

# Package ‘SeqGSEA’

January 19, 2021

**Type** Package

**Title** Gene Set Enrichment Analysis (GSEA) of RNA-Seq Data: integrating differential expression and splicing

**Version** 1.30.0

**Date** 2018-12-12

**Author** Xi Wang <Xi.Wang@newcastle.edu.au>

**Maintainer** Xi Wang <Xi.Wang@dkfz.de>

**Description** The package generally provides methods for gene set enrichment analysis of high-throughput RNA-Seq data by integrating differential expression and splicing. It uses negative binomial distribution to model read count data, which accounts for sequencing biases and biological variation. Based on permutation tests, statistical significance can also be achieved regarding each gene's differential expression and splicing, respectively.

**License** GPL (>= 3)

**Depends** Biobase, doParallel, DESeq

**Imports** methods, biomaRt

**Suggests** easyRNASeq, GenomicRanges

**biocViews** Sequencing, RNASeq, GeneSetEnrichment, GeneExpression, DifferentialExpression, DifferentialSplicing, ImmunoOncology

**git\_url** <https://git.bioconductor.org/packages/SeqGSEA>

**git\_branch** RELEASE\_3\_12

**git\_last\_commit** 8c89028

**git\_last\_commit\_date** 2020-10-27

**Date/Publication** 2021-01-18

## R topics documented:

SeqGSEA-package . . . . .	3
calES . . . . .	5
calES.perm . . . . .	6
convertEnsembl2Symbol . . . . .	6
convertSymbol2Ensembl . . . . .	7

counts-methods . . . . .	8
DENBStat4GSEA . . . . .	8
DENBStatPermut4GSEA . . . . .	9
DENBTest . . . . .	10
DEpermutePval . . . . .	11
DEscore . . . . .	12
DSpermute4GSEA . . . . .	12
DSpermutePval . . . . .	13
DSresultExonTable . . . . .	14
DSresultGeneTable . . . . .	15
estiExonNBstat . . . . .	16
estiGeneNBstat . . . . .	17
exonID . . . . .	18
exonTestability . . . . .	18
geneID . . . . .	19
geneList . . . . .	20
genePermuteScore . . . . .	21
geneScore . . . . .	22
geneSetDescs . . . . .	23
geneSetNames . . . . .	23
geneSetSize . . . . .	24
geneTestability . . . . .	25
genpermuteMat . . . . .	26
getGeneCount . . . . .	26
GSEAResultTable . . . . .	27
GSEnrchAnalyze . . . . .	28
GS_example . . . . .	29
label . . . . .	29
loadExonCountData . . . . .	30
loadGenesets . . . . .	31
newGeneSets . . . . .	32
newReadCountSet . . . . .	33
normES . . . . .	34
normFactor . . . . .	34
plotES . . . . .	35
plotGeneScore . . . . .	36
plotSig . . . . .	37
plotSigGeneSet . . . . .	38
rankCombine . . . . .	39
RCS_example . . . . .	40
ReadCountSet-class . . . . .	40
runDESeq . . . . .	41
runSeqGSEA . . . . .	42
scoreNormalization . . . . .	44
SeqGeneSet-class . . . . .	45
signifES . . . . .	47
size . . . . .	47
subsetByGenes . . . . .	48
topDEGenes . . . . .	49
topDSExons . . . . .	50
topDSGenes . . . . .	51
topGeneSets . . . . .	51

<i>SeqGSEA-package</i>	3
writeScores . . . . .	52
writeSigGeneSet . . . . .	53

<b>Index</b>	<b>55</b>
--------------	-----------

---

SeqGSEA-package	<i>SeqGSEA: a Bioconductor package for gene set enrichment analysis of RNA-Seq data</i>
-----------------	-----------------------------------------------------------------------------------------

---

## Description

SeqGSEA is an R package for gene set enrichment analysis of RNA-Seq data with the ability to integrate differential expression and differential splice in functional analysis.

## Details

Package: SeqGSEA  
 Type: Package  
 License: GPL (>= 3)

A User's Guide is available as well as the usual help page documentation for each of the individual functions.

The most useful functions are listed below:

\* ReadCountSet class

- [ReadCountSet-class](#)
- [ReadCountSet](#)
- [exonID](#)
- [geneID](#)
- [counts-methods](#)
- [label](#)
- [subsetByGenes](#)

\* SeqGeneSet class

- [SeqGeneSet-class](#)
- [geneSetDescs](#)
- [geneSetNames](#)
- [geneSetSize](#)
- [size](#)

\* Load data

- [newReadCountSet](#)
- [loadExonCountData](#)
- [newGeneSets](#)
- [loadGenesets](#)

\* DE analysis

- `getGeneCount`
- `runDESeq`
- `DENBStat4GSEA`
- `DENBStatPermut4GSEA`
- `DENBTest`
- `DEpermutePval`

\* DS analysis

- `DSpermute4GSEA`
- `DSpermutePval`
- `exonTestability`
- `geneTestability`
- `estiExonNBstat`
- `estiGeneNBstat`

\* GSEA main

- `GSEnrichAnalyze`
- `calES`
- `calES.perm`
- `genePermuteScore`
- `geneScore`
- `rankCombine`
- `normES`
- `normFactor`
- `scoreNormalization`
- `signifES`

\* Result tables

- `GSEarResultTable`
- `DSresultExonTable`
- `DSresultGeneTable`
- `topDEGenes`
- `topDSExons`
- `topDSGenes`
- `topGeneSets`

\* Result displays

- `plotES`
- `plotGeneScore`
- `plotSig`
- `plotSigGeneSet`

- [writeSigGeneSet](#)

\* Miscellaneous

- [genpermuteMat](#)
- [convertEnsembl2Symbol](#)
- [convertSymbol2Ensembl](#)

### Author(s)

Xi Wang and Murray J. Cairns

Maintainer: Xi Wang <xi.wang@newcastle.edu.au>

### References

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

---

calES

*Calculate running enrichment scores of gene sets*

---

### Description

This is an internal function to calculate running enrichment scores of each gene set in the SeqGeneSet object specified

### Usage

```
calES(gene.set, gene.score, weighted.type = 1)
```

### Arguments

`gene.set` a SeqGeneSet object.  
`gene.score` a vector of gene scores corresponding to the geneList slot of gene.set.  
`weighted.type` gene score weight type.

### Author(s)

Xi Wang, xi.wang@newcastle.edu.au

### See Also

[GSEnrichAnalyze](#), [calES.perm](#),

### Examples

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(GS_example, package="SeqGSEA")
rES <- calES(GS_example, gene.score)
rES[1,]
```

---

`calES.perm`*Calculate enrichment scores for gene sets in the permutation data sets*

---

**Description**

This is an internal function to calculate enrichment scores for gene sets in the permutation data sets.

**Usage**

```
calES.perm(gene.set, gene.score.perm, weighted.type = 1)
```

**Arguments**

`gene.set` a SeqGeneSet object.  
`gene.score.perm` a matrix of gene scores on the permutation data sets.  
`weighted.type` gene score weight type.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [calES](#),

**Examples**

```
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
ES.perm <- calES.perm(GS_example, gene.score.perm)
ES.perm[1:5,1:5]
```

---

`convertEnsembl2Symbol` *Convert ensembl gene IDs to gene symbols*

---

**Description**

Convert ensembl gene IDs to gene symbols

**Usage**

```
convertEnsembl2Symbol(ensembl.genes)
```

**Arguments**

`ensembl.genes` ensembl gene ID(s).

**Value**

A 2-column matrix showing the correspondence of ensembl gene IDs and gene symbols.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[convertSymbol2Ensembl](#)

**Examples**

```
## Not run:  
convertEnsembl2Symbol("ENSG00000162946") #DISC1  
  
## End(Not run)
```

---

`convertSymbol2Ensembl` *Convert gene symbols to ensembl gene IDs*

---

**Description**

Convert gene symbols to ensembl gene IDs

**Usage**

```
convertSymbol2Ensembl(symbols)
```

**Arguments**

symbols            gene symbol(s).

**Value**

A 2-column matrix showing the correspondence of gene symbols and ensembl gene IDs.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[convertEnsembl2Symbol](#)

**Examples**

```
## Not run:  
convertSymbol2Ensembl("DISC1") #ENSG00000162946  
  
## End(Not run)
```

---

counts-methods	<i>Accessors for the 'counts' slot of a ReadCountSet object.</i>
----------------	------------------------------------------------------------------

---

**Description**

Accessors for the 'counts' slot of a ReadCountSet object.

**Usage**

```
## S4 method for signature 'ReadCountSet'
counts(object)
## S4 replacement method for signature 'ReadCountSet,matrix'
counts(object) <- value
```

**Arguments**

object	a ReadCountSet object
value	a matrix of read counts

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**Examples**

```
data(RCS_example, package="SeqGSEA")
readCounts <- counts(RCS_example)
head(readCounts)
```

---

DENBStat4GSEA	<i>Calculate NB-statistics quantifying differential expression for each gene</i>
---------------	----------------------------------------------------------------------------------

---

**Description**

Calculate NB-statistics quantifying differential expression between two groups of samples compared. The results will be used for GSEA run. Comparing with [DENBTest](#), this function will not calculate NB test p-values.

This function only works with two-group comparison.

**Usage**

```
DENBStat4GSEA(cds)
```

**Arguments**

cds	A CountDataSet object with size factors and dispersion parameters estimated. Recommended to take the output of <a href="#">runDESeq</a> .
-----	-------------------------------------------------------------------------------------------------------------------------------------------



**Value**

A data frame containing each gene's expression means and variances in each group, and each gene's DE NB-statistics.

**Note**

The results with the output of [DENBStatPermut4GSEA](#) can also be used to run [DEpermutePval](#).

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

**See Also**

[DENBTest](#), [runDESeq](#), [DENBStatPermut4GSEA](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
geneCounts <- getGeneCount(RCS_example)
label <- label(RCS_example)
DEG <- runDESeq(geneCounts, label)
DEGres <- DENBStat4GSEA(DEG)
head(DEGres)
```

---

DENBStatPermut4GSEA	<i>Calculate NB-statistics quantifying DE for each gene in the permutation data sets</i>
---------------------	------------------------------------------------------------------------------------------

---

**Description**

Calculate NB-statistics quantifying differential expression for each gene in the permutation data sets. The results will be used for GSEA run.

**Usage**

```
DENBStatPermut4GSEA(DEG, permuteMat)
```

**Arguments**

DEG                    a `CountDataSet` object, can be the output of [runDESeq](#).  
permuteMat            a permutation matrix generated by [genpermuteMat](#).

**Value**

A matrix of NB-statistics. Each row corresponds to each gene, and each column to each permutation.

**Note**

The results with the output of [DENBStat4GSEA](#) can also be used to run [DEpermutePval](#).

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

**See Also**

[DENBStat4GSEA](#), [runDESeq](#), [DEpermutePval](#), [genpermuteMat](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
geneCounts <- getGeneCount(RCS_example)
label <- label(RCS_example)
DEG <- runDESeq(geneCounts, label)
DEpermNBstat <- DENBStatPermut4GSEA(DEG, permuteMat)
DEpermNBstat[1:10,1:10]
```

---

DENBTest

*Perform negative binomial exact test for differential expression*

---

**Description**

Perform negative binomial exact test for differential expression - a modified version of `nbinomTest` in `DESeq` package.

**Usage**

```
DENBTest(cds)
```

**Arguments**

`cds` A `CountDataSet` object with size factors and dispersion parameters estimated. Recommended to take the output of [runDESeq](#).

**Value**

A data frame of the test results. Information contains mean expression values, NB-statistics, (log) fold-changes, p-values, and adjusted p-values.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

## References

Anders, S. and Huber, W. (2010) Differential expression analysis for sequence count data, *Genome Biol*, 11, R106.

## See Also

[runDESeq](#), [DENBStat4GSEA](#)

## Examples

```
data(RCS_example, package="SeqGSEA")
geneCounts <- getGeneCount(RCS_example)
label <- label(RCS_example)
DEG <- runDESeq(geneCounts, label)
DEGres <- DENBTest(DEG)
head(DEGres)
```

---

DEpermutePval

*Permutation for p-values in differential expression analysis*

---

## Description

Calculate permutation p-values in differential expression analysis for each genes.

## Usage

```
DEpermutePval(DEGres, permuteNBstat)
```

## Arguments

DEGres            the output of [DENBStat4GSEA](#).  
permuteNBstat    the output of [DENBStatPermut4GSEA](#).

## Value

A data frame containing the expression means and variances for each gene in each group compared, and NB-stats, permutation p-values and adjusted p-values for each gene.

## Author(s)

Xi Wang, xi.wang@newcastle.edu.au

## See Also

[runDESeq](#), [DENBStat4GSEA](#), [DENBStatPermut4GSEA](#), [DENBTest](#)

**Examples**

```

data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
geneCounts <- getGeneCount(RCS_example)
label <- label(RCS_example)
DEG <- runDESeq(geneCounts, label)
DEGres <- DENBStat4GSEA(DEG)
DEpermNBstat <- DENBStatPermut4GSEA(DEG, permuteMat)
DEGres <- DEpermutePval(DEGres, DEpermNBstat)
head(DEGres)

```

DEscore

*Pre-calculated DE/DS scores***Description**

DEscore and DSscore are pre-calculated DE and DS scores, respectively; DEscore.perm and DSscore.perm are pre-calculated DE and DS scores on the permutation data sets, respectively; They are used in examples of the SeqGSEA package. Note that these scores are of no meaning but to demonstrate the usage of functions.

**Usage**

```

data("DEscore")
data("DEscore.perm")
data("DSscore")
data("DSscore.perm")

```

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. BMC Bioinformatics, 14(Suppl 5):S16.

DSpermute4GSEA

*Compute NB-statistics quantifying differential splicing on the permutation data set.***Description**

This function is to calculate NB-statistics quantifying differential splicing for each gene on each permutation data set. The results will be used for GSEA run as DS background.

**Usage**

```
DSpermute4GSEA(RCS, permuteMat)
```

**Arguments**

RCS                    a ReadCountSet object after running [exonTestability](#).  
permuteMat            a permutation matrix generated by [genpermuteMat](#).

**Details**

Parallel running configuration: TODO

**Value**

A ReadCountSet object with slot permute\_NBstat\_gene updated.

**Note**

Please run [exonTestability](#) before run this function.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. BMC Bioinformatics, 14(Suppl 5):S16.

**See Also**

[exonTestability](#), [genpermuteMat](#), [DENBStatPermut4GSEA](#), [DSpermutePval](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- DSpermute4GSEA(RCS_example, permuteMat)
head(RCS_example@permute_NBstat_gene)
```

---

DSpermutePval

*Permutation for p-values in differential splicing analysis*

---

**Description**

Calculate permutation p-values in differential splicing analysis.

**Usage**

```
DSpermutePval(RCS, permuteMat)
```

**Arguments**

RCS                    a ReadCountSet object after running [estiExonNBstat](#) and [estiGeneNBstat](#).  
permuteMat            a permutation matrix generated by [genpermuteMat](#).

**Details**

Permutation p-values are computed based on NB-statistics for comparison of the studied groups and NB-statistics from the permutation data sets.

**Value**

A ReadCountSet object with slots permute\_NBstat\_exon, permute\_NBstat\_gene, featureData, and featureData\_gene updated.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. BMC Bioinformatics, 14(Suppl 5):S16.

**See Also**

[estiExonNBstat](#), [estiGeneNBstat](#), [genpermuteMat](#), [DSpermute4GSEA](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermutePval(RCS_example, permuteMat)
head(DSresultExonTable(RCS_example))
head(DSresultGeneTable(RCS_example))
```

---

DSresultExonTable      *Form a table for DS analysis results at the Exon level*

---

**Description**

Form a table for differential splicing analysis results at the Exon level.

**Usage**

```
DSresultExonTable(RCS)
```

**Arguments**

RCS                      A ReadCountSet object with [DSpermutePval](#) done.

**Details**

A data frame containing each exon's NB-statistics, p-values and adjusted p-values for differential splicing analysis.

**Value**

A matrix containing exon DS analysis results, including testability, NBstats, p-values and adjusted p-values.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[DSresultGeneTable](#), [DSpermutepval](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermutepval(RCS_example, permuteMat)
head(DSresultExonTable(RCS_example))
```

---

DSresultGeneTable	<i>Form a table for DS analysis results at the gene level</i>
-------------------	---------------------------------------------------------------

---

**Description**

Form a table for differential splicing analysis results at the gene level.

**Usage**

```
DSresultGeneTable(RCS)
```

**Arguments**

RCS                    A ReadCountSet object with [DSpermutepval](#) done.

**Value**

A data frame containing each gene's NB-statistics, p-values and adjusted p-values for differential splicing analysis.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[DSresultExonTable](#), [DSpermutepval](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermutepval(RCS_example, permuteMat)
head(DSresultGeneTable(RCS_example))
```

---

estiExonNBstat	<i>Calculate NB-statistics quantifying differential splicing for individual exons</i>
----------------	---------------------------------------------------------------------------------------

---

**Description**

Calculate NB-statistics quantifying differential splicing for individual exons between two groups of samples compared.

**Usage**

```
estiExonNBstat(RCS)
```

**Arguments**

RCS                    a ReadCountSet object after running exonTestability.

**Value**

A ReadCountSet object with the slot featureData updated.

**Note**

Please run [exonTestability](#) before you run this function.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Weichen Wang, Zhiyi Qin, Zhixing Feng, Xi Wang and Xuegong Zhang (2013). Identifying differentially spliced genes from two groups of RNA-seq samples. *Gene*, 518(1):164-170.

**See Also**

[exonTestability](#), [estiGeneNBstat](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
RCS_example <- exonTestability(RCS_example, cutoff=5)
RCS_example <- estiExonNBstat(RCS_example)
head(fData(RCS_example))
```



---

estiGeneNBstat	<i>Calculate NB-statistics quantifying differential splicing for each gene</i>
----------------	--------------------------------------------------------------------------------

---

### Description

Calculate NB-statistics quantifying differential splicing for each gene between two groups of samples compared. The results will be used for GSEA run (as DS-scores).

### Usage

```
estiGeneNBstat(RCS)
```

### Arguments

RCS                    a ReadCountSet object after running estiExonNBstat.

### Value

A ReadCountSet object with slot featureData\_gene updated.

### Note

Please run [estiExonNBstat](#) before run this function.

### Author(s)

Xi Wang, xi.wang@newcastle.edu.au

### References

Weichen Wang, Zhiyi Qin, Zhixing Feng, Xi Wang and Xuegong Zhang (2013). Identifying differentially spliced genes from two groups of RNA-seq samples. *Gene*, 518(1):164-170.

### See Also

[estiExonNBstat](#)

### Examples

```
data(RCS_example, package="SeqGSEA")
RCS_example <- exonTestability(RCS_example, cutoff=5)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
head(RCS_example@featureData_gene)
```

---

exonID	<i>Accessor to the exonID slot of ReadCountSet objects</i>
--------	------------------------------------------------------------

---

**Description**

Accessor to the exonID slot of ReadCountSet objects

**Usage**

```
exonID(RCS)
exonID(RCS) <- value
```

**Arguments**

RCS	a ReadCountSet object
value	a vector of exon IDs

**Value**

A character vector of exon IDs; or a ReadCountSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[newReadCountSet](#), [geneID](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
exonID(RCS_example)
```

---

exonTestability	<i>Check exon testability</i>
-----------------	-------------------------------

---

**Description**

Check exon testability, filtering out exons with very few (default: 5) read counts

**Usage**

```
exonTestability(RCS, cutoff = 5)
```

**Arguments**

RCS	a ReadCountSet object.
cutoff	exons with read counts less than this cutoff are to be marked as untestable.

**Value**

a ReadCountSet object with slot fData updated.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[geneTestability](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
RCS_example <- exonTestability(RCS_example, cutoff=5)
head(fData(RCS_example))
```

---

geneID	<i>Accessor to the geneID slot of ReadCountSet objects</i>
--------	------------------------------------------------------------

---

**Description**

Accessor to the geneID slot of ReadCountSet objects

**Usage**

```
geneID(RCS)
geneID(RCS) <- value
```

**Arguments**

RCS	a ReadCountSet object
value	a vector of gene IDs

**Value**

A character vector of gene IDs, which can be duplicated; or a ReadCountSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[newReadCountSet](#), [exonID](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
geneID(RCS_example)
```

---

geneList	<i>Get the gene list in a SeqGeneSet object</i>
----------	-------------------------------------------------

---

**Description**

Get the gene list in a SeqGeneSet object

**Usage**

```
geneList(GS)
```

**Arguments**

GS                    A SeqGeneSet object.

**Details**

TBA

**Value**

A vector of gene IDs.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[loadGenesets](#), [SeqGeneSet-class](#)

**Examples**

```
##  
gs <- newGeneSets(GS=list(1:10, 6:15, 11:20),  
                  geneList=paste("Gene", 1:22, sep=""),  
                  GSNames=c("gs1", "gs2", "gs3"),  
                  GSDescs=c("test1", "test2", "test3"),  
                  name="gs examples")  
  
geneList(gs)  
## End
```

---

genePermuteScore	<i>Calculate gene scores on permutation data sets</i>
------------------	-------------------------------------------------------

---

**Description**

Calculate gene scores on permutation data sets

**Usage**

```
genePermuteScore(DEscoreMat, DSscoreMat = NULL, method = c("linear", "quadratic", "rank"),  
                 DEweight = 0.5)
```

**Arguments**

DEscoreMat	normalized DE scores on permutation data sets.
DSscoreMat	normalized DS scores on permutation data sets.
method	one of the integration methods: linear, quadratic, or rank; default: linear.
DEweight	any number between 0 and 1 (included), the weight of differential expression scores (the weight for differential splice is (1-DEweight)).

**Details**

The integration methods including "linear", "quadratic", and "rank" are detailed in Wang and Cairns (2013). Here the rank method refers only to the method using data-set-specific ranks.

For DE-only analysis, just specify DEweight to be 1, and the DSscoreMat value can be NULL.

**Value**

A gene score matrix.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

**See Also**

[geneScore](#)

**Examples**

```
data(DEscore.perm, package="SeqGSEA")  
data(DScore.perm, package="SeqGSEA")  
# linear combination with weight for DE 0.3  
gene.score.perm <- genePermuteScore(DEscore.perm, DScore.perm, method="linear", DEweight=0.3)  
# DE only analysis  
gene.score.perm <- genePermuteScore(DEscore.perm, DEweight=1)
```

---

`geneScore`*Calculate gene scores by integrating DE and DS scores*

---

**Description**

Calculate gene scores by integrating DE and DS scores

**Usage**

```
geneScore(DEscore, DSscore = NULL, method = c("linear", "quadratic", "rank"), DEweight = 0.5)
```

**Arguments**

DEscore	normalized DE scores.
DSscore	normalized DS scores.
method	one of the integration methods: linear, quadratic, or rank; default: linear.
DEweight	any number between 0 and 1 (included), the weight of differential expression scores (the weight for differential splice is (1-DEweight)).

**Details**

The integration methods including "linear", "quadratic", and "rank" are detailed in Wang and Cairns (2013). Here the rank method refers only to the method using data-set-specific ranks.

For DE-only analysis, just specify DEweight to be 1, and the DSscore value can be NULL.

**Value**

A vector of gene scores.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

**See Also**

[genePermuteScore](#)

**Examples**

```
data(DEscore, package="SeqGSEA")
data(DScore, package="SeqGSEA")
# linear combination with weight for DE 0.3
gene.score <- geneScore(DEscore, DScore, method="linear", DEweight = 0.3)
# DE only analysis
gene.score <- geneScore(DEscore, DEweight = 1)
```

---

geneSetDescs	<i>Get the descriptions of gene sets in a SeqGeneSet object</i>
--------------	-----------------------------------------------------------------

---

**Description**

Get the descriptions of gene sets in a SeqGeneSet object

**Usage**

```
geneSetDescs(GS)
```

**Arguments**

GS                    a SeqGeneSet object.

**Details**

Gene sets with size less than GSSizeMin or more than GSSizeMax are not included.

**Value**

A vector of descriptions of each gene set in the SeqGeneSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[geneSetNames](#), [geneSetSize](#), [SeqGeneSet-class](#), [loadGenesets](#)

**Examples**

```
data(GS_example, package="SeqGSEA")
geneSetDescs(GS_example)
```

---

geneSetNames	<i>Get the names of gene set in a SeqGeneSet object</i>
--------------	---------------------------------------------------------

---

**Description**

Get the names of gene set in a SeqGeneSet object

**Usage**

```
geneSetNames(GS)
```

**Arguments**

GS                    a SeqGeneSet object.

**Details**

Gene sets with size less than GSSizeMin or more than GSSizeMax are not included.

**Value**

A vector of gene set names in this SeqGeneSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[geneSetDescs](#), [geneSetSize](#), [SeqGeneSet-class](#), [loadGenesets](#)

**Examples**

```
data(GS_example, package="SeqGSEA")
geneSetNames(GS_example)
```

---

geneSetSize

*Get the numbers of genes in each gene set in a SeqGeneSet object*

---

**Description**

Get the numbers of genes in each gene set in a SeqGeneSet object

**Usage**

```
geneSetSize(GS)
```

**Arguments**

GS                    a SeqGeneSet object.

**Details**

Gene sets with size less than GSSizeMin or more than GSSizeMax are not included.

**Value**

A vector of integers indicating the number of genes in each gene set in this SeqGeneSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[geneSetNames](#), [geneSetDescs](#), [SeqGeneSet-class](#), [loadGenesets](#)



**Examples**

```
data(GS_example, package="SeqGSEA")
geneSetSize(GS_example)
```

---

geneTestability	<i>Check gene testability</i>
-----------------	-------------------------------

---

**Description**

This function is to determine each gene's testability. A gene is testable if at least one of its exons are testable.

**Usage**

```
geneTestability(RCS)
```

**Arguments**

RCS                    a `ReadCountSet` object after exon testability checked, usually the output of [exonTestability](#).

**Details**

This result can applied to filter out genes not expressed.

**Value**

A logical vector indicating which genes are testable, i.e., having at least one exon testable.

**Note**

Please run [exonTestability](#) before run this function.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[exonTestability](#), [subsetByGenes](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
RCS_example <- exonTestability(RCS_example, cutoff=5)
geneTestable <- geneTestability(RCS_example)
head(geneTestable)
```

---

genpermuteMat	<i>Generate permutation matrix</i>
---------------	------------------------------------

---

**Description**

Generate permutation matrix from ReadCountSet objects or from label vectors.

**Usage**

```
genpermuteMat(obj, times = 1000, seed = NULL)
```

**Arguments**

obj	a ReadCountSet object or a label vector. This function needs the original sample label information to generate permutation matrix.
times	an integer indication the times of permutation.
seed	an integer or NULL, to produce the random seed (an integer vector) for generating random permutation matrix: the same seed generates the same permutation matrix, which is introduced for reproducibility.

**Value**

A sample label shuffled matrix, rows corresponding to samples and columns for each permutation.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[DSpermute4GSEA](#), [DENBStatPermut4GSEA](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10, seed=0)
RCS_example <- exonTestability(RCS_example)
RCS_example <- DSpermute4GSEA(RCS_example, permuteMat)
```

---

getGeneCount	<i>Calculate read counts of genes from a ReadCountSet object</i>
--------------	------------------------------------------------------------------

---

**Description**

Calculate read counts of genes from a ReadCountSet object

**Usage**

```
getGeneCount(RCS)
```

**Arguments**

RCS                    a ReadCountSet object

**Details**

This function can be used to get gene read counts from exon read counts.

**Value**

a matrix of gene read counts for each gene (row) and each sample (col).

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[loadExonCountData](#), [runDESeq](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
geneCounts <- getGeneCount(RCS_example)
```

---

GSEAResultTable            *Form a table for GSEA results*

---

**Description**

Form a table for GSEA results.

**Usage**

```
GSEAResultTable(gene.set, GSDesc = FALSE)
```

**Arguments**

gene.set                a SeqGeneSet object after running [GSEnrichAnalyze](#).  
GSDesc                 logical indicating whether to output gene set descriptions. default: FALSE

**Value**

A data frame containing columns of GSName, GSSize, ES, ES.pos, pval, FDR, and FWER.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [topGeneSets](#)

**Examples**

```

data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
head(GSEAResultTable(GS_example))

```

---

GSEnrichAnalyze	<i>Main function of gene set enrichment analysis</i>
-----------------	------------------------------------------------------

---

**Description**

The main function of gene set enrichment analysis

**Usage**

```
GSEnrichAnalyze(gene.set, gene.score, gene.score.perm, weighted.type = 1)
```

**Arguments**

<code>gene.set</code>	a SeqGeneSet object.
<code>gene.score</code>	a vector of integrated gene scores in the same order as genes listed in the <code>gene.list</code> slot of <code>gene.set</code> .
<code>gene.score.perm</code>	a matrix of integrated gene scores on permutation data sets; row: genes; col: permutation.
<code>weighted.type</code>	weight type for gene scores; default: 1.

**Value**

A SeqGeneSet object with many slots updated, such as `GSEA.ES` and `GSEA.pval`.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

**See Also**

[normES](#), [signifES](#)

**Examples**

```

data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
topGeneSets(GS_example, 5)

```

---

GS\_example

*SeqGeneSet object example*


---

**Description**

An exemplified SeqGeneSet object to demonstrate functions in the SeqGSEA package. This object was generated with collection #6 (C6) gene sets of the Molecular Signatures Database (MSigDB) v3.1.

**Usage**

```
data("GS_example")
```

**References**

Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., and Mesirov, J. P. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc Natl Acad Sci USA*, 102(43): 15545-50.

---

label

*Get the labels of samples in a ReadCountSet object*


---

**Description**

Get the labels of samples in a ReadCountSet object

**Usage**

```
label(RCS)
```

**Arguments**

RCS                    a ReadCountSet object

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**[newReadCountSet](#)**Examples**

```
data(RCS_example, package="SeqGSEA")
label(RCS_example)
```

---

loadExonCountData	<i>Load Exon Count Data</i>
-------------------	-----------------------------

---

**Description**

This function is used to load (sub-)exon count data. Exon count data can be got by the Python script `count_in_exons.py`.

**Usage**

```
loadExonCountData(case.files, control.files)
```

**Arguments**

`case.files` a character vector containing the exon count file names for case samples  
`control.files` a character vector containing the exon count file names for control samples

**Details**

You may need the Python script `count_in_exons.py` (released with this package) to generate your exon count files from read mapping results (say BAM files). The detailed usage can be obtained by simply typing `python \path\to\count_in_exons.py`. Users can also use other scripts or software for exon read counting.

The format of the exon count file is:

```
GeneName1:001[tab]Count11
GeneName1:002[tab]Count12
...
GeneName1:00N[tab]Count1N
GeneName2:001[tab]Count21
...
```

**Value**

This function returns a `ReadCountSet` object.

**Author(s)**

Xi Wang, [xi.wang@newcastle.edu.au](mailto:xi.wang@newcastle.edu.au)

**See Also**

[newReadCountSet](#), [ReadCountSet-class](#)

**Examples**

```

library(SeqGSEA)
dat.dir = system.file("extdata", package="SeqGSEA", mustWork=TRUE)
case.pattern <- "^SC"
ctrl.pattern <- "^SN"
case.files <- dir(dat.dir, pattern=case.pattern, full.names = TRUE)
control.files <- dir(dat.dir, pattern=ctrl.pattern, full.names = TRUE)

## Not run:
RCS <- loadExonCountData(case.files, control.files)
RCS

## End(Not run)

```

loadGenesets

*Load gene sets from files***Description**

This function is to load annotation of gene sets from files. The files are in the format of Molecular Signatures Database (MSigDB), and those files can be downloaded at <http://www.broadinstitute.org/gsea/msigdb/index.jsp>.

**Usage**

```
loadGenesets(geneset.file, geneIDs, geneID.type = c("gene.symbol", "ensembl"),
             genesetsize.min = 5, genesetsize.max = 1000, singleCell = FALSE)
```

**Arguments**

geneset.file	the file containing the gene set annotation.
geneIDs	gene IDs that have expression values in the studied data set.
geneID.type	indicating the type of gene IDs, gene symbol or emsembl gene IDs.
genesetsize.min	the minimum number of genes in a gene set that will be treated in the analysis.
genesetsize.max	the maximum number of genes in a gene set that will be treated in the analysis.
singleCell	logical, whether to creat a SeqGeneSet object for scGSEA.

**Details**

TBA

**Value**

A SeqGeneSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[newGeneSets](#), [SeqGeneSet-class](#)

**Examples**

```
## Not run:
data(RCS_example, package="SeqGSEA")
geneIDs <- geneID(RCS_example)
geneID.type <- "ensembl"
geneset.file <- system.file("extdata", "gs_symb.txt", package="SeqGSEA", mustWork=TRUE)
GS <- loadGenesets(geneset.file, geneIDs, geneID.type = geneID.type)
GS

## End(Not run)
```

---

newGeneSets

*Initialize a new SeqGeneSet object*

---

**Description**

This is an internal function to generate a new SeqGeneSet object.

**Usage**

```
newGeneSets(GS, GSNames, GSDescs, geneList, scGSEA = FALSE,
            name = NA_character_, sourceFile = NA_character_,
            GSSizeMin = 5, GSSizeMax = 1000)
```

**Arguments**

GS	a list, each element is an integer vector, indicating the indexes of genes in each gene set. See <i>Details</i> below.
GSNames	a character string vector, each is the name of each gene set.
GSDescs	a character string vector, each is the description of each gene set.
geneList	a character string vector of gene IDs. See <i>Details</i> below.
scGSEA	logical, if this object used for scGSEA.
name	the name of this category of gene sets.
sourceFile	the source file name of this category of gene sets.
GSSizeMin	the minimum number of genes in a gene set to be analyzed. Default: 5
GSSizeMax	the maximum number of genes in a gene set to be analyzed. Default: 1000

**Details**

TBA

**Value**

A SeqGeneSet object.



**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[loadGenesets](#), [SeqGeneSet-class](#)

**Examples**

```
##
gs <- newGeneSets(GS=list(1:10, 6:15, 11:20),
                  geneList=paste("Gene", 1:22, sep=""),
                  GSNames=c("gs1", "gs2", "gs3"),
                  GSDescs=c("test1", "test2", "test3"),
                  name="gs examples")

gs
## End
```

---

newReadCountSet	<i>Generate a new ReadCountSet object</i>
-----------------	-------------------------------------------

---

**Description**

This is an internal function to generate a new ReadCountSet object.

**Usage**

```
newReadCountSet(readCounts, exonIDs, geneIDs)
```

**Arguments**

readCounts	a data frame, read counts for each exon of each samples. Must have colnames, which indicate the label of samples.
exonIDs	a character vector indicating exon IDs.
geneIDs	a character vector indicating gene IDs.

**Value**

A object of the ReadCountSet class.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[loadExonCountData](#), [ReadCountSet-class](#)

**Examples**

```

rcounts <- cbind(t(sapply(1:10, function(x) {rbinom(5, size=10, prob=runif(1))} ) ) ,
                t(sapply(1:10, function(x) {rbinom(5, size=10, prob=runif(1))} ) ) )
colnames(rcounts) <- c(paste("S", 1:5, sep=""), paste("C", 1:5, sep=""))
geneIDs <- c(rep("G1", 4), rep("G2", 6))
exonIDs <- c(paste("E", 1:4, sep=""), paste("E", 1:6, sep=""))
##
RCS <- newReadCountSet(rcounts, exonIDs, geneIDs)
RCS
## End

```

normES

*Normalize enrichment scores***Description**

This is an internal function to normalize enrichment scores. For advanced users only.

**Usage**

```
normES(gene.set)
```

**Arguments**

`gene.set` a SeqGeneSet object after running [cales](#) and [cales.perm](#).

**Value**

A SeqGeneSet object with ES scores normalized.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [signifES](#)

normFactor

*Get normalization factors for normalization DE or DS scores***Description**

Get normalization factors from permutation scores for normalization DE or DS scores

**Usage**

```
normFactor(permStat)
```

**Arguments**

permStat            a matrix of NB-statistics from permutation data sets, with row corresponding to genes and columns to permutations.

**Value**

A vector of normalization factors, each for one gene.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

**See Also**

[scoreNormalization](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermute4GSEA(RCS_example, permuteMat)
## (not run)
DSscore.normFac <- normFactor(RCS_example@permute_NBstat_gene)
DSscore <- scoreNormalization(RCS_example@featureData_gene$NBstat, DSscore.normFac)
DSscore.perm <- scoreNormalization(RCS_example@permute_NBstat_gene, DSscore.normFac)
## End (not run)
```

---

plotES

*Plot the distribution of enrichment scores*

---

**Description**

This function is to plot the distribution of enrichment scores, with comparison with permutation enrichment scores.

**Usage**

```
plotES(gene.set, pdf = NULL)
```

**Arguments**

gene.set            a SeqGeneSet object after running [GSEnrichAnalyze](#).  
pdf                  whether to save the plot to PDF file; if yes, provide the name of the PDF file.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [plotSigGeneSet](#)

**Examples**

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
plotES(GS_example)
```

---

plotGeneScore

*Plot gene (DE/DS) scores*

---

**Description**

This function is to plot gene scores, as well as DE scores and DS scores

**Usage**

```
plotGeneScore(score, perm.score = NULL, pdf = NULL,
              main = c("Overall", "Expression", "Splicing"))
```

**Arguments**

score	the gene/DE/DS score vector.
perm.score	a matrix of the corresponding gene/DE/DS scores on the permutation data sets.
pdf	if a PDF file name provided, plot will be save to that file.
main	the key words representing the type of scores that will be shown in the plot main title.

**Details**

The plot shows the ranked scores from the largest to the smallest. Lines also show the maximum and average scores, values shown on the top left.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**Examples**

```
data(DEscore, package="SeqGSEA")
plotGeneScore(DEscore, main="Expression")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
plotGeneScore(gene.score)
```

---

plotSig

*Plot showing SeqGeneSet's p-values/FDRs vs. NESs*

---

**Description**

The function is to generate a plot of p-values (FDRs) versus normalized enrichment scores (NES). It also shows the distribution of p-values (FDRs) in this gene set category.

**Usage**

```
plotSig(gene.set, pdf = NULL)
```

**Arguments**

`gene.set` a SeqGeneSet object after running [GSEnrichAnalyze](#).  
`pdf` whether to save the plot to PDF file; if yes, provide the name of the PDF file.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [plotSigGeneSet](#)

**Examples**

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
plotSig(GS_example)
```

---

plotSigGeneSet      *Plot gene set details*

---

### Description

This function is to generate a two-panel plot showing detailed information of the gene set specified. One panel is showing the running enrichment scores and the position where the ES appear. The other panel shows the significance level of the ES, comparing with permutation ESs.

### Usage

```
plotSigGeneSet(gene.set, i, gene.score, pdf = NULL)
```

### Arguments

gene.set	a SeqGeneSet object after running <a href="#">GSEnrichAnalyze</a> .
i	the i-th gene set in the SeqGeneSet object. <a href="#">topGeneSets</a> is useful to find the most significantly overrepresented gene set.
gene.score	the gene score vector containing gene scores for each gene.
pdf	whether to save the plot to PDF file; if yes, provide the name of the PDF file.

### Details

See [writeSigGeneSet](#), which writes the detailed gene set information to a file or to the screen.

### Author(s)

Xi Wang, xi.wang@newcastle.edu.au

### See Also

[GSEnrichAnalyze](#), [topGeneSets](#), [plotSig](#), [plotES](#), [writeSigGeneSet](#)

### Examples

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
topGeneSets(GS_example, n=5)
plotSigGeneSet(GS_example, 9, gene.score) # 9th gene set is the most significant one.
```

---

rankCombine	<i>Integration of differential expression and differential splice scores with a rank-based strategy</i>
-------------	---------------------------------------------------------------------------------------------------------

---

### Description

Integration of differential expression and differential splice scores with a rank-based strategy, which simultaneously integrates observed scores and permutation scores using the same ranks.

### Usage

```
rankCombine(DEscore, DSscore, DEscoreMat, DSscoreMat, DEweight = 0.5)
```

### Arguments

DEscore	differential expression scores, normalized.
DSscore	differential splice scores, normalized.
DEscoreMat	differential expression scores in permuted data sets, normalized.
DSscoreMat	differential splice scores in permuted data sets, normalized.
DEweight	any number between 0 and 1 (included), the weight of differential expression scores (so the weight for differential splice is (1-DEweight)).

### Details

This integration method is also known as integration with global ranks. See Wang and Cairns (2013) for details.

### Value

A list with two elements `geneScore` and `genePermuteScore`.

### Author(s)

Xi Wang, xi.wang@newcastle.edu.au

### References

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

### See Also

[geneScore](#), [genePermuteScore](#)

### Examples

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
combine <- rankCombine(DEscore, DSscore, DEscore.perm, DSscore.perm, DEweight=0.3)
gene.score <- combine$geneScore
gene.score.perm <- combine$genePermuteScore
```

RCS\_example

*ReadCountSet object example***Description**

An exemplified ReadCountSet object to demonstrate functions in the SeqGSEA package. This object is comprised of 20 samples across 5,000 exons, a part of the prostate cancer RNA-Seq data set from Kannan et al (2011). Please note that the count data in this example object is incomplete.

**Usage**

```
data("RCS_example")
```

**References**

Kannan, K., Wang, L., Wang, J., Ittmann, M. M., Li, W., and Yen, L. (2001). Recurrent chimeric RNAs enriched in human prostate cancer identified by deep sequencing. *Proc Natl Acad Sci USA*, 108(22): 9172-7.

ReadCountSet-class

*Class "ReadCountSet"***Description**

ReadCountSet class

**Objects from the Class**

Objects can be created by calls of the form [newReadCountSet](#).

**Slots**

featureData\_gene: Object of class "data.frame". Data for each genes.

permute\_NBstat\_exon: Object of class "matrix". NB statistics of exons on the permutation data sets.

permute\_NBstat\_gene: Object of class "matrix". NB statistics of genes on the permutation data sets.

assayData: Object of class "AssayData". The read count data.

phenoData: Object of class "AnnotatedDataFrame". Data for each samples.

featureData: Object of class "AnnotatedDataFrame". Data for each exons.

experimentData: Object of class "MIAXE". Experiment data.

annotation: Object of class "character". Not used.

protocolData: Object of class "AnnotatedDataFrame". Protocol information.

.\_\_classVersion\_\_: Object of class "Versions". Version information.



**Methods**

**counts** Get counts from a ReadCountSet object. See [counts](#).

**counts<-** Set counts to a ReadCountSet object. See [counts](#).

**Extends**

Class "[eSet](#)", directly.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**References**

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. BMC Bioinformatics, 14(Suppl 5):S16.

**See Also**

[newReadCountSet](#), [loadExonCountData](#), [exonID](#), [geneID](#), [counts-methods](#), [label](#), [subsetByGenes](#)

**Examples**

```
showClass("ReadCountSet")
```

---

runDESeq

*Run DESeq for differential expression analysis*

---

**Description**

This function provides a wrapper to run DESeq for differential expression analysis. It includes two steps, `DESeq::estimateSizeFactors` and `DESeq::estimateDispersions`.

**Usage**

```
runDESeq(geneCounts, label)
```

**Arguments**

**geneCounts** a matrix containing read counts for each gene, can be the output of [getGeneCount](#).  
**label** the sample classification labels.

**Value**

A CountDataSet object with size factors and dispersion parameters been estimated.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

## References

Anders, S. and Huber, W. (2010) Differential expression analysis for sequence count data, *Genome Biol*, 11, R106.

## See Also

[getGeneCount](#), [DENBTest](#), [DENBStat4GSEA](#)

## Examples

```
data(RCS_example, package="SeqGSEA")
geneCounts <- getGeneCount(RCS_example)
label <- label(RCS_example)
DEG <- runDESeq(geneCounts, label)
```

---

runSeqGSEA

*An all-in function that allows end users to apply SeqGSEA to their data with one step.*

---

## Description

This function provides typical SeqGSEA analysis pipelines for end users to apply the SeqGSEA method in the easiest fashion. It assumes the pipelines start with exon reads counts, even for the DE-only analysis. Users should specify their file locations and a few parameters before running this pipeline.

It allows DE-only analysis, which will skip the DS analysis portion, and it also allows users to try different weights in integrating DE and DS scores, which will save time in computing the DE and DS scores.

The function returns a list of SeqGSEA analysis results in the format of [GSEAResultTable](#), and generates a few plots and writes a few files, whose name prefix can be specified. The output files will either be in PDF format or TXT format, and generated by [plotGeneScore](#), [writeScores](#), [plotES](#), [plotSig](#), [plotSigGeneSet](#), and [writeSigGeneSet](#).

## Usage

```
runSeqGSEA(data.dir, case.pattern, ctrl.pattern, geneset.file, output.prefix, topGS=10,
  geneID.type=c("gene.symbol", "ensembl"), nCores=1, perm.times=1000, seed=NULL,
  minExonReadCount=5, integrationMethod=c("linear", "quadratic", "rank"),
  DEweight=c(0.5), DEonly=FALSE, minGSsize=5, maxGSsize=1000, GSEA.WeightedType=1)
```

## Arguments

<code>data.dir</code>	a character vector, the path to your count data directory.
<code>case.pattern</code>	a character vector, the unique pattern in the file names of case samples. E.g, if file names starting with "SC", the pattern writes "^SC".
<code>ctrl.pattern</code>	a character vector, the unique pattern in the file names of control samples.
<code>geneset.file</code>	a character vector, the path to your gene set file. The gene set file must be in GMT format. Please refer to the link follows for details. <a href="http://www.broadinstitute.org/cancer/software">http://www.broadinstitute.org/cancer/software</a>
<code>output.prefix</code>	a character vector, the path with prefix for output files.

topGS	an integer, this number of top ranked gene sets will be output with details; if geneset.file contains less than this number of gene sets, all gene sets' result details will be output. Default: 10.
geneID.type	the gene ID type in geneset.file. Currently only support "gene.symbol" and "ensembl". Default: gene.symbol.
nCores	an integer. The number of cores for running SeqGSEA. Default: 1
perm.times	an integer. The number of times for permutation, which will be used for normalizing DE and DS scores and for GSEA significance analysis. Recommended values are greater than 1000. Default: 1000.
seed	an integer or NULL, used for setting the seeds to generate random numbers. The same seed will guarantee the same analysis results given by SeqGSEA. Default: NULL.
minExonReadCount	an integer. An exon with total read count across all samples less than this number will be marked as untestable and be excluded in SeqGSEA analysis. Default: 5.
integrationMethod	one of the three integration methods for DE and DS score integration: linear, quadratic, or rank. Default: linear.
DEweight	a real number between 0 and 1 OR a vector of those. Each number is the DE weight in DE and DS integration. If using a vector of real numbers, SeqGSEA will run with each of them individually. Default: 0.5.
DEonly	logical, whether to run SeqGSEA only considering DE. Default: FALSE
minGSsize	an integer. The minimum gene set size: gene sets with genes less than this number will be skipped. Default: 5.
maxGSsize	an integer. The maximum gene set size: gene sets with genes greater than this number will be skipped. Default: 1000.
GSEA.WeightedType	the weight type of the main GSEA algorithm, can be 0 (unweighted = Kolmogorov-Smirnov), 1 (weighted), and 2 (over-weighted). Default: 1. It is recommended not to change it.

### Value

A list of SeqGSEA analysis results in the format of [GSEAResultTable](#), which allows users for meta-analysis.

### Author(s)

Xi Wang, xi.wang@mdc-berlin.de

### References

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

### See Also

[GSEAResultTable](#), [geneScore](#), [GSEnrichAnalyze](#)

**Examples**

```

### Initialization ###
# input file location and pattern
data.dir <- system.file("extdata", package="SeqGSEA", mustWork=TRUE)
case.pattern <- "^SC" # file name starting with "SC"
ctrl.pattern <- "^SN" # file name starting with "SN"
# gene set file and type
geneset.file <- system.file("extdata", "gs_symb.txt",
                           package="SeqGSEA", mustWork=TRUE)
geneID.type <- "ensembl"
# output file prefix
output.prefix <- "SeqGSEAexample"
# analysis parameters
nCores <- 1
perm.times <- 10
DEonly <- FALSE
DEweight <- c(0.2, 0.5, 0.8) # a vector for different weights
integrationMethod <- "linear"

### one step SeqGSEA running ###
# Caution: if running the following command line, it will generate many files in your working directory
## Not run:
runSeqGSEA(data.dir=data.dir, case.pattern=case.pattern, ctrl.pattern=ctrl.pattern,
           geneset.file=geneset.file, geneID.type=geneID.type, output.prefix=output.prefix,
           nCores=nCores, perm.times=perm.times, integrationMethod=integrationMethod,
           DEonly=DEonly, DEweight=DEweight)

## End(Not run)

```

---

scoreNormalization      *Normalization of DE/DS scores*

---

**Description**

Normalization of DE/DS scores or permutation DE/DS scores.

**Usage**

```
scoreNormalization(scores, norm.factor)
```

**Arguments**

`scores`                    a vector (a nX1 matrix) of a matrix of scores, rows corresponding to genes and columns corresponding to a study or permutation.

`norm.factor`                normalization factor, output of the function `normFactor`.

**Value**

A normalized vector or matrix depending on the input: with the same dimensions as the input.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

## References

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. *BMC Bioinformatics*, 14(Suppl 5):S16.

## See Also

[normFactor](#)

## Examples

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermute4GSEA(RCS_example, permuteMat)
## (not run)
DSscore.normFac <- normFactor(RCS_example@permute_NBstat_gene)
DSscore <- scoreNormalization(RCS_example@featureData_gene$NBstat, DSscore.normFac)
DSscore.perm <- scoreNormalization(RCS_example@permute_NBstat_gene, DSscore.normFac)
## End (not run)
```

---

SeqGeneSet-class	<i>Class "SeqGeneSet"</i>
------------------	---------------------------

---

## Description

SeqGeneSet class

## Objects from the Class

Objects can be created by calls of the function [newGeneSets](#).

## Slots

**name:** Object of class "character" the name of this gene set category  
**sourceFile:** Object of class "character" the source file of gene set category  
**geneList:** Object of class "character" the gene ID list indicating genes involved in this GSEA  
**GS:** Object of class "list" a list of gene indexes corresponding to geneList, each element in the list indicating which genes are in each gene set of this SeqGeneSet object  
**GSNames:** Object of class "character". Gene set names.  
**GSDescs:** Object of class "character". Gene set descriptions.  
**GSSize:** Object of class "numeric". Gene set sizes.  
**GSSizeMin:** Object of class "numeric". The minimum gene set size to be analyzed.  
**GSSizeMax:** Object of class "numeric". The maximum gene set size to be analyzed.  
**GS.Excluded:** Object of class "list". Gene sets excluded to be analyzed.  
**GSNames.Excluded:** Object of class "character". Gene set names excluded to be analyzed.  
**GSDescs.Excluded:** Object of class "character". Gene set descriptions excluded to be analyzed.

GSEA.ES: Object of class "numeric". Enrichment scores.

GSEA.ES.pos: Object of class "numeric". The positions where enrichment scores appear.

GSEA.ES.perm: Object of class "matrix". The enrichment scores of the permutation data sets.

GSEA.score.cumsum: Object of class "matrix". Running enrichment scores.

GSEA.normFlag: Object of class "logical". Logical indicating whether GSEA.ES has been normalized.

GSEA.pval: Object of class "numeric". P-values of each gene set.

GSEA.FWER: Object of class "numeric". Family-wise error rate of each gene set.

GSEA.FDR: Object of class "numeric". False discovery rate of each gene set.

sc.ES: Object of class "numeric". Enrichment scores in scGSEA.

sc.ES.perm: Object of class "matrix". The enrichment scores of the permutation data sets in scGSEA.

sc.normFlag: Object of class "logical". Logical indicating whether sc.ES has been normalized in scGSEA.

scGSEA: Object of class "logical". Whether or not used for scGSEA.

sc.pval: Object of class "numeric". P-values of each gene set in scGSEA.

sc.FWER: Object of class "numeric". Family-wise error rate of each gene set in scGSEA.

sc.FDR: Object of class "numeric". False discovery rate of each gene set in scGSEA.

version: Object of class "Versions". Version information.

## Methods

[ Get a sub-list of gene sets, and return a SeqGeneSet object.

**show** Show basic information of the SeqGeneSet object.

## Author(s)

Xi Wang, xi.wang@newcastle.edu.au

## References

Xi Wang and Murray J. Cairns (2013). Gene Set Enrichment Analysis of RNA-Seq Data: Integrating Differential Expression and Splicing. BMC Bioinformatics, 14(Suppl 5):S16.

## See Also

[newGeneSets](#), [size](#), [geneSetNames](#), [geneSetDescs](#), [geneSetSize](#)

## Examples

```
showClass("SeqGeneSet")
```

---

signifES	<i>Calculate significance of ESs</i>
----------	--------------------------------------

---

**Description**

This is an internal function to calculate significance of ESs of each gene set. For advanced users only.

**Usage**

```
signifES(gene.set)
```

**Arguments**

`gene.set` a GeneSet object after running [normES](#).

**Value**

A SeqGeneSet object with gene set enrichment significance metrics calculated.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [normES](#)

---

size	<i>Number of gene sets in a SeqGeneSet object</i>
------	---------------------------------------------------

---

**Description**

This function to get the number of gene sets in a SeqGeneSet object.

**Usage**

```
size(GS)
```

**Arguments**

`GS` an object of class SeqGeneSet.

**Details**

Gene sets with size less than `GSSizeMin` or more than `GSSizeMax` are not included.

**Value**

The number of gene sets in this SeqGeneSet object.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[SeqGeneSet-class](#), [loadGenesets](#)

**Examples**

```
data(GS_example, package="SeqGSEA")
size(GS_example)
```

---

subsetByGenes

*Get a new ReadCountSet with specified gene IDs.*

---

**Description**

Get a new ReadCountSet with specified gene IDs.

**Usage**

```
subsetByGenes(RCS, genes)
```

**Arguments**

RCS	a ReadCountSet object.
genes	a list of gene IDs.

**Value**

This function returns a new ReadCountSet object, with changes in slots assayData, featureData, featureData\_gene, and permute\_NBstat\_exon and permute\_NBstat\_gene if they have been calculated.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[newReadCountSet](#), [ReadCountSet](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
RCS_example
genes <- c("ENSG00000000938", "ENSG00000000005")
RCS_sub <- subsetByGenes(RCS_example, genes)
RCS_sub
```



---

topDEGenes	<i>Extract top differentially expressed genes.</i>
------------	----------------------------------------------------

---

### Description

This function is to extract top n differentially expressed genes, ranked by either DESeq p-values, DESeq adjusted p-values, permutation p-values, permutation adjusted p-values, or NB-statistics.

### Usage

```
topDEGenes(DEGres, n = 20,  
           sortBy = c("padj", "pval", "perm.pval", "perm.padj", "NBstat", "foldChange"))
```

### Arguments

DEGres	DE analysis results.
n	the number of top DE genes.
sortBy	indicating which method to rank genes.

### Details

If the sortBy method is not among the column names, the function will result in an error.

### Value

A table for top n DE genes with significance metrics.

### Author(s)

Xi Wang, xi.wang@newcastle.edu.au

### See Also

[topDSGenes](#), [topDSExons](#)

### Examples

```
data(RCS_example, package="SeqGSEA")  
geneCounts <- getGeneCount(RCS_example)  
label <- label(RCS_example)  
DEG <- runDESeq(geneCounts, label)  
permuteMat <- genpermuteMat(RCS_example, times=10)  
DEGres <- DENBTest(DEG)  
DEpermNBstat <- DENBStatPermut4GSEA(DEG, permuteMat)  
DEGres <- DEpermutePval(DEGres, DEpermNBstat)  
topDEGenes(DEGres, n = 10, sortBy = "NBstat")
```

---

topDSExons	<i>Extract top differentially spliced exons</i>
------------	-------------------------------------------------

---

### Description

This function is to extract top n differentially spliced exons, ranked by p-values or NB-stats.

### Usage

```
topDSExons(RCS, n = 20, sortBy = c("pvalue", "NBstat"))
```

### Arguments

RCS                    a ReadCountSet object after running [DSpermutePval](#).

n                      the number of top genes.

sortBy                indicating whether p-value or NBstat to be used for ranking genes.

### Value

A table for top n exons. Columns include: geneID, exonID, testable, NBstat, pvalue, padjust, and meanCounts.

### Author(s)

Xi Wang, xi.wang@newcastle.edu.au

### See Also

[topDSGenes](#), [DSpermutePval](#)

### Examples

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermutePval(RCS_example, permuteMat)
topDSExons(RCS_example, 10, "NB")
```

---

topDSGenes	<i>Extract top differentially spliced genes</i>
------------	-------------------------------------------------

---

**Description**

This function to extract top n differentially spliced genes, ranked by p-values or NBstats.

**Usage**

```
topDSGenes(RCS, n = 20, sortBy = c("pvalue", "NBstat"))
```

**Arguments**

RCS	a ReadCountSet object after running <a href="#">DSpermutePval</a> .
n	the number of top genes.
sortBy	indicating whether p-value or NBstat to be used for ranking genes.

**Value**

A table for top n genes. Columns include: geneID, NBstat, pvalue, and padjust.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[topDSExons](#), [DSpermutePval](#)

**Examples**

```
data(RCS_example, package="SeqGSEA")
permuteMat <- genpermuteMat(RCS_example, times=10)
RCS_example <- exonTestability(RCS_example)
RCS_example <- estiExonNBstat(RCS_example)
RCS_example <- estiGeneNBstat(RCS_example)
RCS_example <- DSpermutePval(RCS_example, permuteMat)
topDSGenes(RCS_example, 10, "NB")
```

---

topGeneSets	<i>Extract top significant gene sets</i>
-------------	------------------------------------------

---

**Description**

This function is to extract n top significant gene sets overrepresented in the samples studied, ranked by FDR, p-values, or FWER.

**Usage**

```
topGeneSets(gene.set, n = 20, sortBy = c("FDR", "pvalue", "FWER"), GSDesc = FALSE)
```

**Arguments**

gene.set	an object of class SeqGeneSet after GSEA runs.
n	the number of top gene sets.
sortBy	indicating which method to rank gene sets.
GSDesc	logical indicating whether or not to output gene set descriptions.

**Value**

A data frame for top n gene sets detected with respect to the ranking method specified. Information includes: GSName, GSSize, ES, ES.pos, pval, FDR, and FWER.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [GSEAResultTable](#)

**Examples**

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
topGeneSets(GS_example, n=5)
```

---

writeScores

*Write DE/DS scores and gene scores*

---

**Description**

This function is to write DE and DS scores, and optionally gene scores.

**Usage**

```
writeScores(DEscore, DSscore, geneScore=NULL, geneScoreAttr=NULL, file="")
```

**Arguments**

DEscore	normalized DE scores.
DSscore	normalized DS scores.
geneScore	gene scores integrated from DE and DS scores.
geneScoreAttr	the parameters for integrating DE and DS scores.
file	output file name, if not specified print to screen.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[DEscore](#), [geneScore](#)

**Examples**

```
data(DEscore, package="SeqGSEA")
data(DScore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DScore, method="linear", DEweight = 0.3)
writeScores(DEscore, DScore) # without gene scores
writeScores(DEscore, DScore, geneScore = gene.score,
            geneScoreAttr = "linear,0.3") # gene scores with attr.
```

---

writeSigGeneSet

*Write gene set supporting information*

---

**Description**

This function is to write the specified gene set (whose index is *i*) with significance information, including p-value and FDR, and gene scores for each gene in this set.

**Usage**

```
writeSigGeneSet(gene.set, i, gene.score, file = "")
```

**Arguments**

<code>gene.set</code>	an object of class <code>SeqGeneSet</code> with <a href="#">GSEnrichAnalyze</a> done.
<code>i</code>	the <i>i</i> -th gene set in the <code>SeqGeneSet</code> object. <a href="#">topGeneSets</a> is useful to find the most significantly overrepresented gene set.
<code>gene.score</code>	the vector of gene scores for running GSEA.
<code>file</code>	output file name, if not specified print to screen.

**Details**

See [plotSigGeneSet](#), which shows graphic information of the gene set specified.

**Author(s)**

Xi Wang, xi.wang@newcastle.edu.au

**See Also**

[GSEnrichAnalyze](#), [topGeneSets](#), [plotSigGeneSet](#)

**Examples**

```
data(DEscore, package="SeqGSEA")
data(DSscore, package="SeqGSEA")
gene.score <- geneScore(DEscore, DSscore, method="linear", DEweight = 0.3)
data(DEscore.perm, package="SeqGSEA")
data(DSscore.perm, package="SeqGSEA")
gene.score.perm <- genePermuteScore(DEscore.perm, DSscore.perm, method="linear", DEweight=0.3)
data(GS_example, package="SeqGSEA")
GS_example <- GSEnrichAnalyze(GS_example, gene.score, gene.score.perm)
topGeneSets(GS_example, n=5)
writeSigGeneSet(GS_example, 9, gene.score) # 9th gene set is the most significant one.
```

# Index

- \* **classes**
  - ReadCountSet-class, 40
  - SeqGeneSet-class, 45
- \* **datasets**
  - DEscore, 12
  - GS\_example, 29
  - RCS\_example, 40
- [, SeqGeneSet, numeric, ANY, ANY-method (SeqGeneSet-class), 45
- [, SeqGeneSet, numeric-method (SeqGeneSet-class), 45
  
- calES, 4, 5, 6, 34
- calES.perm, 4, 5, 6, 34
- convertEnsembl2Symbol, 5, 6, 7
- convertSymbol2Ensembl, 5, 7, 7
- counts, 41
- counts (counts-methods), 8
- counts, ReadCountSet-method (counts-methods), 8
- counts-methods, 8
- counts<- , ReadCountSet, matrix-method (counts-methods), 8
  
- DENBStat4GSEA, 4, 8, 10, 11, 42
- DENBStatPermut4GSEA, 4, 9, 9, 11, 13, 26
- DENBTest, 4, 8, 9, 10, 11, 42
- DEpermutePval, 4, 9, 10, 11
- DEscore, 12, 53
- DSpermute4GSEA, 4, 12, 14, 26
- DSpermutePval, 4, 13, 13, 14, 15, 50, 51
- DSresultExonTable, 4, 14, 15
- DSresultGeneTable, 4, 15, 15
- DSscore (DEscore), 12
  
- eSet, 41
- estiExonNBstat, 4, 13, 14, 16, 17
- estiGeneNBstat, 4, 13, 14, 16, 17
- exonID, 3, 18, 19, 41
- exonID<- (exonID), 18
- exonTestability, 4, 12, 13, 16, 18, 25
  
- geneID, 3, 18, 19, 41
- geneID<- (geneID), 19
  
- geneList, 20
- genePermuteScore, 4, 21, 22, 39
- geneScore, 4, 21, 22, 39, 43, 53
- geneSetDescs, 3, 23, 24, 46
- geneSetNames, 3, 23, 23, 24, 46
- geneSetSize, 3, 23, 24, 24, 46
- geneTestability, 4, 19, 25
- genpermuteMat, 5, 9, 10, 12–14, 26
- getGeneCount, 4, 26, 41, 42
- GS\_example, 29
- GSEAResultTable, 4, 27, 42, 43, 52
- GSEnrichAnalyze, 4–6, 27, 28, 34–38, 43, 47, 52, 53
  
- label, 3, 29, 41
- loadExonCountData, 3, 27, 30, 33, 41
- loadGenesets, 3, 20, 23, 24, 31, 33, 48
  
- newGeneSets, 3, 32, 32, 45, 46
- newReadCountSet, 3, 18, 19, 30, 33, 40, 41, 48
- normES, 4, 28, 34, 47
- normFactor, 4, 34, 44, 45
  
- plotES, 4, 35, 38, 42
- plotGeneScore, 4, 36, 42
- plotSig, 4, 37, 38, 42
- plotSigGeneSet, 4, 36, 37, 38, 42, 53
  
- rankCombine, 4, 39
- RCS\_example, 40
- ReadCountSet, 3, 48
- ReadCountSet (ReadCountSet-class), 40
- ReadCountSet-class, 40
- runDESeq, 4, 8–11, 27, 41
- runSeqGSEA, 42
  
- scoreNormalization, 4, 35, 44
- SeqGeneSet (SeqGeneSet-class), 45
- SeqGeneSet-class, 45
- SeqGSEA (SeqGSEA-package), 3
- SeqGSEA-package, 3
- show, SeqGeneSet-method (SeqGeneSet-class), 45
- signifES, 4, 28, 34, 47
- size, 3, 46, 47

subsetByGenes, [3](#), [25](#), [41](#), [48](#)

topDEGenes, [4](#), [49](#)

topDSExons, [4](#), [49](#), [50](#), [51](#)

topDSGenes, [4](#), [49](#), [50](#), [51](#)

topGeneSets, [4](#), [27](#), [38](#), [51](#), [53](#)

writeScores, [42](#), [52](#)

writeSigGeneSet, [5](#), [38](#), [42](#), [53](#)