Package 'BioQC'

April 11, 2018

Type Package

Title Detect tissue heterogeneity in expression profiles with gene sets

Version 1.6.0

Date 2017-07-26

Author Jitao David Zhang <jitao_david.zhang@roche.com>, Laura Badi, Gregor Sturm, Roland Ambs

Maintainer Jitao David Zhang <jitao_david.zhang@roche.com>

Description BioQC performs quality control of high-throughput expression data based on tissue gene signatures. It can detect tissue heterogeneity in gene expression data. The core algorithm is a Wilcoxon-Mann-Whitney test that is optimised for high performance.

Depends utils, Rcpp, Biobase, methods, stats

Collate AllClasses.R AllMethods.R utils.R entropy.R gini.R readGmt.R matchGenes.R wmwTest.R

Suggests testthat, knitr, rmarkdown, lattice, latticeExtra, rbenchmark, gplots, gridExtra, hgu133plus2.db, ineq

VignetteBuilder knitr

biocViews GeneExpression,QualityControl,StatisticalMethod

License GPL (>=3)

RoxygenNote 6.0.1.9000

NeedsCompilation yes

R topics documented:

`absLog10p` *Absolute base-10 logarithm of p-values*

Description

The function returns the absolute values of base-10 logarithm of p-values.

Usage

```
absLog10p(x)
```

Arguments

x	Numeric vector or matrix

Details

The logarithm transformation of p-values is commonly used to visualize results from statistical tests. Although it may cause misunderstanding and therefore its use is disapproved by some experts, it helps to visualize and interpret results of statistical tests intuitively.

The function transforms p-values with base-10 logarithm, and returns its absolute value. The choice of base 10 is driven by the simplicity of interpreting the results.

Value

Numeric vector or matrix.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
testp <- runif(1000, 0, 1)
testp.al <- absLog10p(testp)

print(head(testp))
print(head(testp.al))
```

as.gmtlist *Convert a list of gene symbols into a gmtlist*

Description

Convert a list of gene symbols into a gmtlist

Usage

```
as.gmtlist(list, description = NULL)
```

Arguments

list	A named list with character vectors of genes. Names will become names of gene sets; character vectors will become genes
description	description; will be expanded to the same length of the list

Examples

```
testVec <- list(GeneSet1=c("AKT1", "AKT2"),
                GeneSet2=c("MAPK1", "MAPK3"),
                GeneSet3=NULL)
testVecGmtlist <- as.gmtlist(testVec)
```

BaseIndexList-class *An S4 class to hold a list of indices, with the possibility to specify the offset of the indices. IndexList and SignedIndexList extend this class*

Description

An S4 class to hold a list of indices, with the possibility to specify the offset of the indices. IndexList and SignedIndexList extend this class

Slots

offset An integer specifying the value of first element. Default 1

keepNA Logical, whether NA is kept during construction

keepDup Logical, whether duplicated values are kept during construction

entropy *Shannon entropy and related concepts*

Description

These functions calculate Shannon entropy and related concepts, including diversity, specificity, and specialization. They can be used to quantify gene expression profiles.

Usage

```
entropy(vector)
entropyDiversity(mat, norm=FALSE)
entropySpecificity(mat, norm=FALSE)
sampleSpecialization(mat, norm=TRUE)
```

Arguments

vector	A vector of numbers, or characters. Discrete probability of each item is calculated and the Shannon entropy is returned.
mat	A matrix (usually an expression matrix), with genes (features) in rows and samples in columns.
norm	Logical value. If set to TRUE the scores will be normalized between 0 and 1.

Details

Shannon entropy can be used as measures of gene expression specificity, as well as measures of tissue diversity and specialization. See references below.

We use 2 as base for the entropy calculation, because in this base the unit of entropy is *bit*.

Value

entropy returns one entropy value. entropyDiversity and sampleSpecialization returns a vector as long as the column number of the input matrix. entropySpecificity returns a vector of the length of the row number of the input matrix, namely the specificity score of genes.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

Martinez and Reyes-Valdes (2008) Defining diversity, specialization, and gene specificity in transcriptomes through information theory. PNAS 105(28):9709–9714

Examples

```
myVec0 <- 1:9
entropy(myVec0) ## log2(9)
myVec1 <- rep(1, 9)
entropy(myVec1)

myMat <- rbind(c(3,4,5),c(6,6,6), c(0,2,4))
```

```
entropySpecificity(myMat)
entropySpecificity(myMat, norm=TRUE)
entropyDiversity(myMat)
entropyDiversity(myMat, norm=TRUE)
sampleSpecialization(myMat)
sampleSpecialization(myMat,norm=TRUE)

myRandomMat <- matrix(runif(1000), ncol=20)
entropySpecificity(myRandomMat)
entropySpecificity(myRandomMat, norm=TRUE)
entropyDiversity(myRandomMat)
entropyDiversity(myRandomMat, norm=TRUE)
sampleSpecialization(myRandomMat)
sampleSpecialization(myRandomMat,norm=TRUE)
```

`filterPmat` *Filter rows of p-value matrix under the significance threshold*

Description

Given a p-value matrix and a threshold value, `filterPmat` removes rows where there is no p-values lower than the given threshold.

Usage

```
filterPmat(x, threshold)
```

Arguments

`x`	A matrix of p-values. It must be raw p-values and should not be transformed (e.g. logarithmic).
`threshold`	A numeric value, the minimal p-value used to filter rows. If missing, given the values of `NA`, `NULL` or number `0`, no filtering will be done and the input matrix will be returned.

Value

Matrix of p-values. If no line is left, a empty matrix of the same dimension as input will be returned.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
set.seed(1235)
testMatrix <- matrix(runif(100,0,1), nrow=10)

## filtering
(testMatrix.filter <- filterPmat(testMatrix, threshold=0.05))
## more strict filtering
(testMatrix.strictfilter <- filterPmat(testMatrix, threshold=0.01))
## no filtering
(testMatrix.nofilter <- filterPmat(testMatrix))
```

gini *Calculate Gini Index of a numeric vector*

Description

Calculate the Gini index of a numeric vector.

Usage

```
gini(x)
```

Arguments

x	A numeric vector.

Details

The Gini index (Gini coefficient) is a measure of statistical dispersion. A Gini coefficient of zero expresses perfect equality where all values are the same. A Gini coefficient of one expresses maximal inequality among values.

Value

A numeric value between 0 and 1.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

Gini. C. (1912) *Variability and Mutability*, C. Cuppini, Bologna 156 pages.

Examples

```
testValues <- runif(100)
gini(testValues)
```

GmtList *Convert a list to a GmtList object*

Description

Convert a list to a GmtList object

Usage

```
GmtList(list)
```

Arguments

list	A list of genesets; each geneset is a list of at least three fields: 'name', 'desc', and 'genes'. 'name' and 'desc' contains one character string ('desc' can be NULL while 'name' cannot), and 'genes' can be either NULL or a character vector. For convenience, the function also accepts a list of character vectors, each containing a geneset. In this case, the function works as a wrapper of `as.gmtlist`

See Also

If a list of gene symbols need to be converted into a GmtList, use 'as.gmtlist' instead

Examples

```
testList <- list(list(name="GS_A", desc=NULL, genes=LETTERS[1:3]),
                 list(name="GS_B", desc="gene set B", genes=LETTERS[1:5]),
                 list(name="GS_C", desc="gene set C", genes=NULL))
testGmt <- GmtList(testList)

# as wrapper of as.gmtlist
testGeneList <- list(GS_A=LETTERS[1:3], GS_B=LETTERS[1:5], GS_C=NULL)
testGeneGmt <- GmtList(testGeneList)
```

GmtList-class *An S4 class to hold geneset in the GMT file in a list, each item in the list is in in turn a list containing following items: name, desc, and genes.*

Description

An S4 class to hold geneset in the GMT file in a list, each item in the list is in in turn a list containing following items: name, desc, and genes.

---

gmtlist2signedGenesets *Convert gmtlist into a list of signed genesets*

Description

Convert gmtlist into a list of signed genesets

Usage

```
gmtlist2signedGenesets(gmtlist, posPattern = "_UP$", negPattern = "_DN$",
  nomatch = c("ignore", "pos", "neg"))
```

Arguments

gmtlist	A gmtlist object, probably read-in by readGmt
posPattern	Regular expression pattern of positive gene sets. It is trimmed from the original name to get the stem name of the gene set. See examples below.
negPattern	Regular expression pattern of negative gene sets. It is trimmed from the original name to get the stem name of the gene set. See examples below.
nomatch	Options to deal with gene sets that match neither positive nor negative patterns. ignore: they will be ignored (but not discarded, see details below); pos: they will be counted as positive signs; neg: they will be counted as negative signs

Value

An S4-object of SignedGenesets, which is a list of signed_geneset, each being a two-item list; the first item is 'pos', containing a character vector of positive genes; and the second item is 'neg', containing a character vector of negative genes.

Gene set names are detected whether they are positive or negative. If neither positive nor negative, nomatch will determine how will they be interpreted. In case of pos (or neg), such genesets will be treated as positive (or negative) gene sets.In case nomatch is set to ignore, the gene set will appear in the returned values with both positive and negative sets set to NULL.

Examples

```
testInputList <- list(list(name="GeneSetA_UP",genes=LETTERS[1:3]),
               list(name="GeneSetA_DN", genes=LETTERS[4:6]),
               list(name="GeneSetB", genes=LETTERS[2:4]),
               list(name="GeneSetC_DN", genes=LETTERS[1:3]),
               list(name="GeneSetD_UP", genes=LETTERS[1:3]))
testOutputList.ignore <- gmtlist2signedGenesets(testInputList, nomatch="ignore")
testOutputList.pos <- gmtlist2signedGenesets(testInputList, nomatch="pos")
testOutputList.neg <- gmtlist2signedGenesets(testInputList, nomatch="neg")
```

IndexList *Convert a list to an IndexList object*

Description

Convert a list to an IndexList object

Usage

```
IndexList(object, ..., keepNA = FALSE, keepDup = FALSE, offset = 1L)

## S4 method for signature 'numeric'
IndexList(object, ..., keepNA = FALSE, keepDup = FALSE,
  offset = 1L)

## S4 method for signature 'logical'
IndexList(object, ..., keepNA = FALSE, keepDup = FALSE,
  offset = 1L)
```

```
## S4 method for signature 'list'
IndexList(object, keepNA = FALSE, keepDup = FALSE,
  offset = 1L)
```

Arguments

object	Either a list of unique integer indices, NULL and logical vectors (of same lengths), or a numerical vector or a logical vector. NA is discarded.
...	If object isn't a list, additional vectors can go here.
keepNA	Logical, whether NA indices should be kept or not. Default: FALSE (removed)
keepDup	Logical, whether duplicated indices should be kept or not. Default: FALSE (removed)
offset	Integer, the starting index. Default: 1 (as in the convention of R)

Value

The function returns a list of vectors

Examples

```
testList <- list(GS_A=c(1,2,3,4,3),
                 GS_B=c(2,3,4,5),
                 GS_C=NULL,
                 GS_D=c(1,3,5,NA),
                 GS_E=c(2,4))
testIndexList <- IndexList(testList)
IndexList(c(FALSE, TRUE, TRUE), c(FALSE, FALSE, TRUE), c(TRUE, FALSE, FALSE), offset=0)
IndexList(list(A=1:3, B=4:5, C=7:9))
IndexList(list(A=1:3, B=4:5, C=7:9), offset=0)
```

IndexList-class *An S4 class to hold a list of integers as indices, with the possibility to specify the offset of the indices*

Description

An S4 class to hold a list of integers as indices, with the possibility to specify the offset of the indices

Slots

offset An integer specifying the value of first element. Default 1

keepNA Logical, whether NA is kept during construction

keepDup Logical, whether duplicated values are kept during construction

matchGenes *Match genes in a list-like object to a vector of genesymbols*

Description

Match genes in a list-like object to a vector of genesymbols

Usage

```
matchGenes(list, object, ...)

## S4 method for signature 'GmtList,character'
matchGenes(list, object)

## S4 method for signature 'GmtList,matrix'
matchGenes(list, object)

## S4 method for signature 'GmtList,eSet'
matchGenes(list, object, col = "GeneSymbol")

## S4 method for signature 'character,character'
matchGenes(list, object)

## S4 method for signature 'character,matrix'
matchGenes(list, object)

## S4 method for signature 'character,eSet'
matchGenes(list, object)

## S4 method for signature 'SignedGenesets,character'
matchGenes(list, object)

## S4 method for signature 'SignedGenesets,matrix'
matchGenes(list, object)

## S4 method for signature 'SignedGenesets,eSet'
matchGenes(list, object, col = "GeneSymbol")
```

Arguments

list	A GmtList, list, character or SignedGenesets object
object	Gene symbols to be matched; they can come from a vector of character strings, or a column in the fData of an eSet object.
...	additional arguments like col
col	Column name of fData in an eSet to specify where gene symbols are stored. The default value is set to "GeneSymbol"

offset,BaseIndexList-method *Get offset from an IndexList object*

Description

Get offset from an IndexList object

Usage

```
## S4 method for signature 'BaseIndexList'
offset(object)
```

Arguments

object	An IndexList object

Examples

```
myIndexList <- IndexList(list(1:5, 2:7, 3:8), offset=1L)
offset(myIndexList)
```

offset<- *Set the offset of an* IndexList *or a* SignedIndexList *object*

Description

Set the offset of an IndexList or a SignedIndexList object

Usage

```
`offset<-`(object, value)

## S4 replacement method for signature 'IndexList,numeric'
offset(object) <- value

## S4 replacement method for signature 'SignedIndexList,numeric'
offset(object) <- value
```

Arguments

object	An IndexList or a SignedIndexList object
value	The value, that the offset of object is set too. If it isn't an integer, it's coerced into an integer.

Examples

```
myIndexList <- IndexList(list(1:5, 2:7, 3:8), offset=1L)
offset(myIndexList)
offset(myIndexList) <- 3
offset(myIndexList)
```

readGmt *Read in gene sets from a GMT file*

Description

Read in gene sets from a GMT file

Usage

```
readGmt(filename)
```

Arguments

filename	GMT file name

Value

A gene set list, wrapped in a S3-class `gmtlist`. Each list item is a list with three items: gene set name (`name`), gene set description (`desc`), and gene list (a character vector, `genes`).

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
gmt_file <- system.file("extdata/exp.tissuemark.affy.roche.symbols.gmt", package="BioQC")
gmt_list <- readGmt(gmt_file)
```

readSignedGmt *Read signed GMT files*

Description

Read signed GMT files

Usage

```
readSignedGmt(filename, posPattern = "_UP$", negPattern = "_DN$",
  nomatch = c("ignore", "pos", "neg"))
```

Arguments

filename	A gmt file
posPattern	Pattern of positive gene sets
negPattern	Pattern of negative gene sets
nomatch	options to deal with gene sets that match to neither posPattern nor negPattern patterns

See Also

gmtlist2signedGenesets for parameters posPattern, negPattern, and nomatch

Examples

```
testGmtFile <- system.file("extdata/test.gmt", package="BioQC")
testSignedGenesets.ignore <- readSignedGmt(testGmtFile, nomatch="ignore")
testSignedGenesets.pos <- readSignedGmt(testGmtFile, nomatch="pos")
testSignedGenesets.neg <- readSignedGmt(testGmtFile, nomatch="neg")
```

SignedGenesets *Convert a list to a SignedGenesets object*

Description

Convert a list to a SignedGenesets object

Usage

```
SignedGenesets(list)
```

Arguments

list	A list of genesets; each geneset is a list of at least three fields: 'name', 'pos', and 'neg'. 'name' contains one non-null character string, and both 'pos' and 'neg' can be either NULL or a character vector.

See Also

GmtList

Examples

```
testList <- list(list(name="GS_A", pos=NULL, neg=LETTERS[1:3]),
                 list(name="GS_B", pos=LETTERS[1:5], neg=LETTERS[7:9]),
                 list(name="GS_C", pos=LETTERS[1:5], neg=NULL),
                 list(name="GS_D", pos=NULL, neg=NULL))
testSigndGS <- SignedGenesets(testList)
```

SignedGenesets-class *An S4 class to hold signed genesets, each item in the list is in in turn a list containing following items: name, pos, and neg.*

Description

An S4 class to hold signed genesets, each item in the list is in in turn a list containing following items: name, pos, and neg.

SignedIndexList *Convert a list into a SignedIndexList*

Description

Convert a list into a SignedIndexList

Usage

```
SignedIndexList(object, ...)

## S4 method for signature 'list'
SignedIndexList(object, keepNA = FALSE, keepDup = FALSE,
  offset = 1L)
```

Arguments

object	A list of atleast one list of atleast one list or Vector called either ‘pos‘ or ‘neg‘
...	additional arguments, currently none are used
keepNA	Logical, whether NA indices should be kept or not. Default: FALSE (removed)
keepDup	Logical, whether duplicated indices should be kept or not. Default: FALSE (removed)
offset	offset; 1 if missing

Value

A SignedIndexList of lists (named like the second list-level of the input) containing two vectors named ‘positive‘ and ‘negative‘, which contain the same Argumetns as the IndexList resulting of the ‘pos‘ and ‘neg‘ lists or vectors of the input.

Examples

```
myList <- list(a = list(pos = list(1, 2, 2, 4), neg = c(TRUE, FALSE, TRUE)),
b = list(NA), c = list(pos = c(c(2, 3), c(1, 3))))
SignedIndexList(myList)
```

SignedIndexList-class *An S4 class to hold a list of signed integers as indices, with the possibility to specify the offset of the indices*

Description

An S4 class to hold a list of signed integers as indices, with the possibility to specify the offset of the indices

Slots

offset An integer specifying the value of first element. Default 1

keepNA Logical, whether NA is kept during construction

keepDup Logical, whether duplicated values are kept during construction

simplifyMatrix *Simplify matrix in case of single row/columns*

Description

Simplify matrix in case of single row/columns

Usage

```
simplifyMatrix(matrix)
```

Arguments

matrix	A matrix of any dimension If only one row/column is present, the dimension is dropped and a vector will be returned

Examples

```
testMatrix <- matrix(round(rnorm(9),2), nrow=3)
simplifyMatrix(testMatrix)
simplifyMatrix(testMatrix[1L,,drop=FALSE])
simplifyMatrix(testMatrix[,1L,drop=FALSE])
```

valTypes *prints the options of valTypes of wmwTest*

Description

prints the options of valTypes of wmwTest

Usage

```
valTypes()
```

wmwTest *Wilcoxon-Mann-Whitney rank sum test for high-throughput expression profiling data*

Description

We have implemented a highly efficient Wilcoxon-Mann-Whitney rank sum test for high-throughput expression profiling data. For datasets with more than 100 features (genes), the function can be more than 1,000 times faster than its R implementations (`wilcox.test` in `stats`, or `rankSumTestWithCorrelation` in `limma`).

Usage

```
wmwTest(x, indexList, col = "GeneSymbol", valType = c("p.greater", "p.less",
  "p.two.sided", "U", "abs.log10p.greater", "log10p.less",
  "abs.log10p.two.sided", "Q"), simplify = TRUE)

## S4 method for signature 'matrix,IndexList'
wmwTest(x, indexList, valType, simplify = TRUE)

## S4 method for signature 'numeric,IndexList'
wmwTest(x, indexList, valType, simplify = TRUE)

## S4 method for signature 'matrix,GmtList'
wmwTest(x, indexList, valType, simplify = TRUE)

## S4 method for signature 'eSet,GmtList'
wmwTest(x, indexList, col = "GeneSymbol",
  valType = c("p.greater", "p.less", "p.two.sided", "U", "abs.log10p.greater",
  "log10p.less", "abs.log10p.two.sided", "Q"), simplify = TRUE)

## S4 method for signature 'eSet,numeric'
wmwTest(x, indexList, col = "GeneSymbol",
  valType = c("p.greater", "p.less", "p.two.sided", "U", "abs.log10p.greater",
  "log10p.less", "abs.log10p.two.sided", "Q"), simplify = TRUE)

## S4 method for signature 'eSet,logical'
wmwTest(x, indexList, col = "GeneSymbol",
  valType = c("p.greater", "p.less", "p.two.sided", "U", "abs.log10p.greater",
  "log10p.less", "abs.log10p.two.sided", "Q"), simplify = TRUE)

## S4 method for signature 'eSet,list'
wmwTest(x, indexList, col = "GeneSymbol",
  valType = c("p.greater", "p.less", "p.two.sided", "U", "abs.log10p.greater",
  "log10p.less", "abs.log10p.two.sided", "Q"), simplify = TRUE)

## S4 method for signature 'ANY,numeric'
wmwTest(x, indexList, valType, simplify = TRUE)

## S4 method for signature 'ANY,logical'
wmwTest(x, indexList, valType, simplify = TRUE)

## S4 method for signature 'ANY,list'
wmwTest(x, indexList, valType, simplify = TRUE)

## S4 method for signature 'matrix,SignedIndexList'
wmwTest(x, indexList, valType,
  simplify = TRUE)

## S4 method for signature 'numeric,SignedIndexList'
wmwTest(x, indexList, valType,
  simplify = TRUE)

## S4 method for signature 'eSet,SignedIndexList'
```

```
wmwTest(x, indexList, valType,
  simplify = TRUE)
```

Arguments

x	A numeric matrix. All other data types (e.g. numeric vectors or ExpressionSet objects) are coerced into matrix.
indexList	A list of integer indices (starting from 1) indicating signature genes. Can be of length zero. Other data types (e.g. a list of numeric or logical vectors, or a numeric or logical vector) are coerced into such a list. See details below for a special case using GMT files.
col	a string sometimes used with a eSet
valType	The value type to be returned, allowed values include p.greater, p.less, abs.log10p.greater and abs.log10p.less (one-sided tests),p.two.sided, and U statistic, and their log10 transformation variants. See details below.
simplify	Logical. If not, the returning value is in matrix format; if set to TRUE, the results are simplified into vectors when possible (default).

Details

The basic application of the function is to test the enrichment of gene sets in expression profiling data or differentially expressed data (the matrix with feature/gene in rows and samples in columns).

A special case is when x is an eSet object (e.g. ExpressionSet), and indexList is a list returned from readGmt function. In this case, the only requirement is that one column named GeneSymbol in the featureData contain gene symbols used in the GMT file. See the example below.

Besides the conventional value types such as 'p.greater', 'p.less', 'p.two.sided' , and 'U' (the U-statistic), wmwTest (from version 0.99-1) provides further value types: abs.log10p.greater and log10p.less perform log10 transformation on respective *p*-values and give the transformed value a proper sign (positive for greater than, and negative for less than); abs.log10p.two.sided transforms two-sided *p*-values to non-negative values; and Q score reports absolute log10-transformation of *p*-value of the two-side variant, and gives a proper sign to it, depending on whether it is rather greater than (positive) or less than (negative).

Value

A numeric matrix or vector containing the statistic.

Methods (by class)

- x = matrix,indexList = IndexList: x is a matrix and indexList is a IndexList
- x = numeric,indexList = IndexList: x is a numeric and indexList is a IndexList
- x = matrix,indexList = GmtList: x is a matrix and indexList is a GmtList
- x = eSet,indexList = GmtList: x is a eSet and indexList is a GmtList
- x = eSet,indexList = numeric: x is a eSet and indexList is a numeric
- x = eSet,indexList = logical: x is a eSet and indexList is a logical
- x = eSet,indexList = list: x is a eSet and indexList is a list
- x = ANY,indexList = numeric: x is ANY and indexList is a numeric
- x = ANY,indexList = logical: x is ANY and indexList is a logical
- x = ANY,indexList = list: x is ANY and indexList is a list

- `x = matrix,indexList = SignedIndexList`: x is a `matrix` and `indexList` is a `SignedIndexList`
- `x = numeric,indexList = SignedIndexList`: x is a `numeric` and `indexList` is a `SignedIndexList`
- `x = eSet,indexList = SignedIndexList`: x is a `eSet` and `indexList` is a `SignedIndexList`

Note

The function has been optimized for expression profiling data. It avoids repetitive ranking of data as done by native R implementations and uses efficient C code to increase the performance and control memory use. Simulation studies using expression profiles of 22000 genes in 2000 samples and 200 gene sets suggested that the C implementation can be >1000 times faster than the R implementation. And it is possible to further accelerate by parallel calling the function with `mclapply` in the `multicore` package.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

Barry, W.T., Nobel, A.B., and Wright, F.A. (2008). A statistical framework for testing functional categories in microarray data. _Annals of Applied Statistics_ 2, 286-315.

Wu, D, and Smyth, GK (2012). Camera: a competitive gene set test accounting for inter-gene correlation. _Nucleic Acids Research_ 40(17):e133

Zar, JH (1999). _Biostatistical Analysis 4th Edition_. Prentice-Hall International, Upper Saddle River, New Jersey.

See Also

codewilcox.test in the `stats` package, and `rankSumTestWithCorrelation` in the `limma` package. #' @examples ## R-native data structures set.seed(1887) rd <- rnorm(1000) rl <- sample(c(TRUE, FALSE), 1000, replace=TRUE) wmwTest(rd, rl, valType="p.two.sided") wmwTest(rd, which(rl), valType="p.two.sided") rd1 <- rd + ifelse(rl, 0.5, 0) wmwTest(rd1, rl, valType="p.greater") wmwTest(rd1, rl, valType="U") rd2 <- rd - ifelse(rl, 0.2, 0) wmwTest(rd2, rl, valType="p.greater") wmwTest(rd2, rl, valType="p.two.sided") wmwTest(rd2, rl, valType="p.less")

matrix forms rmat <- matrix(c(rd, rd1, rd2), ncol=3, byrow=FALSE) wmwTest(rmat, rl, valType="p.two.sided") wmwTest(rmat, rl, valType="p.greater")

wmwTest(rmat, which(rl), valType="p.two.sided") wmwTest(rmat, which(rl), valType="p.greater")

other valTypes wmwTest(rmat, which(rl), valType="U") wmwTest(rmat, which(rl), valType="abs.log10p.greater") wmwTest(rmat, which(rl), valType="log10p.less") wmwTest(rmat, which(rl), valType="abs.log10p.two.sided") wmwTest(rmat, which(rl), valType="Q")

using ExpressionSet data(sample.ExpressionSet) testSet <- sample.ExpressionSet fData(testSet)$GeneSymbol <- paste("GENE_",1:nrow(testSet), sep="") mySig1 <- sample(c(TRUE, FALSE), nrow(testSet), prob=c(0.25, 0.75), replace=TRUE) wmwTest(testSet, which(mySig1), valType="p.greater")

using integer exprs(testSet)[,1L] <- exprs(testSet)[,1L] + ifelse(mySig1, 50, 0) wmwTest(testSet, which(mySig1), valType="p.greater")

using lists mySig2 <- sample(c(TRUE, FALSE), nrow(testSet), prob=c(0.6, 0.4), replace=TRUE) wmwTest(testSet, list(first=mySig1, second=mySig2)) ## using GMT file gmt_file <- system.file("extdata/exp.tissuemark package="BioQC") gmt_list <- readGmt(gmt_file)

gss <- sample(unlist(sapply(gmt_list, function(x) x$genes)), 1000) eset<-new("ExpressionSet", exprs=matrix(rnorm(10000), nrow=1000L), phenoData=new("AnnotatedDataFrame", data.frame(Sample=LETTERS[1:10

featureData=new("AnnotatedDataFrame",data.frame(GeneSymbol=gss))) esetWmwRes <- wmwTest(eset ,gmt_list, valType="p.greater") summary(esetWmwRes)

`wmwTestInR` *Wilcoxon-Mann-Whitney test in R*

Description

Wilcoxon-Mann-Whitney test in R

Usage

```
wmwTestInR(x, sub, valType = c("p.greater", "p.less", "p.two.sided", "W"))
```

Arguments

`x`	A numerical vector
`sub`	A logical vector or integer vector to subset `x`. Numbers in `sub` are compared with numbers out of `sub`
`valType`	Type of retured-value. Supported values: p.greater, p.less, p.two.sided, and W statistic (note it is different from the U statistic)

Examples

```
testNums <- 1:10
testSub <- rep_len(c(TRUE, FALSE), length.out=length(testNums))
wmwTestInR(testNums, testSub)
wmwTestInR(testNums, testSub, valType="p.two.sided")
wmwTestInR(testNums, testSub, valType="p.less")
wmwTestInR(testNums, testSub, valType="W")
```

Index