Package 'pcaExplorer'

October 16, 2018

Type Package

Title Interactive Visualization of RNA-seq Data Using a Principal Components Approach

Version 2.6.0

Date 2018-04-04

Maintainer Federico Marini <marinif@uni-mainz.de>

Description This package provides functionality for interactive visualization of RNA-seq datasets based on Principal Components Analysis. The methods provided allow for quick information extraction and effective data exploration. A Shiny application encapsulates the whole analysis.

License MIT + file LICENSE

LazyData TRUE

Imports DESeq2, SummarizedExperiment, GenomicRanges, IRanges, S4Vectors, genefilter, ggplot2 (>= 2.0.0), d3heatmap, scales, NMF, plyr, topGO, limma, GOstats, GO.db, AnnotationDbi, shiny (>= 0.12.0), shinydashboard, shinyBS, ggrepel, DT, shinyAce, threejs, biomaRt, pheatmap, knitr, rmarkdown, tidyr, grDevices, methods

Suggests testthat, BiocStyle, airway, org.Hs.eg.db

URL https://github.com/federicomarini/pcaExplorer

BugReports https://github.com/federicomarini/pcaExplorer/issues

biocViews Visualization, RNASeq, DimensionReduction, PrincipalComponent, QualityControl, GUI, ReportWriting

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Author Federico Marini [aut, cre]

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

git_branch RELEASE_3_7

git_last_commit 21cd327

git_last_commit_date 2018-04-30

Date/Publication 2018-10-15

R topics documented:

`correlatePCs` *Principal components (cor)relation with experimental covariates*

Description

Computes the significance of (cor)relations between PCA scores and the sample experimental covariates, using Kruskal-Wallis test for categorial variables and the `cor.test` based on Spearman's correlation for continuous variables

Usage

```
correlatePCs(pcaobj, coldata, pcs = 1:4)
```

Arguments

`pcaobj`	A `prcomp` object
`coldata`	A `data.frame` object containing the experimental covariates
`pcs`	A numeric vector, containing the corresponding PC number

Value

A `data.frame` object with computed p values for each covariate and for each principal component

Examples

```
library(DESeq2)
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- rlogTransformation(dds)
pcaobj <- prcomp(t(assay(rlt)))
correlatePCs(pcaobj,colData(dds))
```

---

distro_expr *Plot distribution of expression values*

Description

Plot distribution of expression values

Usage

```
distro_expr(rld, plot_type = "density")
```

Arguments

rld	A DESeqTransform object.
plot_type	Character, choose one of boxplot, violin or density. Defaults to density

Value

A plot with the distribution of the expression values

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- DESeq2::rlogTransformation(dds)
distro_expr(rlt)
```

---

geneprofiler *Extract and plot the expression profile of genes*

Description

Extract and plot the expression profile of genes

Usage

```
geneprofiler(se, genelist = NULL, intgroup = "condition", plotZ = FALSE)
```

Arguments

se	A DESeqDataSet object, or a DESeqTransform object.
genelist	An array of characters, including the names of the genes of interest of which the profile is to be plotted
intgroup	A factor, needs to be in the colnames of colData(se)
plotZ	Logical, whether to plot the scaled expression values. Defaults to FALSE

Value

A plot of the expression profile for the genes

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- DESeq2::rlogTransformation(dds)
geneprofiler(rlt,paste0("gene",sample(1:1000,20)))
geneprofiler(rlt,paste0("gene",sample(1:1000,20)),plotZ=TRUE)
```

---

genespca *Principal components analysis on the genes*

Description

Computes and plots the principal components of the genes, eventually displaying the samples as in a typical biplot visualization.

Usage

```
genespca(x, ntop, choices = c(1, 2), arrowColors = "steelblue",
  groupNames = "group", biplot = TRUE, scale = 1, pc.biplot = TRUE,
  obs.scale = 1 - scale, var.scale = scale, groups = NULL,
  ellipse = FALSE, ellipse.prob = 0.68, labels = NULL, labels.size = 3,
  alpha = 1, var.axes = TRUE, circle = FALSE, circle.prob = 0.69,
  varname.size = 4, varname.adjust = 1.5, varname.abbrev = FALSE,
  returnData = FALSE, coordEqual = FALSE, scaleArrow = 1,
  useRownamesAsLabels = TRUE, point_size = 2, annotation = NULL)
```

Arguments

x	A DESeqTransform object, with data in assay(x), produced for example by either rlog or varianceStabilizingTransformation
ntop	Number of top genes to use for principal components, selected by highest row variance
choices	Vector of two numeric values, to select on which principal components to plot
arrowColors	Vector of character, either as long as the number of the samples, or one single value
groupNames	Factor containing the groupings for the input data. Is efficiently chosen as the (interaction of more) factors in the colData for the object provided

biplot	Logical, whether to additionally draw the samples labels as in a biplot representation
scale	Covariance biplot (scale = 1), form biplot (scale = 0). When scale = 1, the inner product between the variables approximates the covariance and the distance between the points approximates the Mahalanobis distance.
pc.biplot	Logical, for compatibility with biplot.princomp()
obs.scale	Scale factor to apply to observations
var.scale	Scale factor to apply to variables
groups	Optional factor variable indicating the groups that the observations belong to. If provided the points will be colored according to groups
ellipse	Logical, draw a normal data ellipse for each group
ellipse.prob	Size of the ellipse in Normal probability
labels	optional Vector of labels for the observations
labels.size	Size of the text used for the labels
alpha	Alpha transparency value for the points (0 = transparent, 1 = opaque)
var.axes	Logical, draw arrows for the variables?
circle	Logical, draw a correlation circle? (only applies when prcomp was called with scale = TRUE and when var.scale = 1)
circle.prob	Size of the correlation circle in Normal probability
varname.size	Size of the text for variable names
varname.adjust	Adjustment factor the placement of the variable names, >= 1 means farther from the arrow
varname.abbrev	Logical, whether or not to abbreviate the variable names
returnData	Logical, if TRUE returns a data.frame for further use, containing the selected principal components for custom plotting
coordEqual	Logical, default FALSE, for allowing brushing. If TRUE, plot using equal scale cartesian coordinates
scaleArrow	Multiplicative factor, usually >=1, only for visualization purposes, to allow for distinguishing where the variables are plotted
useRownamesAsLabels	Logical, if TRUE uses the row names as labels for plotting
point_size	Size of the points to be plotted for the observations (genes)
annotation	A `data.frame` object, with row.names as gene identifiers (e.g. ENSEMBL ids) and a column, `gene_name`, containing e.g. HGNC-based gene symbols

Details

The implementation of this function is based on the beautiful `ggbiplot` package developed by Vince Vu, available at https://github.com/vqv/ggbiplot. The adaptation and additional parameters are tailored to display typical genomics data such as the transformed counts of RNA-seq experiments

Value

An object created by `ggplot`, which can be assigned and further customized.

Examples

```
library(DESeq2)
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- rlogTransformation(dds)
groups <- colData(dds)$condition
groups <- factor(groups,levels=unique(groups))
cols <- scales::hue_pal()(2)[groups]
genespca(rlt,ntop=100,arrowColors=cols,groupNames=groups)

groups_multi <- interaction(as.data.frame(colData(rlt)[,c("condition","tissue")]))
groups_multi <- factor(groups_multi,levels=unique(groups_multi))
cols_multi <- scales::hue_pal()(length(levels(groups_multi)))[factor(groups_multi)]
genespca(rlt,ntop=100,arrowColors=cols_multi,groupNames=groups_multi)
```

---

get_annotation *Get an annotation data frame from biomaRt*

Description

Get an annotation data frame from biomaRt

Usage

```
get_annotation(dds, biomart_dataset, idtype)
```

Arguments

dds	A DESeqDataSet object
biomart_dataset	A biomaRt dataset to use. To see the list, type `mart = useMart('ensembl')`, followed by `listDatasets(mart)`.
idtype	Character, the ID type of the genes as in the row names of dds, to be used for the call to getBM

Value

A data frame for ready use in pcaExplorer, retrieved from biomaRt.

Examples

```
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
                                             colData = colData(airway),
                                             design=~dex+cell)
## Not run:
get_annotation(dds_airway,"hsapiens_gene_ensembl","ensembl_gene_id")

## End(Not run)
```

get_annotation_orgdb *Get an annotation data frame from org db packages*

Description

Get an annotation data frame from org db packages

Usage

```
get_annotation_orgdb(dds, orgdb_species, idtype)
```

Arguments

dds	A DESeqDataSet object
orgdb_species	Character string, named as the org.XX.eg.db package which should be available in Bioconductor
idtype	Character, the ID type of the genes as in the row names of dds, to be used for the call to mapIds

Value

A data frame for ready use in pcaExplorer, retrieved from the org db packages

Examples

```
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
                                             colData = colData(airway),
                                             design=~dex+cell)
## Not run:
get_annotation_orgdb(dds_airway,"org.Hs.eg.db","ENSEMBL")

## End(Not run)
```

hi_loadings *Extract genes with highest loadings*

Description

Extract genes with highest loadings

Usage

```
hi_loadings(pcaobj, whichpc = 1, topN = 10, exprTable = NULL,
  annotation = NULL, title = "Top/bottom loadings - ")
```

Arguments

pcaobj	A prcomp object
whichpc	An integer number, corresponding to the principal component of interest
topN	Integer, number of genes with top and bottom loadings
exprTable	A matrix object, e.g. the counts of a DESeqDataSet. If not NULL, returns the counts matrix for the selected genes
annotation	A data.frame object, with row.names as gene identifiers (e.g. ENSEMBL ids) and a column, gene_name, containing e.g. HGNC-based gene symbols
title	The title of the plot

Value

A base plot object, or a matrix, if exprTable is not null

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD = 3,betaSD_tissue = 1)
rlt <- DESeq2::rlogTransformation(dds)
pcaobj <- prcomp(t(SummarizedExperiment::assay(rlt)))
hi_loadings(pcaobj,topN = 20)
hi_loadings(pcaobj,topN = 10,exprTable=dds)
hi_loadings(pcaobj,topN = 10,exprTable=counts(dds))
```

---

limmaquickpca2go *Functional interpretation of the principal components, based on simple overrepresentation analysis*

Description

Extracts the genes with the highest loadings for each principal component, and performs functional enrichment analysis on them using the simple and quick routine provided by the limma package

Usage

```
limmaquickpca2go(se, pca_ngenes = 10000, inputType = "ENSEMBL",
  organism = "Mm", loadings_ngenes = 500, background_genes = NULL,
  scale = FALSE, ...)
```

Arguments

se	A DESeqTransform object, with data in assay(se), produced for example by either rlog or varianceStabilizingTransformation
pca_ngenes	Number of genes to use for the PCA
inputType	Input format type of the gene identifiers. Deafults to ENSEMBL, that then will be converted to ENTREZ ids. Can assume values such as ENTREZID,GENENAME or SYMBOL, like it is normally used with the select function of AnnotationDbi
organism	Character abbreviation for the species, using org.XX.eg.db for annotation

`loadings_ngenes`
: Number of genes to extract the loadings (in each direction)

`background_genes`
: Which genes to consider as background.

`scale`
: Logical, defaults to FALSE, scale values for the PCA

`...`
: Further parameters to be passed to the topGO routine

Value

A nested list object containing for each principal component the terms enriched in each direction. This object is to be thought in combination with the displaying feature of the main `pcaExplorer` function

Examples

```
library(airway)
library(DESeq2)
library(limma)
data(airway)
airway
dds_airway <- DESeqDataSet(airway, design= ~ cell + dex)
## Not run:
rld_airway <- rlogTransformation(dds_airway)
goquick_airway <- limmaquickpca2go(rld_airway,
                                   pca_ngenes = 10000,
                                   inputType = "ENSEMBL",
                                   organism = "Hs")

## End(Not run)
```

---

makeExampleDESeqDataSet_multifac

Make a simulated DESeqDataSet for two or more experimental factors

---

Description

Constructs a simulated dataset of Negative Binomial data from different conditions. The fold changes between the conditions can be adjusted with the `betaSD_condition` and the `betaSD_tissue` arguments.

Usage

```
makeExampleDESeqDataSet_multifac(n = 1000, m = 12, betaSD_condition = 1,
  betaSD_tissue = 3, interceptMean = 4, interceptSD = 2,
  dispMeanRel = function(x) 4/x + 0.1, sizeFactors = rep(1, m))
```

Arguments

n	number of rows (genes)
m	number of columns (samples)
betaSD_condition	the standard deviation for condition betas, i.e. beta ~ N(0,betaSD)
betaSD_tissue	the standard deviation for tissue betas, i.e. beta ~ N(0,betaSD)
interceptMean	the mean of the intercept betas (log2 scale)
interceptSD	the standard deviation of the intercept betas (log2 scale)
dispMeanRel	a function specifying the relationship of the dispersions on 2^trueIntercept
sizeFactors	multiplicative factors for each sample

Details

This function is designed and inspired following the proposal of makeExampleDESeqDataSet from the DESeq2 package. Credits are given to Mike Love for the nice initial implementation

Value

a DESeqDataSet with true dispersion, intercept for two factors (condition and tissue) and beta values in the metadata columns. Note that the true betas are provided on the log2 scale.

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
dds
dds2 <- makeExampleDESeqDataSet_multifac(betaSD_condition = 1,betaSD_tissue = 4)
dds2
```

---

pair_corr *Pairwise scatter and correlation plot of counts*

Description

Pairwise scatter and correlation plot of counts

Usage

```
pair_corr(df, method = "pearson")
```

Arguments

df	A data frame, containing the (raw/normalized/transformed) counts
method	Character string, one of pearson (default), kendall, or spearman as in cor

Value

A plot with pairwise scatter plots and correlation coefficients

Examples

```
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
                                             colData = colData(airway),
                                             design=~dex+cell)
pair_corr(counts(dds_airway)[1:100,]) # use just a subset for the example
```

---

pca2go *Functional interpretation of the principal components*

Description

Extracts the genes with the highest loadings for each principal component, and performs functional enrichment analysis on them using routines and algorithms from the topGO package

Usage

```
pca2go(se, pca_ngenes = 10000, annotation = NULL,
  inputType = "geneSymbol", organism = "Mm", ensToGeneSymbol = FALSE,
  loadings_ngenes = 500, background_genes = NULL, scale = FALSE, ...)
```

Arguments

se	A DESeqTransform object, with data in assay(se), produced for example by either rlog or varianceStabilizingTransformation
pca_ngenes	Number of genes to use for the PCA
annotation	A data.frame object, with row.names as gene identifiers (e.g. ENSEMBL ids) and a column, gene_name, containing e.g. HGNC-based gene symbols
inputType	Input format type of the gene identifiers. Will be used by the routines of topGO
organism	Character abbreviation for the species, using org.XX.eg.db for annotation
ensToGeneSymbol	Logical, whether to expect ENSEMBL gene identifiers, to convert to gene symbols with the annotation provided
loadings_ngenes	Number of genes to extract the loadings (in each direction)
background_genes	Which genes to consider as background.
scale	Logical, defaults to FALSE, scale values for the PCA
...	Further parameters to be passed to the topGO routine

Value

A nested list object containing for each principal component the terms enriched in each direction. This object is to be thought in combination with the displaying feature of the main pcaExplorer function

Examples

```
library(airway)
library(DESeq2)
data(airway)
airway
dds_airway <- DESeqDataSet(airway, design= ~ cell + dex)
## Not run:
rld_airway <- rlogTransformation(dds_airway)

# constructing the annotation object
anno_df <- data.frame(gene_id = rownames(dds_airway),
                      stringsAsFactors=FALSE)
library("AnnotationDbi")
library("org.Hs.eg.db")
anno_df$gene_name <- mapIds(org.Hs.eg.db,
                            keys=anno_df$gene_id,
                            column="SYMBOL",
                            keytype="ENSEMBL",
                            multiVals="first")
rownames(anno_df) <- anno_df$gene_id
bg_ids <- rownames(dds_airway)[rowSums(counts(dds_airway)) > 0]
library(topGO)
pca2go_airway <- pca2go(rld_airway,
                        annotation = anno_df,
                        organism = "Hs",
                        ensToGeneSymbol = TRUE,
                        background_genes = bg_ids)

## End(Not run)
```

`pcaExplorer` *pcaExplorer: analyzing time-lapse microscopy imaging, from detection to tracking*

Description

pcaExplorer provides functionality for interactive visualization of RNA-seq datasets based on Principal Components Analysis. The methods provided allow for quick information extraction and effective data exploration. A Shiny application encapsulates the whole analysis.

Launch a Shiny App for interactive exploration of a dataset from the perspective of Principal Components Analysis

Usage

```
pcaExplorer(dds = NULL, rlt = NULL, countmatrix = NULL, coldata = NULL,
  pca2go = NULL, annotation = NULL)
```

Arguments

dds	A DESeqDataSet object. If not provided, then a countmatrix and a coldata need to be provided. If none of the above is provided, it is possible to upload the data during the execution of the Shiny App
rlt	A DESeqTransform object. Can be computed from the dds object if left NULL. If none is provided, then a countmatrix and a coldata need to be provided. If none of the above is provided, it is possible to upload the data during the execution of the Shiny App
countmatrix	A count matrix, with genes as rows and samples as columns. If not provided, it is possible to upload the data during the execution of the Shiny App
coldata	A data.frame containing the info on the covariates of each sample. If not provided, it is possible to upload the data during the execution of the Shiny App
pca2go	An object generated by the pca2go function, which contains the information on enriched functional categories in the genes that show the top or bottom loadings in each principal component of interest. If not provided, it is possible to compute live during the execution of the Shiny App
annotation	A data.frame object, with row.names as gene identifiers (e.g. ENSEMBL ids) and a column, gene_name, containing e.g. HGNC-based gene symbols

Details

pcaExplorer provides functionality for interactive visualization of RNA-seq datasets based on Principal Components Analysis. The methods provided allow for quick information extraction and effective data exploration. A Shiny application encapsulates the whole analysis.

Value

A Shiny App is launched for interactive data exploration

Author(s)

Federico Marini <marinif@uni-mainz.de>, 2016

Maintainer: Federico Marini <marinif@uni-mainz.de>

Examples

```
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
                                             colData = colData(airway),
                                             design=~dex+cell)
## Not run:
rld_airway <- DESeq2::rlogTransformation(dds_airway)

pcaExplorer(dds_airway,rld_airway)

pcaExplorer(countmatrix = counts(dds_airway), coldata = colData(dds_airway))

pcaExplorer() # and then upload count matrix, covariate matrix (and eventual annotation)
```

```
## End(Not run)
```

`pcaplot` *Sample PCA plot for transformed data*

Description

Plots the results of PCA on a 2-dimensional space

Usage

```
pcaplot(x, intgroup = "condition", ntop = 500, returnData = FALSE,
  title = NULL, pcX = 1, pcY = 2, text_labels = TRUE, point_size = 3,
  ellipse = TRUE, ellipse.prob = 0.95)
```

Arguments

`x`	A `DESeqTransform` object, with data in `assay(x)`, produced for example by either `rlog` or `varianceStabilizingTransformation`
`intgroup`	Interesting groups: a character vector of names in `colData(x)` to use for grouping
`ntop`	Number of top genes to use for principal components, selected by highest row variance
`returnData`	logical, if TRUE returns a data.frame for further use, containing the selected principal components and intgroup covariates for custom plotting
`title`	The plot title
`pcX`	The principal component to display on the x axis
`pcY`	The principal component to display on the y axis
`text_labels`	Logical, whether to display the labels with the sample identifiers
`point_size`	Integer, the size of the points for the samples
`ellipse`	Logical, whether to display the confidence ellipse for the selected groups
`ellipse.prob`	Numeric, a value in the interval [0;1)

Value

An object created by `ggplot`, which can be assigned and further customized.

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- DESeq2::rlogTransformation(dds)
pcaplot(rlt, ntop=200)
```

pcaplot3d *Sample PCA plot for transformed data*

Description

Plots the results of PCA on a 3-dimensional space, interactively

Usage

```
pcaplot3d(x, intgroup = "condition", ntop = 500, returnData = FALSE,
  title = NULL, pcX = 1, pcY = 2, pcZ = 3, text_labels = TRUE,
  point_size = 3)
```

Arguments

x	A DESeqTransform object, with data in assay(x), produced for example by either rlog or varianceStabilizingTransformation
intgroup	Interesting groups: a character vector of names in colData(x) to use for grouping
ntop	Number of top genes to use for principal components, selected by highest row variance
returnData	logical, if TRUE returns a data.frame for further use, containing the selected principal components and intgroup covariates for custom plotting
title	The plot title
pcX	The principal component to display on the x axis
pcY	The principal component to display on the y axis
pcZ	The principal component to display on the z axis
text_labels	Logical, whether to display the labels with the sample identifiers
point_size	Integer, the size of the points for the samples

Value

A html-based visualization of the 3d PCA plot

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- DESeq2::rlogTransformation(dds)
pcaplot3d(rlt, ntop=200)
```

pcascree *Scree plot of the PCA on the samples*

Description

Produces a scree plot for investigating the proportion of explained variance, or alternatively the cumulative value

Usage

```
pcascree(obj, type = c("pev", "cev"), pc_nr = NULL, title = NULL)
```

Arguments

obj	A prcomp object
type	Display absolute proportions or cumulative proportion. Possible values: "pev" or "cev"
pc_nr	How many principal components to display max
title	Title of the plot

Value

An object created by ggplot, which can be assigned and further customized.

Examples

```
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- DESeq2::rlogTransformation(dds)
pcaobj <- prcomp(t(SummarizedExperiment::assay(rlt)))
pcascree(pcaobj,type="pev")
pcascree(pcaobj,type="cev",title="Cumulative explained proportion of variance - Test dataset")
```

plotPCcorrs *Plot significance of (cor)relations of covariates VS principal components*

Description

Plots the significance of the (cor)relation of each covariate vs a principal component

Usage

```
plotPCcorrs(pccorrs, pc = 1, logp = TRUE)
```

Arguments

pccorrs	A data.frame object generated by correlatePCs
pc	An integer number, corresponding to the principal component of interest
logp	Logical, defaults to TRUE, displays the -log10 of the pvalue instead of the p value itself

Value

A base plot object

Examples

```
library(DESeq2)
dds <- makeExampleDESeqDataSet_multifac(betaSD_condition = 3,betaSD_tissue = 1)
rlt <- rlogTransformation(dds)
pcaobj <- prcomp(t(assay(rlt)))
res <- correlatePCs(pcaobj,colData(dds))
plotPCcorrs(res)
```

---

topGOtable *Extract functional terms enriched in the DE genes, based on topGO*

Description

A wrapper for extracting functional GO terms enriched in the DE genes, based on the algorithm and the implementation in the topGO package

Usage

```
topGOtable(DEgenes, BGgenes, ontology = "BP", annot = annFUN.org,
  mapping = "org.Mm.eg.db", geneID = "symbol", topTablerows = 200,
  fullNamesInRows = TRUE, addGeneToTerms = TRUE, plotGraph = FALSE,
  plotNodes = 10, writeOutput = FALSE, outputFile = "")
```

Arguments

DEgenes	A vector of (differentially expressed) genes
BGgenes	A vector of background genes, e.g. all (expressed) genes in the assays
ontology	Which Gene Ontology domain to analyze: BP (Biological Process), MF (Molecular Function), or CC (Cellular Component)
annot	Which function to use for annotating genes to GO terms. Defaults to annFUN.org
mapping	Which org.XX.eg.db to use for annotation - select according to the species
geneID	Which format the genes are provided. Defaults to symbol, could also be entrez or ENSEMBL
topTablerows	How many rows to report before any filtering

fullNamesInRows	Logical, whether to display or not the full names for the GO terms
addGeneToTerms	Logical, whether to add a column with all genes annotated to each GO term
plotGraph	Logical, if TRUE additionally plots a graph on the identified GO terms
plotNodes	Number of nodes to plot
writeOutput	Logical, if TRUE additionally writes out the result to a file
outputFile	Name of the file the result should be written into

Value

A table containing the computed GO Terms and related enrichment scores

Examples

```
library(airway)
library(DESeq2)
data(airway)
airway
dds_airway <- DESeqDataSet(airway, design= ~ cell + dex)

# Example, performing extraction of enriched functional categories in
# detected significantly expressed genes

## Not run:
dds_airway <- DESeq(dds_airway)
res_airway <- results(dds_airway)
library("AnnotationDbi")
library("org.Hs.eg.db")
res_airway$symbol <- mapIds(org.Hs.eg.db,
                            keys=row.names(res_airway),
                            column="SYMBOL",
                            keytype="ENSEMBL",
                            multiVals="first")
res_airway$entrez <- mapIds(org.Hs.eg.db,
                            keys=row.names(res_airway),
                            column="ENTREZID",
                            keytype="ENSEMBL",
                            multiVals="first")
resOrdered <- as.data.frame(res_airway[order(res_airway$padj),])
de_df <- resOrdered[resOrdered$padj < .05 & !is.na(resOrdered$padj),]
de_symbols <- de_df$symbol
bg_ids <- rownames(dds_airway)[rowSums(counts(dds_airway)) > 0]
bg_symbols <- mapIds(org.Hs.eg.db,
                     keys=bg_ids,
                     column="SYMBOL",
                     keytype="ENSEMBL",
                     multiVals="first")
library(topGO)

topgoDE_airway <- topGOtable(de_symbols, bg_symbols,
                             ontology = "BP",
                             mapping = "org.Hs.eg.db",
                             geneID = "symbol")
```

```
## End(Not run)
```

Index