

# Package ‘broadSeq’

February 20, 2025

**Title** broadSeq : for streamlined exploration of RNA-seq data

**Version** 1.1.0

**Description** This package helps user to do easily RNA-seq data analysis with multiple methods (usually which needs many different input formats). Here the user will provide the expression data as a SummarizedExperiment object and will get results from different methods. It will help user to quickly evaluate different methods.

**License** MIT + file LICENSE

**URL** <https://github.com/dasroy/broadSeq>

**BugReports** <https://github.com/dasroy/broadSeq/issues>

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**biocViews** GeneExpression, DifferentialExpression, RNASeq, Transcriptomics, Sequencing, Coverage, GeneSetEnrichment, GO

**Suggests** knitr, limma (>= 3.54.0), rmarkdown, stats (>= 4.2.2), samr

**Depends** dplyr, ggpubr, SummarizedExperiment

**Imports** BiocStyle, DELocal, EBSeq (>= 1.38.0), DESeq2 (>= 1.38.2), NOISeq, forcats (>= 1.0.0), genefilter, ggplot2, ggplotify, plyr, clusterProfiler (>= 4.8.2), pheatmap, sechm (>= 1.6.0), stringr, purrr (>= 0.3.5), edgeR (>= 3.40.1)

**VignetteBuilder** knitr

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

**git\_branch** devel

**git\_last\_commit** 78af4d8

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2025-02-20

**Author** Rishi Das Roy [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-3276-7279>>)

**Maintainer** Rishi Das Roy <rishi.dasroy@gmail.com>

## Contents

broadSeq-package . . . . .	2
combinedEnrichment . . . . .	3
genes_plot . . . . .	4
normalizeEdgerCPM . . . . .	5
plotHeatmapCluster . . . . .	6
plot_MDS . . . . .	7
prcompTidy . . . . .	8
round_df . . . . .	9
sampleAssay_plot . . . . .	10
transformDESeq2 . . . . .	10
use_DELocal . . . . .	11
use_deseq2 . . . . .	12
use_EBSeq . . . . .	13
use_edgeR_GLM . . . . .	14
use_limma_trend . . . . .	15
use_multDE . . . . .	16
use_NOIseq . . . . .	17
use_SAMseq . . . . .	18
volcanoPlot . . . . .	19
<b>Index</b>	<b>21</b>

---

broadSeq-package	<i>broadSeq : for streamlined exploration of RNA-seq data</i>
------------------	---

---

### Description

This package helps user to do easily RNA-seq data analysis with multiple methods (usually which needs many different input formats). Here the user will provide the expression data as a SummarizedExperiment object and will get results from different methods. It will help user to quickly evaluate different methods.

### Author(s)

**Maintainer:** Rishi Das Roy <rishi.dasroy@gmail.com> ([ORCID](#))

### See Also

Useful links:

- <https://github.com/dasroy/broadSeq>
- Report bugs at <https://github.com/dasroy/broadSeq/issues>

---

combinedEnrichment      *Provides GO gene set enrichment and over-representation analysis*

---

### Description

This wrapper function combines clusterProfiler::gseGO and clusterProfiler::enrichGO. The input type of these two methods are different; order ranked geneList and a vector of entrez gene id. Here combinedEnrichment function internally generates these two data types from user defined DEG\_table (differentially expressed genes).

### Usage

```
combinedEnrichment(
  DEG_table,
  geneCol = "ID",
  logCol = "logFoldChange",
  OrgDB = "org.Hs.eg.db",
  keyType,
  universe,
  ont = "BP",
  logfoldCut = 1,
  pvalueCutoff = 0.05,
  qvalueCutoff = 0.05
)
```

### Arguments

DEG_table	A data.frame atleast with two columns.
geneCol	The column name of DEG_table which provides gene ids and should be compatible with keytype parameter.
logCol	The column name of DEG_table which provides logfold(numeric) values to create a order ranked geneList for gseGO funtion.
OrgDB	OrgDb; passed to clusterProfiler functions
keyType	keytype of input gene(geneCol). One of the keytypes(OrgDB); passed to clusterProfiler functions
universe	background genes; passed to clusterProfiler::enrichGO.
ont	one of "BP", "MF", and "CC" subontologies, or "ALL" for all three.; passed to clusterProfiler functions
logfoldCut	to filter genes based on parameter logCol
pvalueCutoff	; passed to clusterProfiler functions
qvalueCutoff	; passed to clusterProfiler functions

### Value

a named list of three data.frames which are output of gseGO("gseResult") and enrichGO ("oraUP" and "oraDOWN").

---

genes_plot	<i>Expression of multiple genes/features from a single assay as boxplot (or added dotplot)</i>
------------	--

---

### Description

Expression of multiple genes/features from a single assay as boxplot (or added dotplot)

Boxplot of a single gene/feature from multiple assays

### Usage

```
genes_plot(se, features, assayName = "counts", facet.by = "feature", x, ...)
```

```
assay_plot(se, feature, assayNames = c("counts"), x, ...)
```

### Arguments

se	Object of <a href="#">SummarizedExperiment</a> class
features	a character vector of rownames or named list of character vectors where name is one of the colnames of rowData.
assayName	One of the values from <code>SummarizedExperiment::assayNames(se)</code> ; default is "counts" assay
facet.by	must be one of the column names of <code>rowData(se)</code> . default "feature" which is equivalent to rownames of <code>rowData</code>
x	a column name of <code>colData</code> which will be used in x-axis
...	other arguments to be passed to <code>ggpubr::ggboxplot</code>
feature	a character vector of rownames or named list of character vectors where name is one of the column of <code>rowData</code> .
assayNames	names from <code>SummarizedExperiment::assayNames(se)</code> ; default value is "counts"

### Value

ggplot object

return an object of class `ggarrange`, which is a `ggplot` or a list of `ggplot`.

### Examples

```
se <- readRDS(system.file("extdata", "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))
# The normalized values are added with the assay name "logCPM"
se <- broadSeq::normalizeEdgerCPM(se ,method = "none",cpm.log = TRUE )

broadSeq::genes_plot(se,
  features = list(mouse_gene_id = c("ENSMUSG00000022510" ,
    "ENSMUSG00000027985")),
```

```

        facet.by = "symbol", # column of rowData
        x = "stage", fill="stage")

broadSeq::genes_plot(se,
  features = list(symbol=c("Shh", "Edar") ),
  facet.by = "symbol", # column of rowData
  x = "stage", fill="stage")

broadSeq::assay_plot(se, feature = c("Shh"),
  assays = c("counts", "logCPM"),
  x = "stage", fill="stage", add="dotplot", palette = "npg")

```

---

normalizeEdgerCPM      *Use of edgeR package to normalize count data*

---

## Description

Use of edgeR package to normalize count data

## Usage

```
normalizeEdgerCPM(se, method = "TMM", cpm.log = TRUE, ...)
```

## Arguments

se	Object of <a href="#">SummarizedExperiment</a> class
method	value for <code>edgeR::normLibSizes</code> function. default "TMM"
cpm.log	value for <code>edgeR::cpm</code> function. default TRUE
...	passed to <code>normLibSizes</code> function

## Value

Object of [SummarizedExperiment](#) class where a new assay is added to the input object.

## Examples

```

se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

se <- broadSeq::normalizeEdgerCPM(se , method = "TMM", cpm.log = FALSE )
# The normalized values are added with the assay name "TMM"
SummarizedExperiment::assayNames(se)

```

---

plotHeatmapCluster      *Plot clustered heatmaps*

---

### Description

Plot clustered heatmaps from SummarizedExperiment with pheatmap and return object as ggplot

### Usage

```
plotHeatmapCluster(
  se,
  scaledAssay = "vst",
  ntop = 500L,
  features = NULL,
  show_geneAs = NULL,
  annotation_col = NA,
  annotation_row = NA,
  ...
)
```

### Arguments

se	Object of <a href="#">SummarizedExperiment</a> class
scaledAssay	an scaled assay name from <code>SummarizedExperiment::assayNames(se)</code>
ntop	number of most-variable genes to select. Igores if "features" is specified.
features	character vector features/genes to be used to measure distance between the samples
show_geneAs	a character vector of colnames of <code>rowData(se)</code>
annotation_col	a character vector of colnames of <code>colData(se)</code>
annotation_row	a list of character vector of colnames of <code>rowData(se)</code>
...	other arguments like color or shape whose values should be similar to <code>colData</code> columns names passed to <a href="#">pheatmap</a>

### Value

ggplot object

### Examples

```
se <- readRDS(system.file("extdata", "rat_vole_mouseSE_salmon.rds", package = "broadSeq"))
se <- broadSeq::normalizeEdgerCPM(se ,method = "none",cpm.log = TRUE )
broadSeq::plotHeatmapCluster(
  se,
  scaledAssay = "logCPM",
```

```

    annotation_col = c("species", "stage"),
    annotation_row = c("Class", "gene_biotype"),
    ntop = 30, show_geneAs = "symbol",
    cluster_cols = TRUE, cluster_rows = FALSE,
    show_rownames = TRUE, show_colnames = FALSE,
    main = "Top 30 variable gene vst"
  )

```

---

plot\_MDS

*Classical multidimensional scaling*


---

## Description

Classical multidimensional scaling is based on measuring the distance between the samples.

## Usage

```
plot_MDS(se, scaledAssay = "vst", ntop = 500L, features = NULL, ...)
```

## Arguments

se	Object of <a href="#">SummarizedExperiment</a> class
scaledAssay	an scaled assay name from <code>SummarizedExperiment::assayNames(se)</code>
ntop	number of most-variable genes to select. Igores if "features" is specified.
features	character vector features/genes to be used to measure distance between the samples
...	other arguments like color or shape whose values should be similar to colData columns names passed to <code>ggpubr::ggscatter</code>

## Value

ggplot object

## Examples

```

se <- readRDS(system.file("extdata", "rat_vole_mouseSE_salmon.rds", package = "broadSeq"))

se <- broadSeq::transformDESeq2(se, method = "vst" )

broadSeq::plot_MDS(se, scaledAssay = "vst", ntop=500,
  color = "species", shape = "stage",
  ellipse=TRUE, legend = "bottom")

```

prcompTidy

*Perform Principal Components Analysis***Description**

This function returns the results of `stats::prcomp` in a tidy list format. This is more flexible for further custom PCA , biplot and exploring gene(factor) loading of the PCA.

**Usage**

```
prcompTidy(se, scaledAssay = "vst", ntop = 500L, features = NULL, ...)

plotAnyPC(computedPCA, x = 1, y = 2, ...)

biplotAnyPC(computedPCA, x = 1, y = 2, genes = NULL, genesLabel = NULL, ...)

getFeatureLoadRanking(computedPCA, pcs = seq_len(5), topN = 10, keep)
```

**Arguments**

se	Object of <a href="#">SummarizedExperiment</a> class
scaledAssay	an scaled assay name from <code>SummarizedExperiment::assayNames(se)</code>
ntop	number of most-variable genes to select. Igored if "features" is specified.
features	character vector features/genes to be used for PCA
...	other arguments like color or shape whose values should be similar to <code>colData</code> columns names passed to <code>ggpubr::ggscatter</code>
computedPCA	a list of data.frame returned by <a href="#">prcompTidy</a>
x	PC number for x-axis default 1
y	PC number for y-axis default 2
genes	if genes is NULL then top max and min loaded genes of each PCs are plotted
genesLabel	one of <code>rowData</code> column names
pcs	The numbers of PCs
topN	Number of features per PC
keep	the column names of <code>rowData</code> to keep the corresponding information

**Details**

Reused code



**Value**

a list with four data.frame objects: pc\_scores, eigen\_values, loadings (eigen vectors) and the original data.

ggplot object

ggplot object

a data.frame

**Examples**

```
se <- readRDS(system.file("extdata", "rat_vole_mouseSE_salmon.rds", package = "broadSeq"))
```

```
se <- broadSeq::normalizeEdgerCPM(se, method = "none", cpm.log = TRUE )
computedPCA_logCPM <- broadSeq::prcompTidy(se, scaledAssay = "logCPM", ntop = 500)
```

```
plotAnyPC(computedPCA = computedPCA_logCPM, x = 1, y = 2, color = "species",
           shape = "stage", legend = "bottom")
```

```
plotAnyPC(computedPCA = computedPCA_logCPM, x = 2, y = 3, color = "species",
           shape = "stage", legend = "bottom")
```

```
computedPCA_logCPM$eigen_values %>%
  dplyr::filter(var_exp >= 0.5) %>% # Selecting PC explaining more than 1% variance
  ggbarplot(x="PC", y="var_exp", label = TRUE, label.pos = "out")
```

---

round\_df

*Applies round function only on numeric columns of a data.frame.*

---

**Description**

Applies round function only on numeric columns of a data.frame.

**Usage**

```
round_df(df, digits)
```

**Arguments**

df                    data.frame object

digits                passed to [round](#)

**Value**

data.frame object

**Examples**

```
data("iris")
iris %>% round_df(digits = 0) %>% head()
```

---

sampleAssay_plot	<i>Useful to visualize distribution of assay values for each sample. Plots 'boxplot' of any assay for each sample. Aesthetic can be added from colData.</i>
------------------	---

---

**Description**

Useful to visualize distribution of assay values for each sample. Plots 'boxplot' of any assay for each sample. Aesthetic can be added from colData.

**Usage**

```
sampleAssay_plot(se, assayName = "counts", ...)
```

**Arguments**

se	Object of <a href="#">SummarizedExperiment</a> class
assayName	One of the values from <code>SummarizedExperiment::assayNames(se)</code>
...	other arguments to be passed to <code>ggpubr::ggboxplot</code>

**Value**

ggplot object

**Examples**

```
se <- readRDS(system.file("extdata", "rat_vole_mouseSE_salmon.rds", package = "broadSeq"))

sampleAssay_plot(se, assayName = "counts",
  fill="stage", # stage is a column name of colData(se)
  yscale="log2")

se <- broadSeq::normalizeEdgerCPM(se ,method = "none",cpm.log = TRUE )

sampleAssay_plot(se, assayName = "logCPM", fill="stage")
```

---

transformDESeq2	<i>Transform SummarizedExperiment with DESeq2 package</i>
-----------------	---

---

**Description**

To use `SummarizedExperiment` with `DESeq2`, this function makes sure that 'counts' assay should be the first in assays list and the mode is integer.

**Usage**

```
transformDESeq2(se, method = "vst", ...)
```

**Arguments**

se	Object of <code>SummarizedExperiment</code> class
method	"vst", "normTransform" or "rlog" to choose either of <code>DESeq2::varianceStabilizingTransformation</code> , <code>DESeq2::normTransform</code> and <code>DESeq2::rlog</code> function. default is "vst"
...	arguments passed to <code>varianceStabilizingTransformation</code> , <code>normTransform</code> and <code>rlog</code>

**Value**

Object of `SummarizedExperiment` class where a new assay is added to the input object.

**Examples**

```
se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

se <- broadSeq::transformDESeq2(se,method = "vst" )
# The transformed values are added with the assay name "vst"
SummarizedExperiment::assayNames(se)
```

---

use\_DELocal

*To use SummarizedExperiment with package DELocal*


---

**Description**

A wrapper function of `DELocal` where input is an object of `SummarizedExperiment`

**Usage**

```
use_DELocal(se, colData_id, control, treatment, rank = FALSE, ...)
```

**Arguments**

se	Object of <code>SummarizedExperiment</code> class
colData_id	One of the columns of <code>colData(se)</code> . It should be factors of more than one value.
control	Base level and one of the factor values of <code>colData(se)[[colData_id]]</code>
treatment	one of the factor values of <code>colData(se)[[colData_id]]</code>
rank	Logical value default FALSE. If true the result will have an additional column named "rank" and the results are ranked on "relative.logFC"
...	other arguments to be passed to main function <code>DELocal::DELocal</code> .

**Value**

a data.frame from `DELocal`

**Examples**

```
se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"]

result <-
  use_DELocal(se = se,
    colData_id = "stage", control = "Bud", treatment = "Cap",
    rank = TRUE)
```

---

 use\_deseq2

*To use SummarizedExperiment with package DESeq2*


---

**Description**

A wrapper function of DESeq2 where input is an object of [SummarizedExperiment](#)

**Usage**

```
use_deseq2(se, colData_id, control, treatment, rank = FALSE, ...)
```

**Arguments**

se	Object of <a href="#">SummarizedExperiment</a> class
colData_id	One of the columns of colData(se). It should be factors of more than one value.
control	Base level and one of the factor values of colData(se)[[colData_id]]
treatment	one of the factor values of colData(se)[[colData_id]]
rank	Logical value default FALSE. If true the result will have an additional column named "rank" and the results are ranked on "padj"
...	other arguments to be passed to main function DESeq2:: <a href="#">results</a> .

**Value**

a data.frame converted from DESeq2::[DESeqResults](#)

**Examples**

```
se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"]
```

```

result <-
  use_deseq2(se = se,
            colData_id = "stage", control = "Bud", treatment = "Cap",
            rank = TRUE)

```

---

use\_EBSeq

*To use SummarizedExperiment with package EBSeq*


---

## Description

A wrapper function of EBSeq where input is an object of [SummarizedExperiment](#)

## Usage

```
use_EBSeq(se, colData_id, control, treatment, rank = FALSE, ...)
```

## Arguments

se	Object of <a href="#">SummarizedExperiment</a> class
colData_id	One of the columns of colData(se). It should be factors of more than one value.
control	Base level and one of the factor values of colData(se)[[colData_id]]
treatment	one of the factor values of colData(se)[[colData_id]]
rank	Logical value default FALSE. If true the result will have an additional column named "rank" and the results are ranked on "PPDE"
...	other arguments to be passed to main function EBSeq::GetDEResults.

## Value

a data.frame object converted from the output of EBSeq::GetDEResults.

## Examples

```

se <- readRDS(system.file("extdata",
                        "rat_vole_mouseSE_salmon.rds",
                        package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2, colData(se)$species == "Mouse"]

result <-
  use_EBSeq(se = se,
            colData_id = "stage", control = "Bud", treatment = "Cap",
            rank = TRUE)

```

---

 use\_edgeR\_GLM

*To use SummarizedExperiment with package edgeR*


---

### Description

A wrapper function of DESeq2 where input is an object of [SummarizedExperiment](#)

### Usage

```
use_edgeR_GLM(se, colData_id, control, treatment, rank = FALSE, ...)
```

```
use_edgeR_exact(se, colData_id, control, treatment, rank = FALSE, ...)
```

```
use_edgeR(
  se,
  colData_id,
  control,
  treatment,
  rank = FALSE,
  edgeR.n = Inf,
  edgeR.adjust.method = "BH",
  edgeR.sort.by = "PValue",
  option = "GLM",
  ...
)
```

### Arguments

se	Object of <a href="#">SummarizedExperiment</a> class
colData_id	One of the columns of colData(se). It should be factors of more than one value.
control	Base level and one of the factor values of colData(se)[[colData_id]]
treatment	one of the factor values of colData(se)[[colData_id]]
rank	Logical value default FALSE. If true the result will have an additional column named "rank"
...	other arguments to be passed to <a href="#">edgeR::glmLRT</a> or <a href="#">edgeR::exactTest</a>
edgeR.n	argument for <a href="#">edgeR::topTags</a>
edgeR.adjust.method	argument for <a href="#">edgeR::topTags</a>
edgeR.sort.by	argument for <a href="#">edgeR::topTags</a>
option	"GLM" or "exact" to indicate to use either <a href="#">edgeR::glmLRT</a> or <a href="#">edgeR::exactTest</a>

### Value

a data.frame of output from [edgeR::topTags](#)

**Examples**

```

se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"]

result <-
  use_edgeR(se = se,
    colData_id = "stage", control = "Bud", treatment = "Cap",
    rank = TRUE)

```

---

use_limma_trend	<i>To use SummarizedExperiment with package limma</i>
-----------------	---

---

**Description**

A wrapper function of limma where input is an object of [SummarizedExperiment](#)

**Usage**

```
use_limma_trend(se, colData_id, control, treatment, rank = FALSE, ...)
```

```
use_limma_voom(se, colData_id, control, treatment, rank = FALSE, ...)
```

```

use_limma(
  se,
  colData_id,
  control,
  treatment,
  rank = FALSE,
  useVoom = TRUE,
  showPlot = FALSE,
  limma.adjust = "BH",
  limma.sort.by = "p",
  limma.number = Inf,
  ...
)

```

**Arguments**

se	Object of <a href="#">SummarizedExperiment</a> class
colData_id	One of the columns of colData(se). It should be factors of more than one value.
control	Base level and one of the factor values of colData(se)[[colData_id]]
treatment	one of the factor values of colData(se)[[colData_id]]

rank	Logical value default FALSE. If true the result will have an additional column named "rank"
...	other arguments to be passed to main function edgeR::calcNormFactors .
useVoom	whether to use limma::voom or edgeR::cpm
showPlot	whether to use limma::plotSA ; default FALSE
limma.adjust	argument for limma::topTable
limma.sort.by	argument for limma::topTable
limma.number	argument for limma::topTable

**Value**

a data.frame of output from limma::topTable

**Examples**

```
se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"]
result <-
  use_limma(se = se,
    colData_id = "stage", control = "Bud", treatment = "Cap",
    rank = TRUE)
```

---

use\_multDE

*To identify differentially expressed genes by multiple methods*

---

**Description**

To identify differentially expressed genes by multiple methods

**Usage**

```
use_multDE(
  deFun_list,
  return.df = FALSE,
  se,
  colData_id,
  control,
  treatment,
  ...
)
```



**Arguments**

deFun_list	a list of function which can perform differential expression analysis
return.df	whether to return all results aggregated form of data.frame or a list of results. Default is FALSE
se	Object of <a href="#">SummarizedExperiment</a> class
colData_id	One of the columns of colData(se). It should be factors of more than one value.
control	Base level and one of the factor values of colData(se)[[colData_id]]
treatment	one of the factor values of colData(se)[[colData_id]]
...	other arguments to be passed to functions listed in deFun_list

**Value**

a list or data.frame

**Examples**

```
se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"]

# First define a named list of functions
funs <- list(limma_trend = use_limma_trend, limma_voom = use_limma_voom,
  edgeR_exact = use_edgeR_exact, edgeR_glm = use_edgeR_GLM,
  deseq2 = use_deseq2,
  DELocal = use_DELocal, noiseq = use_NOIseq,
  EBSeq = use_EBSeq)

multi_result <- broadSeq::use_multDE(
  se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"],
  deFun_list = funs, return.df = TRUE,
  colData_id = "stage", control = "Bud", treatment = "Cap",
  rank = TRUE)
```

---

 use\_NOIseq

*Differential expression method for NOISeq*


---

**Description**

This is a wrapper function of NOISeq::[noiseqbio](#) whose input class is 'eSet' and output class is Output which are not widely used. We can use as(se, "ExpressionSet") to get an eSet easily but then it will be hard to refer the treatment and control. The order of factors influence the log fold change sign. To keep it comparable to other methods the NOISeq::readData() is used internally.

**Usage**

```
use_NOIseq(se, colData_id, control, treatment, rank = FALSE, ...)
```

**Arguments**

<code>se</code>	Object of <code>SummarizedExperiment</code> class
<code>colData_id</code>	One of the columns of <code>colData(se)</code> . It should be factors of more than one value.
<code>control</code>	Base level and one of the factor values of <code>colData(se)[[colData_id]]</code>
<code>treatment</code>	one of the factor values of <code>colData(se)[[colData_id]]</code>
<code>rank</code>	Logical value default <code>FALSE</code> . If true the result will have an additional column named "rank" which is ordered by 'prob' values returned by function <code>NOISeq::noiseqbio</code> .
<code>...</code>	other arguments to be passed to main function <code>NOISeq::noiseqbio</code> . The 'input' and 'factor' argument should not be used.

**Value**

A data.frame object from the results of `NOISeq::noiseqbio()`. For details check the documentation of 'NOISeq'

**Examples**

```
se <- readRDS(system.file("extdata",
  "rat_vole_mouseSE_salmon.rds",
  package = "broadSeq"))

# To reduce runtime
se <- se[rowData(se)$chromosome_name == 2,colData(se)$species == "Mouse"]

result_Noiseq <-
  use_NOIseq(se = se,
    colData_id = "stage", control = "Bud", treatment = "Cap",
    rank = TRUE,
    r = 10) # r is an argument of NOISeq::noiseqbio
```

---

use\_SAMseq

*To use SummarizedExperiment with package samr*

---

**Description**

To use SummarizedExperiment with package samr

**Usage**

```
use_SAMseq(se, colData_id, control, treatment, rank = FALSE, ...)
```

**Arguments**

se	Object of <code>SummarizedExperiment</code> class
colData_id	One of the columns of <code>colData(se)</code> . It should be factors of more than one value.
control	Base level and one of the factor values of <code>colData(se)[[colData_id]]</code>
treatment	one of the factor values of <code>colData(se)[[colData_id]]</code>
rank	Logical value default FALSE. If true the result will have an
...	other arguments to be passed to <code>samr::SAMseq</code>

**Value**

a data.frame object as a result

---

volcanoPlot	<i>Volcano plot with formatted x and y axis label.</i>
-------------	--

---

**Description**

Volcano plot with formatted x and y axis label.

**Usage**

```
volcanoPlot(
  df,
  pValName,
  lFCName,
  sigThreshold = 0.05,
  logFCThreshold = 1,
  labelName = NULL,
  selectedLabel = NULL,
  palette = "nejm"
)
```

**Arguments**

df	a data.frame object
pValName	column name of df which provides p-values
lFCName	column name of df which provides log fold change values
sigThreshold	Threshold for p-values
logFCThreshold	Threshold for log fold change values
labelName	column name of df to label the dots
selectedLabel	which dots to highlight
palette	one of "npg", "aaas", "lancet", "jco", "ucscgb", "uchicago", "simpsons" and "nejm" or similar to <code>viridis::cividis(3)</code>

**Value**

ggplot object

# Index

assay\_plot (genes\_plot), 4

biplotAnyPC (prcompTidy), 8

broadSeq (broadSeq-package), 2

broadSeq-package, 2

combinedEnrichment, 3

cpm, 5

DELocal, 11

DESeqResults, 12

exactTest, 14

genes\_plot, 4

GetDEResults, 13

getFeatureLoadRanking (prcompTidy), 8

ggboxplot, 4, 10

ggscatter, 7, 8

glmLRT, 14

noiseqbio, 17, 18

normalizeEdgerCPM, 5

normLibSizes, 5

normTransform, 11

pheatmap, 6

plot\_MDS, 7

plotAnyPC (prcompTidy), 8

plotHeatmapCluster, 6

prcomp, 8

prcompTidy, 8, 8

results, 12

rlog, 11

round, 9

round\_df, 9

sampleAssay\_plot, 10

SummarizedExperiment, 4–8, 10–15, 17–19

topTags, 14

transformDESeq2, 10

use\_DELocal, 11

use\_deseq2, 12

use\_EBSeq, 13

use\_edgeR (use\_edgeR\_GLM), 14

use\_edgeR\_exact (use\_edgeR\_GLM), 14

use\_edgeR\_GLM, 14

use\_limma (use\_limma\_trend), 15

use\_limma\_trend, 15

use\_limma\_voom (use\_limma\_trend), 15

use\_multDE, 16

use\_NOIseq, 17

use\_SAMseq, 18

varianceStabilizingTransformation, 11

volcanoPlot, 19