to their overall tissue-specificity using Shannon entropy after data processing and (ii) detects tissues specific to each gene if any exist using an Akaike's information criterion (AIC) procedure.

Usage

```
ROKU(data, upper.limit = 0.25, sort = FALSE)
```

Arguments

data	numeric matrix or data frame containing microarray data (on log2 scale), where each row indicates the gene or probeset ID, each column indicates the tissue, and each cell indicates a (log2-transformed) expression value of the gene in the tissue. Numeric vector can also be accepted for a single gene expression vector.

`upper.limit`	numeric value (between 0 and 1) specifying the maximum percentage of tissues (or samples) as outliers to each gene.
`sort`	logical. If `TRUE`, results are sorted in descending order of the entropy scores.

Details

As shown in Figure 1 in the original study of ROKU (Kadota et al., 2006), Shannon entropy H of a gene expression vector $(x_1, x_2, ..., x_N)$ for N tissues can range from zero to $log_2 N$, with the value 0 for genes expressed in a single tissue and $log_2 N$ for genes expressed uniformly in all the tissues. Researchers therefore rely on the low entropy score for the identification of tissue-specific patterns. However, direct calculation of the entropy for raw gene expression vector works well only for detecting tissue-specific patterns when over-expressed in a small number of tissues but unexpressed or slightly expressed in others: The H scores of tissue-specific patterns such as $(8, 8, 2, 8, 8, 8, 8, 8, 8, 8)$ for the 3rd tissue-specific down-regulation (see the Figure 1e) are close to the maximum value ($log_2 N = 3.32$ when $N = 10$) and cannot identify such patterns as tissue-specific. To detect various kinds of tissue-specific patterns by low entropy score, ROKU processes the original gene expression vector and makes a new vector $(x_{1'}, x_{2'}, ..., x_{N'})$. The data processing is done by subtracting the one-step Tukey biweight and by taking the absolute value. In case of the above example, ROKU calculates the H score from the processed vector $(0, 0, 6, 0, 0, 0, 0, 0, 0, 0)$, giving very low score (from $H = 3.26$ before processing to $H' = 0$ after processing). A major characteristic of ROKU is, therefore, to be able to rank various tissue-specific patterns by using the modified entropy scores.

Note that the modified entropy does not explain to which tissue a gene is specific, only measuring the degree of overall tissue specificity of the gene. ROKU employs an AIC-based outlier detection method (Ueda, 1996). Consider, for example, a hypothetical mixed-type of tissue-selective expression pattern $(1.2, 5.1, 5.2, 5.4, 5.7, 5.9, 6.0, 6.3, 8.5, 8.8)$ where we imagine a total of three tissues are specific (down-regulated in tissue1; up-regulated in tissues 9 and 10). The method first normalize the expression values by subtracting the mean and dividing by the standard deviation (i.e., z-score transformation), then sorted in order of increasing magnitude by $(-2.221, -0.342, -0.294, -0.198, -0.053, 0.043, 0.092, 0.236, 1.296, 1.441)$. The method evaluates various combinations of outlier candidates starting from both sides of the values: model1 for non-outlier, model2 for one outlier for high-side, model3 for two outliers for high-side, ..., model$x$ for one outlier for down-side, ..., modely for two outliers for both up- and down sides, and so on. Then, it calculates AIC-like statistic (called U) for each combination of model and search the best combination that achieves the lowest U value and is termed the minimum AIC estimate (MAICE). Since the upper.limit value corresponds to the maximum number of the outlier candidates, it decides the number of combinations. The AIC-based method output a vector (1 for up-regulated outliers, -1 for down-regulated outliers, and 0 for non-outliers) that corresponds to the input vector. For example, the method outputs a vector $(-1, 0, 0, 0, 0, 0, 0, 0, 1, 1)$ when using `upper.limit = 0.5` and $(-1, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ when using `upper.limit = 0.25` (as default). See the Kadota et al., 2007 for detailed discussion about the effect of different parameter settings.

Value

A list containing following fields:

`outlier`	A numeric matrix when the input `data` are data frame or matrix. A numeric vector when the input `data` are numeric vector. Both matrix or vector consist of

1, -1, and 0: 1 for over-expressed outliers, -1 for under-expressed outliers, and 0 for non-outliers.

`H` A numeric vector when the input `data` are data frame or matrix. A numeric scalar when the input `data` are numeric vector. Both vector or scalar consist of original entropy (H) score(s) calculated from an original gene expression vector.

`modH` A numeric vector when the input `data` are data frame or matrix. A numeric scalar when the input `data` are numeric vector. Both vector or scalar consist of modified entropy (H') score(s) calculated from a processed gene expression vector.

`rank` A numeric vector or scalar consisting of the rank(s) of `modH`.

`Tbw` a numeric vector or scalar consisting of one-step Tukey's biweight as an iteratively reweighted measure of central tendency. This value is in general similar to median value and the same as the output of `tukey.biweight` with default parameter settings in `affy` package. The data processing is done by subtracting this value for each gene expression vector and by taking the absolute value.

References

Kadota K, Konishi T, Shimizu K: Evaluation of two outlier-detection-based methods for detecting tissue-selective genes from microarray data. Gene Regulation and Systems Biology 2007, 1: 9-15.

Kadota K, Ye J, Nakai Y, Terada T, Shimizu K: ROKU: a novel method for identification of tissue-specific genes. BMC Bioinformatics 2006, 7: 294.

Kadota K, Nishimura SI, Bono H, Nakamura S, Hayashizaki Y, Okazaki Y, Takahashi K: Detection of genes with tissue-specific expression patterns using Akaike's Information Criterion (AIC) procedure. Physiol Genomics 2003, 12: 251-259.

Ueda T. Simple method for the detection of outliers. Japanese J Appl Stat 1996, 25: 17-26.

Examples

```
data(hypoData_ts)

result <- ROKU(hypoData_ts)
```

---

simulateReadCounts *Generate simulation data from negative binomial (NB) distribution*

Description

This function generates simulation data with arbitrary defined experimental condition.

Usage

```
simulateReadCounts(Ngene = 10000, PDEG = 0.20, DEG.assign = NULL,
                   DEG.foldchange = NULL, replicates = NULL, group = NULL,
                   fc.matrix = NULL)
```

Arguments

Ngene	numeric scalar specifying the number of genes.
PDEG	numeric scalar specifying the proportion of differentially expressed genes (DEGs).
DEG.assign	numeric vector specifying the proportion of DEGs up- or down-regulated in individual groups to be compared. The number of elements should be the same as that of replicates if replicates is specified. The indication of replicates means a single-factor experimental design. The number of elements in DEG.assign should be the same as the number of columns in DEG.foldchange. Both DEG.foldchange as data frame and group should simultaneously be specified and those indication means a multi-factor experimental design.
DEG.foldchange	numeric vector for single-factor experimental design and data frame for multi-factor experimental design. Both DEG.foldchange as numeric vector and replicates should simultaneously be specified for single-factor experimental design. The i-th element in DEG.foldchange vector indicates the degree of fold-change for Group i. The default is DEG.foldchange = c(4, 4), indicating that the levels of DE are four-fold in both groups. Both DEG.foldchange as data frame and group should simultaneously be specified for multi-factor experimental design. Numeric values in the DEG.foldchange object indicate the degree of fold-change for individual conditions or factors.
replicates	numeric vector indicating the numbers of (biological) replicates for individual groups compared. Ignored if group is specified.
group	data frame specifying the multi-factor experimental design.
fc.matrix	fold change matrix generated by makeFCMatrix for simulating DEGs with the fold-change under un-uniform distributions.

Details

The empirical distribution of read counts used in this function is calculated from a RNA-seq dataset obtained from *Arabidopsis* data (three biological replicates for both the treated and non-treated samples), the arab object, in NBPSeq package (Di et al., 2011). The overall design about the simulation conditions introduced can be viewed as a pseudo-color image by the plotFCPseudocolor function.

Value

A TCC-class object containing following fields:

count	numeric matrix of simulated count data.
group	data frame indicating which group (or condition or factor) each sample belongs to.
norm.factors	numeric vector as a placeholder for normalization factors.
stat	list for storing results after the execution of the calcNormFactors (and estimateDE) function.
estimatedDEG	numeric vector as a placeholder for indicating which genes are up-regulated in particular group compared to the others. The values in this field will be populated after the execution of the estimateDE function.

simulation — list containing four fields: trueDEG, DEG.foldchange, PDEG, and params. The trueDEG field (numeric vector) stores information about DEGs: 0 for non-DEG, 1 for DEG up-regulated in Group 1, 2 for DEG up-regulated in Group 2, and so on. The information for the remaining three fields is the same as those indicated in the corresponding arguments.

Examples

```
# Generating a simulation data for comparing two groups
# (G1 vs. G2) without replicates (single-factor experimental design).
# the levels of DE are 3-fold in G1 and 7-fold in G2.
tcc <- simulateReadCounts(Ngene = 10000, PDEG = 0.2,
                          DEG.assign = c(0.9, 0.1),
                          DEG.foldchange = c(3, 7),
                          replicates = c(1, 1))
dim(tcc$count)
head(tcc$count)
str(tcc$simulation)
head(tcc$simulation$trueDEG)


# Generating a simulation data for comparing three groups
# (G1 vs. G2 vs. G3) with biological replicates
# (single-factor experimental design).
# the first 3000 genes are DEGs, where the 70%, 20%, and 10% are
# up-regulated in G1, G2, G3, respectively. The levels of DE are
# 3-, 10-, and 6-fold in individual groups.
tcc <- simulateReadCounts(Ngene = 10000, PDEG = 0.3,
                          DEG.assign = c(0.7, 0.2, 0.1),
                          DEG.foldchange = c(3, 10, 6),
                          replicates = c(2, 4, 3))
dim(tcc$count)
head(tcc$count)
str(tcc$simulation)
head(tcc$simulation$trueDEG)


# Generating a simulation data consisting of 10,000 rows (i.e., Ngene = 10000)
# and 8 columns (samples) for two-factor experimental design
# (condition and time). The first 3,000 genes are DEGs (i.e., PDEG = 0.3).
# Of the 3,000 DEGs, 40% are differentially expressed in condition (or GROUP) "A"
# compared to the other condition (i.e., condition "B"), 40% are differentially
# expressed in condition (or GROUP) "B" compared to the other condition
# (i.e., condition "A"), and the remaining 20% are differentially expressed at
# "10h" in association with the second factor: DEG.assign = c(0.4, 0.4, 0.2).
# The levels of fold-change are (i) 2-fold up-regulation in condition "A" for
# the first 40% of DEGs, (ii) 4-fold up-regulation in condition "B" for the
# second 40%, and (iii) 0.4- and 0.6-fold up-regulation at "10h" in "A" and
# 5-fold up-regulation at "10h" in "B".

group <- data.frame(
   GROUP = c( "A",  "A",   "A",   "A",  "B",  "B",   "B",   "B"),
```

```
   TIME  = c("2h", "2h", "10h", "10h", "2h", "2h", "10h", "10h")
)
DEG.foldchange <- data.frame(
   FACTOR1 = c(2, 2,   2,   2, 1, 1, 1, 1),
   FACTOR1 = c(1, 1,   1,   1, 4, 4, 4, 4),
   FACTOR2 = c(1, 1, 0.4, 0.6, 1, 1, 5, 5)
)
tcc <- simulateReadCounts(Ngene = 10000, PDEG = 0.3,
                          DEG.assign = c(0.4, 0.4, 0.2),
                          DEG.foldchange = DEG.foldchange,
                          group = group)
tcc
```

TCC *A package for differential expression analysis from tag count data with robust normalization strategies*

Description

This package performs differential expression analysis from transcriptome data that are produced from high-throughput sequencing (HTS) and microarray technologies. A notable feature of this package is to provide robust normalization methods whose strategy is to remove data assigned as potential differentially expressed genes (DEGs) before performing normalization for RNA-seq count data (Kadota et al., 2012; Sun et al., 2013).

Details

TCC is a package for differential expression analysis from transcriptome data produced from RNA-seq and microarray data. This package implements some functions for calculating normalization factors, identifying DEGs, depicting so-called M-A plot, and generating simulation data.

To utilize this package, the count matrix coupled with label information should be stored to a TCC-class object using the new method. All functions, except for two recently added functions (i.e., ROKU and WAD) for microarray data, used in this package require this TCC-class object. Using this object, the calcNormFactors function calculates normalization factors and the estimateDE function estimates the degree of differential expression (DE) for individual genes. The estimated normalization factors obtained by using the calcNormFactors function are used within the statistical model for differential analysis in the estimateDE function. Both two functions internally call functions from other packages (edgeR, baySeq, and EBSeq) when specified. TCC also provides some useful functions: simulateReadCounts for generating simulation data with various experimental designs, plot for depicting a M-A plot, plotFCPseudocolor for depicting a pseudo-color image of simulation condition that the user specified, WAD for identifying DEGs from two-group microarray data (single-factor design), and ROKU for identifying tissue-specific genes from microarray data for many tissues.

See Also

TCC-class

Examples

```
data(hypoData)
group <- c(1, 1, 1, 2, 2, 2)
tcc <- new("TCC", hypoData, group)
show(tcc)
```

TCC-class *A container for storing information used in TCC*

Description

This is a container class for TCC. This class initially contains count data matrix and some information for the analysis of count data. It also provides further fields that are populated during the analysis.

Details

This class is implemented as an R5 reference class. Functions calling such methods copies the object prior to calling the method to keep the semantics of functional programming. This class can be created by the generic `new` function with count data and associated information of experimental design.

The values (defaults to all 1) in the `norm.factors` field will be changed after performing the `calcNormFactors` function. The `DEGES` field stores information related to our DEGES-based normalization pipeline after performing the `calcNormFactors` function. The `stat` and `estimatedDEG` fields store results after performing the `estimateDE` function. The `simulation` field stores parameters used when performing the `simulateReadCounts` function.

Fields

This class contains the following fields:

count numeric matrix containing count data.

gene_id character vector indicating the identifier of the count unit, usually gene.

group data frame indicating information about experimental design.

norm.factors numeric vector containing normalization factors (default to 1).

stat list for storing results after the execution of the `calcNormFactors` and `estimateDE` functions.

estimatedDEG numeric vector as a placeholder for indicating either DEGs (flagged as "1") or non-DEGs (as "0") for individual genes. The values in this field will be populated after the execution of the `estimateDE` function.

simulation list. This field is only used for analyzing simulation data.

DEGES list for storing the information about normalization steps.

Examples

```
tcc <- simulateReadCounts(Ngene = 10000, PDEG = 0.2, DEG.assign = c(0.8, 0.2),
                          DEG.foldchange = c(4, 4), replicates = c(3, 3))

# Check the TCC class object.
tcc

# Check the fields of TCC class object.
names(tcc)
head(tcc$count)

# Check the normalization factors.
tcc <- calcNormFactors(tcc, norm.method = "tmm", test.method = "edger",
                       iteration = 1, FDR = 0.1, floorPDEG = 0.05)
tcc$norm.factors

# Check the p-values and q-values.
tcc <- estimateDE(tcc, test.method = "edger", FDR = 0.1)
tcc

# Compare the breakdowns of estimated DEGs with the truth.
head(tcc$estimatedDEG)
head(tcc$simulation$trueDEG)

# M-A plotting.
plot(tcc)
```

WAD *Calculate WAD statistic for individual genes*

Description

This function performs WAD method to identify differentially expressed genes (DEGs) from two-group gene expression data. A high absolute value for the WAD statistic is evident of a high degree of differential expression.

Usage

```
WAD(data, group, logged = FALSE, floor = 1, sort = FALSE)
```

Arguments

data	numeric matrix or data frame containing count data or microarray data, where each row indicates the gene (or transcript or probeset ID), each column indicates the sample (or library), and each cell indicates the expression value (i.e., number of counts or signal intensity) of the gene in the sample.
group	numeric vector indicating the experimental group for each sample (or library).

logged	logical. If TRUE, the input data are regarded as log2-transformed. If FALSE, the log2-transformation is performed after the floor setting. The default is logged = FALSE.
floor	numeric scalar (> 0) specifying the floor value for taking logarithm. The default is floor = 1, indicating that values less than 1 are replaced by 1. Ignored if logged = TRUE.
sort	logical. If TRUE, the retrieved results are sorted in order of the rank of absolute WAD statistic. If FALSE, the results are retrieved by the original order.

Value

A numeric vector of WAD statistic for individual genes

References

Kadota K, Nakai Y, Shimizu K: A weighted average difference method for detecting differentially expressed genes from microarray data. Algorithms Mol Biol. 2008, 3: 8.

Examples

```
data(nakai)
group <- c(1, 1, 1, 1, 2, 2, 2, 2)

wad <- WAD(nakai, group, logged = TRUE, sort = TRUE)
```

Index

∗ **classes**
TCC-class, 26
∗ **datasets**
arab, 2
hypoData, 13
hypoData_mg, 14
hypoData_ts, 15
nakai, 17
∗ **methods**
calcAUCValue, 3
estimateDE, 8
filterLowCountGenes, 10
getNormalizedData, 11
getResult, 12
makeFCMatrix, 16
plot, 17
plotFCPseudocolor, 19
simulateReadCounts, 22
∗ **packages**
TCC, 25
[*(*TCC-class*)*, 26
[,TCC,ANY,ANY,ANY-method *(*TCC-class*)*, 26
[,TCC,ANY,ANY-method *(*TCC-class*)*, 26
[,TCC,ANY-method *(*TCC-class*)*, 26
[,TCC-method *(*TCC-class*)*, 26

arab, 2

calcAUCValue, 3, *4*
calcNormFactors, 4, *5*, *9*, *11*, *23*, *25, 26*
calcNormFactors,DGEList-method
*(*calcNormFactors*)*, 4
calcNormFactors,TCC-method
*(*calcNormFactors*)*, 4
clusterSample, 7

DESeqDataSet, *8*

eBayes, *9*
estimateCommonDisp, *9*
estimateDE, *4, 5*, 8, *12, 13*, *18*, *23*, *25, 26*
estimateDispersions, *9*
estimateGLMCommonDisp, *9*
estimateGLMTagwiseDisp, *9*
estimateGLMTrendedDisp, *9*
estimateSizeFactors, *5*
estimateTagwiseDisp, *9*
exactTest, *9*

filterLowCountGenes, 10

getLikelihoods, *9*
getNormalizedData, 11
getPriors, *9*
getPriors.NB, *9*
getResult, 12
glmFit, *8, 9*
glmLRT, *8, 9*

hclust, *7*
hypoData, 13
hypoData_mg, 14
hypoData_ts, 15

length *(*TCC-class*)*, 26
length,TCC-method *(*TCC-class*)*, 26
lmFit, *8, 9*

makeFCMatrix, 16, *23*

nakai, 17
names *(*TCC-class*)*, 26
names,TCC-method *(*TCC-class*)*, 26
nbinomLRT, *8, 9*
nbinomWaldTest, *9*

p.adjust, *13*
plot, *12*, 17, *18*, *25*
plot.default, *18*
plotFCPseudocolor, 19, *23*, *25*

results, *8*
ROKU, *15*, 20, *25*

show *(*TCC-class*)*, 26
show,TCC-method *(*TCC-class*)*, 26
show.TCC *(*TCC-class*)*, 26
simulateReadCounts, *4*, *14*, *15*, *18*, *20*, 22, *25*, *26*
subset *(*TCC-class*)*, 26
subset,TCC-method *(*TCC-class*)*, 26

TCC, 25
TCC-class, *3–5*, *8*, *10–12*, *18*, *19*, *23*, *25*, 26
TCC-package *(*TCC*)*, 25
topCounts, *9*

voom, *9*

WAD, *9*, *25*, 27