Package ‘BioQC’

March 20, 2024

Type Package

Title Detect tissue heterogeneity in expression profiles with gene sets

Version 1.30.0

Date 2021-08-18

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.

biocViews GeneExpression,QualityControl,StatisticalMethod, GeneSetEnrichment

Depends R (>= 3.5.0), Biobase

Imports edgeR, Rcpp, methods, stats, utils


Suggests testthat, knitr, markdown, lattice, latticeExtra, rbenchmark, gplots, gridExtra, org.Hs.eg.db, hgu133plus2.db, ggplot2, reshape2, plyr, ineq, covr, limma, RColorBrewer

LinkingTo Rcpp

VignetteBuilder knitr

License GPL (>=3) + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

URL https://accio.github.io/BioQC

BugReports https://accio.github.io/BioQC/issues

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

git_branch RELEASE_3_18

git_last_commit e232c77
R topics documented:

- absLog10p
- appendGmtList
- as.GmtList
- BaseIndexList-class
- entropy
- entropyDiversity
- entropySpecificity
- filterBySize
- filterPmat
- getLeadingEdgeIndexFromVector
- gini
- GmtList
- GmtList-class
- gmtlist2signedGenesets
- gsDesc
- gsGeneCount
- gsGenes
- gsName
- gsNamespace
- gsNamespace<-
- hasNamespace
- IndexList
- IndexList-class
- isValidBaseIndexList
- isValidGmtList
- isValidIndexList
- isValidSignedGenesets
- isValidSignedIndexList
- matchGenes
- offset
- offset<-
- prettySigNames
- readCurrentSignatures
- readGmt
Description

Absolute base-10 logarithm of p-values

Usage

absLog10p(x)

Arguments

x  Numeric vector or matrix

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

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.
appendGmtList

Value
Numeric vector or matrix.

Author(s)
Jitao David Zhang <jitao_david.zhang@roche.com>

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

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

Description
Append a GmtList object to another one

Usage
```r
appendGmtList(gmtList, newGmtList, ...)
```

Arguments
- `gmtList`: A GmtList object
- `newGmtList`: Another GmtList object to be appended
- `...`: Further GmtList object to be appended

Value
A new GmtList list, with all elements in the input appended in the given order

Examples
```r
test_gmt_file<- system.file("extdata/test.gmt", package="BioQC")
testGmtList1 <- readGmt(test_gmt_file, namespace="test1")
testGmtList2 <- readGmt(test_gmt_file, namespace="test2")
testGmtList3 <- readGmt(test_gmt_file, namespace="test3")
testGmtAppended <- appendGmtList(testGmtList1, testGmtList2, testGmtList3)
```
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, uniqGenes = TRUE, namespace = NULL)

Arguments

list
A named list with character vectors of genes. Names will become names of gene sets; character vectors will become genes

description
Character, description of gene-sets. The value will be expanded to the same length of the list.

uniqGenes
Logical, whether redundant genes should be made unique?

namespace
Character or NULL, namespace of the gene-set

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
Shannon entropy

Usage

entropy(vector)

Arguments

vector A vector of numbers, or characters. Discrete probability of each item is calculated and the Shannon entropy is returned.

Value

Shannon entropy

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.

Author(s)

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

References


Examples

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

entropy(LETTERS)
test_entropy(LETTERS)
test_entropy(rep(LETTERS, 5))
```
entropyDiversity

Entropy-based sample diversity

Description

Entropy-based sample diversity

Usage

entropyDiversity(mat, norm = FALSE)

Arguments

mat  A matrix (usually an expression matrix), with genes (features) in rows and samples in columns.

norm Logical, whether the diversity should be normalized by \(\log_2(nrow(mat))\).

Value

A vector as long as the column number of the input matrix

References


See Also

entropy and sampleSpecialization

Examples

myMat <- rbind(c(3,4,5),c(6,6,6), c(0,2,4))
entropyDiversity(myMat)
entropyDiversity(myMat, norm=TRUE)

myRandomMat <- matrix(runif(1000), ncol=20)
entropyDiversity(myRandomMat)
entropyDiversity(myRandomMat, norm=TRUE)
Entropy-based gene-expression specificity

**Description**

Entropy-based gene-expression specificity

**Usage**

```r
entropySpecificity(mat, norm = FALSE)
```

**Arguments**

- `mat`: A matrix (usually an expression matrix), with genes (features) in rows and samples in columns.
- `norm`: Logical, whether the specificity should be normalized by log2(ncol(mat)).

**Value**

A vector of the length of the row number of the input matrix, namely the specificity score of genes.

**References**


**See Also**

`entropy`

**Examples**

```r
myMat <- rbind(c(3,4,5),c(6,6,6), c(0,2,4))
entropySpecificity(myMat)
entropySpecificity(myMat, norm=TRUE)

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

*Filter a GmtList by size*

**Description**

Filter a GmtList by size

**Usage**

`filterBySize(x, min, max)`

**Arguments**

- `x`: A GmtList object
- `min`: Numeric, gene-sets with fewer genes than `min` will be removed
- `max`: Numeric, gene-sets with more genes than `max` will be removed

**Value**

A GmtList object with sizes (count of genes) between `min` and `max` (inclusive).

---

**filterPmat**

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

**Description**

Filter rows of p-value matrix under the significance 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

```r
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))
```

getLeadingEdgeIndexFromVector

*Getting leading-edge indices from a vector*

Description

Getting leading-edge indices from a vector

Usage

```r
getLeadingEdgeIndexFromVector(
  x,
  index,
  comparison = c("greater", "less"),
  reference = c("background", "geneset")
)
```

```r
getLeadingEdgeIndexFromMatrix(
  x,
  index,
  comparison = c("greater", "less"),
  reference = c("background", "geneset")
)
```

Arguments

- `x` A numeric vector (getLeadingEdgeIndexFromVector) or a numeric matrix (getLeadingEdgeIndexFromMatrix).
- `index` An integer vector, indicating the indices of genes in a gene-set.
- `comparison` Character string, are values greater than or less than the reference value considered as leading-edge? This depends on the type of value requested by the user in wmwTest.
Character string, which reference is used? If background, genes with expression higher than the median of the background are reported. Otherwise in the case of geneset, genes with expression higher than the median of the gene-set is reported. Default is background, which is consistent with the results of the Wilcoxon-Mann-Whitney tests.

Value

An integer vector, indicating the indices of leading-edge genes.

Functions

- `getLeadingEdgeIndexFromMatrix`: x is a matrix.

See Also

`wmwTest`

Examples

```r
myProfile <- c(rnorm(5, 3), rnorm(15, -3), rnorm(100, 0))
gini(myProfile)

getLeadingEdgeIndexFromVector(myProfile, 1:20)
getLeadingEdgeIndexFromVector(myProfile, 1:20, comparison="less")
getLeadingEdgeIndexFromVector(myProfile, 1:20, comparison="less", reference="geneset")

myProfile2 <- c(rnorm(15, 3), rnorm(5, -3), rnorm(100, 0))
myProfileMat <- cbind(myProfile, myProfile2)
getLeadingEdgeIndexFromMatrix(myProfileMat, 1:20)
getLeadingEdgeIndexFromMatrix(myProfileMat, 1:20, comparison="less")
getLeadingEdgeIndexFromMatrix(myProfileMat, 1:20, comparison="less", reference="geneset")
```

---

**gini**

*Calculate Gini Index of a numeric vector*

**Description**

Calculate the Gini index of a numeric vector.

**Usage**

```r
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


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. In addition, ’namespace’ is accepted to represent the namespace. 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

```r
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")
```

---

### gsDesc

#### Gene-set descriptions

**Description**

Gene-set descriptions

**Usage**

```r
gsDesc(x)
```

**Arguments**

- `x`  
  A GmtList object

**Value**

Descriptions as a vector of character strings of the same length as `x`
**gsGeneCount**

*Gene-set gene counts*

**Description**

Gene-set gene counts

*gsSize* is the synonym of *gsGeneCount*

**Usage**

```r
gsGeneCount(x, uniqGenes = TRUE)
```

```r
gsSize(x, uniqGenes = TRUE)
```

**Arguments**

- `x`: A `GmtList` or similar object
- `uniqGenes`: Logical, whether only unique genes are counted

**Value**

Gene counts (aka gene-set sizes) as a vector of integer of the same length as `x`

---

**gsGenes**

*Gene-set member genes*

**Description**

Gene-set member genes

**Usage**

```r
gsGenes(x)
```

**Arguments**

- `x`: A `GmtList` object

**Value**

A list of genes as character strings of the same length as `x`
### gsName

**Gene-set names**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene-set names</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>gsName(x)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>x A GmtList object</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Names as a vector of character strings of the same length as x</td>
</tr>
</tbody>
</table>

### gsNamespace

**Gene-set namespaces**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene-set namespaces</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>gsNamespace(x)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>x A GmtList object</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespaces as a vector of character strings of the same length as x</td>
</tr>
</tbody>
</table>
gsNamespace<-  

Description

gsNamespace<- is the synonym of setGsNamespace

Usage

gsNamespace(x) <- value

Arguments

x A GmtList object
value namespace in setGsNamespace. It can be either a function that applies to a gene-set list element of the object (for instance function(x) x$desc to extract description), or a vector of the same length of x, or in the special case NULL, which will erase the field namespace.

hasNamespace

Description

Whether namespace is set

Usage

hasNamespace(x)

Arguments

x A GmtList object

Value

Logical, whether all gene-sets have the field namespace set
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

isValidBaseIndexList Function to validate a BaseIndexList object

Description

Function to validate a BaseIndexList object

Usage

isValidBaseIndexList(object)

Arguments

object A BaseIndexList object Use setValidity("BaseIndexList", "isValidBaseIndexList") to check integrity of BaseIndexList objects. It can be very slow, therefore the feature is not turned on by default
isValidGmtList  
*Function to validate a GmtList object*

**Description**

Function to validate a GmtList object

**Usage**

`isValidGmtList(object)`

**Arguments**

- **object**  
  A GmtList object. Use `setValidity("GmtList", "isValidGmtList")` to check integrity of GmtList objects. It can be very slow, therefore the feature is not turned on by default.

isValidIndexList  
*Function to validate an IndexList object*

**Description**

Function to validate an IndexList object

**Usage**

`isValidIndexList(object)`

**Arguments**

- **object**  
  An IndexList object. Use `setValidity("BaseIndexList", "isValidBaseIndexList")` to check integrity of IndexList objects. It can be very slow, therefore the feature is not turned on by default.
isValidSignedGenesets  Function to validate a SignedGenesets object

Description

Function to validate a SignedGenesets object

Usage

isValidSignedGenesets(object)

Arguments

object  A SignedGenesets object Use setValidity("SignedGenesets", "isValidSignedGenesets") to check integrity of SignedGenesets objects. It can be very slow, therefore the feature is not turned on by default

isValidSignedIndexList  Function to validate a SignedIndexList object

Description

Function to validate a SignedIndexList object

Usage

isValidSignedIndexList(object)

Arguments

object  a SignedIndexList object Use setValidity("SignedIndexList", "isValidSignedIndexList") to check integrity of SignedIndexList objects. It can be very slow, therefore the feature is not turned on by default
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 'character,DGEList'
matchGenes(list, object, col = "GeneSymbol")

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

## 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")

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

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.

- **col**
  Column name of fData in an eSet object, or genes in an DGEList object, to specify where gene symbols are stored. The default value is set to "GeneSymbol"

Value

An IndexList object, which is essentially a list of the same length as input (length of 1 in case characters are used as input), with matching indices.

Examples

```r
## test GmtList, character
testGenes <- sprintf("gene%d", 1:10)
testGeneSets <- GmtList(list(gs1=c("gene1", "gene2"), gs2=c("gene9", "gene10"), gs3=c("gene100")))
matchGenes(testGeneSets, testGenes)

## test GmtList, matrix
testGenes <- sprintf("gene%d", 1:10)
testGeneSets <- GmtList(list(gs1=c("gene1", "gene2"), gs2=c("gene9", "gene10"), gs3=c("gene100")))
testGeneExprs <- matrix(rnorm(100), nrow=10, dimnames=list(testGenes, sprintf("sample%d", 1:10)))
matchGenes(testGeneSets, testGeneExprs)

## test GmtList, eSet
testGenes <- sprintf("gene%d", 1:10)
testGeneSets <- GmtList(list(gs1=c("gene1", "gene2"), gs2=c("gene9", "gene10"), gs3=c("gene100")))
testGeneExprs <- matrix(rnorm(100), nrow=10, dimnames=list(testGenes, sprintf("sample%d", 1:10)))
testFeat <- data.frame(GeneSymbol=rownames(testGeneExprs), row.names=testGenes)
testPheno <- data.frame(SampleId=colnames(testGeneExprs), row.names=colnames(testGeneExprs))
testEset <- ExpressionSet(assayData=testGeneExprs, featureData=AnnotatedDataFrame(testFeat), phenoData=AnnotatedDataFrame(testPheno))
matchGenes(testGeneSets, testGeneExprs)

## force using row names
matchGenes(testGeneSets, testEset, col=NULL)

## test GmtList, DGEList
if(requireNamespace("edgeR")) {
  mat <- matrix(rbinom(100, mu=5, size=2), ncol=10)
  rownames(mat) <- sprintf("gene%d", 1:nrow(mat))
  y <- edgeR::DGEList(counts=mat, group=rep(1:2, each=5))

  ## if genes are not set, row names of the count matrix will be used for lookup
  myGeneSet <- GmtList(list(gs1=rownames(mat)[1:2], gs2=rownames(mat)[9:10], gs3="gene100")
  matchGenes(myGeneSet, y)
}
```
matchGenes(c("gene1", "gene2"), y)
## alternatively, use 'col' parameter to specify the column in 'genes'
y2 <- edgeR::DGEList(counts=mat,
    group=rep(1:2, each=5),
    genes=data.frame(GeneIdentifier=rownames(mat), row.names=rownames(mat)))
matchGenes(myGeneSet, y2, col="GeneIdentifier")
}

## test character, character
matchGenes(c("gene1", "gene2"), testGenes)

## test character, matrix
matchGenes(c("gene1", "gene2"), testGeneExprs)

## test character, eset
matchGenes(c("gene1", "gene2"), testEset)

offset

### offset

Get offset from an IndexList object

**Description**

Get offset from an IndexList object

**Usage**

offset(object)

## 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)`

```r
## 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

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

prettySigNames

---

Prettify default signature names

Description

Prettify default signature names

Usage

`prettySigNames(names, includeNamespace = TRUE)`

Arguments

- **names**: Character strings, signature names
- **includeNamespace**: Logical, whether the namespace of the signatures should be included
readCurrentSignatures

Value

Character strings, pretty signature names

Examples

```r
sig <- readCurrentSignatures()
prettyNames <- prettySigNames(names(sig))
```

Description

Load current BioQC signatures

Usage

```r
readCurrentSignatures(uniqGenes = TRUE, namespace = NULL)
```

Arguments

- `uniqGenes`: Logical, whether duplicated genes should be removed, passed to `readGmt`
- `namespace`: Character, namespace of the gene-set, or code NULL, passed to `readGmt`

Value

A GmtList

See Also

`readGmt`

Examples

```r
readCurrentSignatures()
```
readGmt

Read in gene-sets from a GMT file

Description
Read in gene-sets from a GMT file

Usage
readGmt(..., uniqGenes = TRUE, namespace = NULL)

Arguments

... Named or unnamed character string vector, giving file names of one or more GMT format files.
uniqGenes Logical, whether duplicated genes should be removed
namespace Character, namespace of the gene-set. It can be used to specify namespace or sources of the gene-sets. If NULL is given, no namespace is used and all gene-sets are assumed to come from the same unspecified namespace. The option can be helpful when gene-sets from multiple namespaces are jointly used.

Value
A GmtList object, which is a S4-class wrapper of a list. Each element in the object is a list of (at least) three items:

• gene-set name (field name), character string, accessible with gsName
• gene-set description (field desc), character string, accessible with gsDesc
• genes (field genes), a vector of character strings, accessible with gsGenes
• namespace (field namespace), accessible with gsNamespace

Note
Currently, when namespace is set as NULL, no namespace is used. This may change in the future, since we may use file base name as the default namespace.

Examples

gmt_file <- system.file("extdata/exp.tissuemark.affy.roche.symbols.gmt", package="BioQC")
gmt_list <- readGmt(gmt_file)
gmt_nonUniqGenes_list <- readGmt(gmt_file, uniqGenes=FALSE)
gmt_namespace_list <- readGmt(gmt_file, uniqGenes=FALSE, namespace="myNamespace")

## suppose we have two lists of gene-sets to read in
test_gmt_file <- system.file("extdata/test.gmt", package="BioQC")
gmt_twons_list <- readGmt(gmt_file, test_gmt_file, namespace=c("BioQC", "test"))
## alternatively
gmt_twons_list <- readGmt(BioQC=gmt_file, test=test_gmt_file)
readSignedGmt

Read signed GMT files

Description

Read signed GMT files

Usage

readSignedGmt(
  filename,
  posPattern = "_UP$",
  negPattern = "_DN$",
  nomatch = c("ignore", "pos", "neg"),
  uniqGenes = TRUE,
  namespace = NULL
)

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
uniqGenes Logical, whether genes should be made unique
namespace Character string or NULL, namespace of gene-sets

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")
sampleSpecialization  

Entropy-based sample specialization

Description

Entropy-based sample specialization

Usage

```r
sampleSpecialization(mat, norm = TRUE)
```

Arguments

- `mat`  
  A matrix (usually an expression matrix), with genes (features) in rows and samples in columns.
- `norm`  
  Logical, whether the specialization should be normalized by \( \log_2(\text{ncol}(\text{mat})) \).

Value

A vector as long as the column number of the input matrix

References


See Also

`entropy` and `entropyDiversity`

Examples

```r
myMat <- rbind(c(3,4,5),c(6,6,6), c(0,2,4))
sampleSpecialization(myMat)
sampleSpecialization(myMat, norm=TRUE)

myRandomMat <- matrix(runif(1000), ncol=20)
sampleSpecialization(myRandomMat)
sampleSpecialization(myRandomMat, norm=TRUE)
```
setDescAsNamespace: Set gene-set description as namespace

Description
Set gene-set description as namespace

Usage
setDescAsNamespace(x)

Arguments
x
A GmtList object
This function wraps setNamespace to set gene-set description as namespace

See Also
setNamespace

setNamespace: Set the namespace field in each gene-set within a GmtList

Description
Set the namespace field in each gene-set within a GmtList

Usage
setNamespace(x, namespace)

Arguments
x
A GmtList object encoding a list of gene-sets
namespace
It can be either a function that applies to a gene-set list element of the object (for instance function(x) x$desc to extract description), or a vector of the same length of x, or in the special case NULL, which will erase the field namespace.

Note that using vectors as namespace leads to poor performance when the input object has many gene-sets.
**Examples**

```r
gmtlist <- GmtList(list(list(name="GeneSet1", desc="Namespace1", genes=LETTERS[1:3]),
                   list(name="GeneSet2", desc="Namespace1", genes=rep(LETTERS[4:6],2)),
                   list(name="GeneSet1", desc="Namespace1", genes=LETTERS[4:6]),
                   list(name="GeneSet3", desc="Namespace2", genes=LETTERS[1:5])))

hasNamespace(gmtlist)
gmtlist2 <- setNamespace(gmtlist, namespace=function(x) x$desc)

gsNamespace(gmtlist2)
## the function can provide flexible ways to encode the gene-set namespace

myGmtList3 <- setNamespace(myGmtList, namespace=function(x) gsub("Namespace", "C", x$desc))

gsNamespace(myGmtList3)
## using vectors

myGmtList4 <- setNamespace(myGmtList, namespace=c("C1", "C1", "C1", "C2"))

gsNamespace(myGmtList4)

myGmtList2null <- setNamespace(myGmtList2, namespace=NULL)

hasNamespace(myGmtList2null)
```

---

**Description**

Show method for GmtList

**Usage**

```r
## S4 method for signature 'GmtList'
show(object)
```

**Arguments**

- `object`: An object of the class GmtList

---

**Description**

Show method for IndexList

**Usage**

```r
## S4 method for signature 'IndexList'
show(object)
```

**Arguments**

- `object`: An object of the class IndexList
Show method for SignedGenesets

Description
Show method for SignedGenesets

Usage
## S4 method for signature 'SignedGenesets'
show(object)

Arguments
- object: An object of the class SignedGenesets

Show method for SignedIndexList

Description
Show method for SignedIndexList

Usage
## S4 method for signature 'SignedIndexList'
show(object)

Arguments
- object: An object of the class SignedIndexList
**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**

```r
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 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 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 lists, each with two elements named ‘pos’ or ‘neg’, can be logical vectors or integer indices
- **...**: additional arguments, currently ignored
- **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, a list of lists, containing two vectors named ‘positive’ and ‘negative’, which contain the indices of genes that are either positively or negatively associated with a certain pheno-type

Examples

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

## a special case of input is a single list with two elements, \code{pos} and \code{neg}
SignedIndexList(myList[[1]])
```
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])
uniqGenesetsByNamespace

Make names of gene-sets unique by namespace, and member genes of gene-sets unique

Description

Make names of gene-sets unique by namespace, and member genes of gene-sets unique

Usage

uniqGenesetsByNamespace(gmtList)

Arguments

gmtList A GmtList object, probably from readGmt. The object must have namespaces defined by setNamespace.

The function make sure that

- names of gene-sets within each namespace are unique, by merging gene-sets with duplicated names
- genes within each gene-set are unique, by removing duplicated genes

Gene-sets with duplicated names and different desc are merged, desc are made unique, and in case of multiple values, concatenated (with | as the collapse character).

Value

A GmtList object, with unique gene-sets and unique gene lists. If not already present, a new item namespace is appended to each list element in the GmtList object, recording the namespace used to make gene-sets unique. The order of the returned GmtList object is given by the unique gene-set name of the input object.

Examples

myGmtList <- GmtList(list(list(name="GeneSet1", desc="Namespace1", genes=LETTERS[1:3]),
  list(name="GeneSet2", desc="Namespace1", genes=rep(LETTERS[4:6],2)),
  list(name="GeneSet1", desc="Namespace1", genes=LETTERS[4:6]),
  list(name="GeneSet3", desc="Namespace2", genes=LETTERS[1:5])))

print(myGmtList)
myGmtList <- setNamespace(myGmtList, namespace=function(x) x$desc)
myUniqGmtList <- uniqGenesetsByNamespace(myGmtList)
print(myUniqGmtList)
valTypes

prints the options of valTypes of wmwTest

Description

prints the options of valTypes of wmwTest

Usage

valTypes()

wmwLeadingEdge

Identify BioQC leading-edge genes of one gene-set

Description

Identify BioQC leading-edge genes of one gene-set

Usage

wmwLeadingEdge(
  matrix,
  indexVector,
  valType = c("p.greater", "p.less", "p.two.sided", "U", "abs.log10p.greater",
             "log10p.less", "abs.log10p.two.sided", "Q", "r", "f", "U1", "U2"),
  thr = 0.05,
  reference = c("background", "geneset")
)

Arguments

matrix: A numeric matrix
indexVector: An integer vector, giving indices of a gene-set of interest
valType: Value type, consistent with the types in wmwTest
thr: Threshold of the value, greater or less than which the gene-set is considered significantly enriched in one sample
reference: Character string, which reference is used? If background, genes with expression higher than the median of the background are reported. Otherwise in the case of geneset, genes with expression higher than the median of the gene-set is reported. Default is background, which is consistent with the results of the Wilcoxon-Mann-Whitney tests.
Value

A list of integer vectors.

BioQC leading-edge genes are defined as those features whose expression is higher than the median expression of the background in a sample. The function identifies leading-edge genes of a given dataset (specified by the index vector) in a number of samples (specified by the matrix, with genes/features in rows and samples in columns) in three steps. The function calls `wmwTest` to run BioQC and identify samples in which the gene-set is significantly enriched. The enrichment criteria is specified by `valType` and `thr`. Then the function identifies genes in the gene-set that have greater or less expression than the median value of the reference in those samples showing significant enrichment. Finally, it reports either leading-edge genes in individual samples, or the intersection/union of leading-edge genes in multiple samples.

See Also

`wmwTest`

Examples

```r
myProfile <- c(rnorm(5, 3), rnorm(15, -3), rnorm(100, 0))
myProfile2 <- c(rnorm(15, 3), rnorm(5, -3), rnorm(100, 0))
myProfile3 <- c(rnorm(10, 5), rnorm(10, 0), rnorm(100, 0))
myProfileMat <- cbind(myProfile, myProfile2, myProfile3)
wmwLeadingEdge(myProfileMat, 1:20, valType="p.greater")
wmwLeadingEdge(myProfileMat, 1:20, valType="log10p.less")
wmwLeadingEdge(myProfileMat, 1:20, valType="U", reference="geneset")
wmwLeadingEdge(myProfileMat, 1:20, valType="abs.log10p.greater")
```

wmwTest

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

Description

`wmwTest` is a highly efficient Wilcoxon-Mann-Whitney rank sum test for high-dimensional data, such as gene expression profiling. 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

```r
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", "r", "f", "U1", "U2"),
  simplify = TRUE
)```
wmwTest

```r
## S4 method for signature 'matrix,IndexList'
wmwTest(x, indexList, valType = "p.greater", simplify = TRUE)

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

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

## S4 method for signature 'eSet,GmtList'
wmwTest(
  x,
  indexList,
  col = "GeneSymbol",
  valType = "p.greater",
  simplify = TRUE
)

## S4 method for signature 'eSet,numeric'
wmwTest(
  x,
  indexList,
  col = "GeneSymbol",
  valType = "p.greater",
  simplify = TRUE
)

## S4 method for signature 'eSet,logical'
wmwTest(
  x,
  indexList,
  col = "GeneSymbol",
  valType = "p.greater",
  simplify = TRUE
)

## S4 method for signature 'eSet,list'
wmwTest(
  x,
  indexList,
  col = "GeneSymbol",
  valType = "p.greater",
  simplify = TRUE
)

## S4 method for signature 'ANY,numeric'

)
```
wmwTest(x, indexList, valType = "p.greater", simplify = TRUE)

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

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

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

## S4 method for signature 'matrix,SignedGenesets'
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)

## S4 method for signature 'eSet,SignedGenesets'
wmwTest(
  x,
  indexList,
  col = "GeneSymbol",
  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 (or more specifically, either U1 or U2), and a few other 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. The same applies to signed Gmt files. 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).

From version 1.19.1, the rank-biserial correlation coefficient (‘r’) and the common language effect size (‘\( f \)’) are supported value types.

Before version 1.19.3, the ‘U’ statistic returned is in fact ‘U2’. From version 1.19.3, ‘U1’ is returned when ‘U’ is used, and users can specify additional parameter values ‘U1’ and ‘U2’. The sum of ‘U1’ and ‘U2’ is the product of the sizes of two vectors to be compared.

Value

A numeric matrix or vector containing the statistic.

Methods (by class)

- \( x = \text{matrix}, \text{indexList} = \text{IndexList} \): \( x \) is a matrix and indexList is a IndexList
- \( x = \text{numeric}, \text{indexList} = \text{IndexList} \): \( x \) is a numeric and indexList is a IndexList
- \( x = \text{matrix}, \text{indexList} = \text{GmtList} \): \( x \) is a matrix and indexList is a GmtList
- \( x = \text{eSet}, \text{indexList} = \text{GmtList} \): \( x \) is a eSet and indexList is a GmtList
- \( x = \text{eSet}, \text{indexList} = \text{numeric} \): \( x \) is a eSet and indexList is a numeric
- \( x = \text{eSet}, \text{indexList} = \text{logical} \): \( x \) is a eSet and indexList is a logical
- \( x = \text{eSet}, \text{indexList} = \text{list} \): \( x \) is a eSet and indexList is a list
- \( x = \text{ANY}, \text{indexList} = \text{numeric} \): \( x \) is ANY and indexList is a numeric
- \( x = \text{ANY}, \text{indexList} = \text{logical} \): \( x \) is ANY and indexList is a logical
- \( x = \text{ANY}, \text{indexList} = \text{list} \): \( x \) is ANY and indexList is a list
- \( x = \text{matrix}, \text{indexList} = \text{SignedIndexList} \): \( x \) is a matrix and indexList is a SignedIndexList
- \( x = \text{matrix}, \text{indexList} = \text{SignedGenesets} \): \( x \) is a eSet and indexList is a SignedIndexList
- \( x = \text{numeric}, \text{indexList} = \text{SignedIndexList} \): \( x \) is a numeric and indexList is a SignedIndexList
- \( x = \text{eSet}, \text{indexList} = \text{SignedIndexList} \): \( x \) is a eSet and indexList is a SignedIndexList
- \( x = \text{eSet}, \text{indexList} = \text{SignedGenesets} \): \( x \) 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>, with critical inputs from Jan Aettig and Iakov Davydov about U statistics.

**References**


**See Also**

codewilcox.test in the stats package, and rankSumTestWithCorrelation in the limma package.

**Examples**

```r
## 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.less")
wmwTest(rd2, rl, valType="r")
wmwTest(rd2, rl, valType="f")

## 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")
wmwTest(rmat, which(rl), valType="r")
wmwTest(rmat, which(rl), valType="f")

## 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.affy.roche.symbols.gmt", 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)

## using signed GMT file
signed_gmt_file <- system.file("extdata/test.gmt", package="BioQC")
signed_gmt <- readSignedGmt(signed_gmt_file)
esetSignedWmwRes <- wmwTest(eset, signed_gmt, valType="p.greater")
esetMat <- exprs(eset); rownames(esetMat) <- fData(eset)$GeneSymbol
esetSignedWmwRes2 <- wmwTest(esetMat, signed_gmt, valType="p.greater")

---

**wmwTestInR**

Wilcoxon-Mann-Whitney test in R

**Description**

Wilcoxon-Mann-Whitney test in R
Usage

\texttt{wmwTestInR(x, sub, valType = c("p.greater", "p.less", "p\_two\_sided", "W\")})

Arguments

- \texttt{x} A numerical vector
- \texttt{sub} A logical vector or integer vector to subset \texttt{x}. Numbers in \texttt{sub} are compared with numbers out of \texttt{sub}
- \texttt{valType} Type of returned-value. Supported values: p.greater, p.less, p\_two\_sided, and W statistic (note it is different from the U statistic)

Examples

```r
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")
```

[.\texttt{GmtList} 

\textit{Subsetting GmtList object into another GmtList object}

Description

Subsetting GmtList object into another GmtList object

Usage

```r
## S3 method for class 'GmtList'
x[i, drop = FALSE]
```

Arguments

- \texttt{x} A GmtList object
- \texttt{i} Index to subset
- \texttt{drop} In case only one element remains, should a list representing the single geneset returned? Default: FALSE

Examples

```r
myGmtList <- GmtList(list(gs1=letters[1:3], gs2=letters[3:4], gs3=letters[4:5]))
myGmtList[1:2]
myGmtList[1] # default behaviour: not dropping
myGmtList[1,drop=TRUE] # force dropping
```
Description

Subsetting GmtList object to fetch one gene-set

Usage

```r
## S3 method for class 'GmtList'
x[[i]]
```

Arguments

- `x` A GmtList object
- `i` The index to subset

Examples

```r
myGmtList <- GmtList(list(gs1=letters[1:3], gs2=letters[3:4], gs3=letters[4:5]))
myGmtList[[1]]
```
Index

[.GmtList, 44
[[.GmtList, 45

absLog1op, 3
appendGmtList, 4
as.GmtList, 5

BaseIndexList-class, 5

entropy, 6, 7, 8, 29
entropyDiversity, 7, 29
entropySpecificity, 8

filterBySize, 9
filterPmat, 9

getLeadingEdgeIndexFromMatrix
  (getLeadingEdgeIndexFromVector), 10
getLeadingEdgeIndexFromVector, 10

gini, 11
GmtList, 12
GmtList-class, 13
gmtlist2signedGenesets, 13, 28
gsDesc, 14, 27
gsGeneCount, 15
gsGenes, 15, 27
gsName, 16, 27
gsNamespace, 16, 27
gsNamespace<-, 17
gsSize (gsGeneCount), 15

hasNamespace, 17

IndexList, 18
IndexList, list-method (IndexList), 18
IndexList, logical-method (IndexList), 18
IndexList, numeric-method (IndexList), 18
IndexList-class, 19
isValidBaseIndexList, 19
isValidGmtList, 20

isValidIndexList, 20
isValidSignedGenesets, 21
isValidSignedIndexList, 21

matchGenes, 22
matchGenes, character, character-method
  (matchGenes), 22
matchGenes, character, DGEList-method
  (matchGenes), 22
matchGenes, character, eSet-method
  (matchGenes), 22
matchGenes, character, matrix-method
  (matchGenes), 22
matchGenes, GmtList, character-method
  (matchGenes), 22
matchGenes, GmtList, DGEList-method
  (matchGenes), 22
matchGenes, GmtList, eSet-method
  (matchGenes), 22
matchGenes, GmtList, matrix-method
  (matchGenes), 22
matchGenes, SignedGenesets, character-method
  (matchGenes), 22
matchGenes, SignedGenesets, DGEList-method
  (matchGenes), 22
matchGenes, SignedGenesets, eSet-method
  (matchGenes), 22
matchGenes, SignedGenesets, matrix-method
  (matchGenes), 22

offset, 24
offset, BaseIndexList-method (offset), 24
offset-set (offset<-), 25
offset<-, 25
offset<-, IndexList, numeric-method
  (offset<-), 25
offset<-, SignedIndexList, numeric-method
  (offset<-), 25

prettySigNames, 25
INDEX

readCurrentSignatures, 26
readGmt, 26, 27, 36
readSignedGmt, 28

sampleSpecialization, 7, 29
setDescAsNamespace, 30
setNamespace, 30, 30
show,GmtList-method, 31
show,IndexList-method, 31
show,SignedGenesets-method, 32
show,SignedIndexList-method, 32
SignedGenesets, 33
SignedGenesets-class, 33
SignedIndexList, 34
SignedIndexList,list-method (SignedIndexList), 34
SignedIndexList-class, 35
simplifyMatrix, 35

uniqGenesetsByNamespace, 36

valTypes, 37

wmwLeadingEdge, 37
wmwTest, 11, 38, 38
wmwTest,ANY,list-method (wmwTest), 38
wmwTest,ANY,logical-method (wmwTest), 38
wmwTest,ANY,numeric-method (wmwTest), 38
wmwTest,eSet,GmtList-method (wmwTest), 38
wmwTest,eSet,list-method (wmwTest), 38
wmwTest,eSet,logical-method (wmwTest), 38
wmwTest,eSet,numeric-method (wmwTest), 38
wmwTest,eSet,SignedGenesets-method (wmwTest), 38
wmwTest,eSet,SignedIndexList-method (wmwTest), 38
wmwTest,matrix,GmtList-method (wmwTest), 38
wmwTest,matrix,IndexList-method (wmwTest), 38
wmwTest,matrix,SignedGenesets-method (wmwTest), 38
wmwTest,matrix,SignedIndexList-method (wmwTest), 38
wmwTest,numeric,IndexList-method (wmwTest), 38
wmwTest,numeric,SignedIndexList-method (wmwTest), 38
wmwTestInR, 43