Package ‘mosdef’

May 30, 2024

Title MOSt frequently used and useful Differential Expression Functions

Version 1.0.0

Description This package provides functionality to run a number of tasks in the differential expression analysis workflow. This encompasses the most widely used steps, from running various enrichment analysis tools with a unified interface to creating plots and beautifying table components linking to external websites and databases. This streamlines the generation of comprehensive analysis reports.

Depends R (>= 4.4.0)

Imports DT, ggplot2, ggforce, ggrepel, htmltools, methods, AnnotationDbi, topGO, GO.db, clusterProfiler, goseq, utils, RColorBrewer, rlang, DESeq2, scales, SummarizedExperiment, S4Vectors, stats

Suggests knitr, rmarkdown, macrophage, org.Hs.eg.db, GeneTonic, testthat (>= 3.0.0), TxDb.Hsapiens.UCSC.hg38.knownGene, BiocStyle

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

VignetteBuilder knitr

URL https://github.com/imbeimainz/mosdef

BugReports https://github.com/imbeimainz/mosdef/issues

Config/testthat/edition 3

biocViews GeneExpression, Software, Transcription, Transcriptomics, DifferentialExpression, Visualization, ReportWriting, GeneSetEnrichment, GO

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

git_branch RELEASE_3_19

git_last_commit 1c35951
Contents

.git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29
Author Leon Dammer [aut] (<https://orcid.org/0009-0008-4132-7639>),
Federico Marini [aut, cre] (<https://orcid.org/0000-0003-3252-7758>)
Maintainer Federico Marini <marinif@uni-mainz.de>

Contents

.info_enrichrun .................................................. 3
buttonifier ......................................................... 3
create_link_dbPTM .............................................. 4
create_link_ENSEMBL ............................................. 5
create_link_GeneCards .......................................... 6
create_link_GO .................................................... 6
create_link_GTEX ............................................... 7
create_link_HPA .................................................. 7
create_link_NCBI ................................................. 8
create_link_PubMed .............................................. 9
create_link_UniProt ............................................. 9
deresult_to_df ................................................... 10
de_table_painter ................................................ 11
de_volcano ......................................................... 12
geneinfo_to_html ............................................... 14
gene_plot .......................................................... 15
get_expr_values ................................................. 17
go_to_html ......................................................... 18
go_volcano ........................................................ 19
map_to_color ...................................................... 20
mosdef-pkg ......................................................... 21
mosdef_de_container_check ................................... 22
mosdef_res_check ............................................... 23
plot_ma ............................................................ 23
res_enrich_macrophage_cluPro ................................ 25
res_enrich_macrophage_goseq ................................ 26
res_enrich_macrophage_topGO ................................ 26
res_macrophage_IFN
d vs naive .................................................. 27
run_cluPro ........................................................ 28
run_goseq ........................................................ 30
run_topGO ........................................................ 32
styleColorBar_divergent ...................................... 34

Index 36
.info_enrichrun

Description

Printing some info before the enrichment runs

Usage

.info_enrichrun(n_de, n_de_selected, de_type, res_de = NULL)

Arguments

n_de Numeric, number of DE genes (in total)
n_de_selected Character vector, containing the selected DE genes
de_type Character string, specifying up/down/both direction of DE regulation
res_de The res_de container as expected in most mosdef functions.

Value

Prints out an informative summary message.

Examples

# .info_enrichrun(10, c("geneA", "geneB"), "up")

buttonifier

Create sets of buttons for gene symbols

Description

A function to turn Gene Symbols into buttons in an Rmarkdown linking to various portals for further info about these genes.

Usage

buttonifier(
  df,
  create_buttons_to = c("PUBMED", "GC", "UNIPROT"),
  col_to_use = "SYMBOL",
  output_format = "DT",
  ens_col = NULL,
  ens_species = NULL
)
create_link_dbPTM

Arguments

df: A dataframe with at least one column with gene Symbols named: SYMBOL
create_buttons_to: At least one of: "GC", "NCBI", "GTEX", "UNIPROT", "dbPTM", "HPA", "PUBMED"
col_to_use: name of the columns were the gene symbols are stored. Default is SYMBOL
output_format: a parameter deciding which output format to return, either a "DT" (\texttt{DT::datatable()}
, recommended), or a simple dataframe ("DF"). In the latter case it is important that if the data is visualized with the \texttt{DT::datatable} function the parameter \texttt{escape} must be set to FALSE
ens_col: Character string, name of the columns were the ENSEMBL IDs are stored.
ens_species: The species you are working with to link to the correct gene on ENSEMBL

Details

Current supported portals are: GeneCards, NCBI, GTEx, Uniprot, dbPTM, Human Protein Atlas

Value

A data.frame or a \texttt{DT::datatable} object with columns adding HTML objects that link to websites with further information on the genes in question.

Examples

data(res_de_macrophage, package = "mosdef")

res_de <- res_macrophage_IFNg_vs_naive
res_df <- deresult_to_df(res_de)

## Subsetting for quicker run
res_df <- res_df[1:100, ]
buttonifier(res_df)

buttonifier(res_df,
create_buttons_to = c("NCBI", "HPA"),
ens_col = "id",
ens_species = "Homo_sapiens"
)

create_link_dbPTM \hspace{1cm} Link to dbPTM database

Description

Link to dbPTM database

Usage

create_link_dbPTM(val)
create_link_ENSEMBL

**Arguments**

val  Character, the gene symbol

**Value**

HTML for an action button

**Examples**

```r
create_link_dbPTM("Oct4")

data(res_de_macrophage, package = "mosdef")
res_macrophage_IFNg_vs_naive$SYMBOL <-
create_link_dbPTM(res_macrophage_IFNg_vs_naive$SYMBOL)
```

create_link_ENSEMBL  *Link to ENSEMBL database*

**Description**

Link to ENSEMBL database

**Usage**

```r
create_link_ENSEMBL(val, species = "Mus_musculus")
```

**Arguments**

val  Character, the gene symbol

species  The species to be analyzed e.g "Mus_musculus"

**Value**

HTML for an action button

**Examples**

```r
create_link_ENSEMBL("ENSMUSG00000024406")

data(res_de_macrophage, package = "mosdef")
rownames(res_macrophage_IFNg_vs_naive) <- create_link_ENSEMBL(
  rownames(res_macrophage_IFNg_vs_naive))
```
create_link_GeneCards  

*Link to the GeneCards database*

**Description**

Link to the GeneCards database

**Usage**

```r
create_link_GeneCards(val)
```

**Arguments**

- **val** Character, the gene symbol of interest

**Value**

HTML for an action button

**Examples**

```r
create_link_GeneCards("Oct4")
```

```r
data(res_de_macrophage, package = "mosdef")
res_macrophage_IFNg_vs_naive$SYMBOL <-
create_link_GeneCards(res_macrophage_IFNg_vs_naive$SYMBOL)
```

---

create_link_GO  

*Link to AMIGO database*

**Description**

Link to AMIGO database

**Usage**

```r
create_link_GO(val)
```

**Arguments**

- **val** Character, the GOID

**Value**

HTML for an action button
**create_link_GTEX**

**Examples**

```r
create_link_GO("GO:0008150")
```

**Description**

Link to the GTEx Portal

**Usage**

```r
create_link_GTEX(val)
```

**Arguments**

- `val` Character, the gene symbol of interest

**Value**

HTML for an action button

**Examples**

```r
create_link_GTEX("Oct4")
data(res_de_macrophage, package = "mosdef")
res_macrophage_IFNg_vs_naive$SYMBOL <-
create_link_GTEX(res_macrophage_IFNg_vs_naive$SYMBOL)
```

---

**create_link_HPA**

**Link to the Human Protein Atlas**

**Description**

Link to the Human Protein Atlas

**Usage**

```r
create_link_HPA(val)
```

**Arguments**

- `val` Character, the gene symbol
---

**create_link_NCBI**  
Link to NCBI database

### Description

Link to NCBI database

### Usage

```r
create_link_NCBI(val)
```

### Arguments

- **val**  
  Character, the gene symbol

### Value

HTML for an action button

### Examples

```r
create_link_NCBI("Oct4")

data(res_de_macrophage, package = "mosdef")
res_macrophage_IFNg_vs_naive$SYMBOL <-
  create_link_HPA(res_macrophage_IFNg_vs_naive$SYMBOL)
```
create_link_PubMed

Description
Link to Pubmed

Usage
create_link_PubMed(val)

Arguments
val Character, the gene symbol

Value
HTML for an action button

Examples
create_link_PubMed("Oct4")

data(res_de_macrophage, package = "mosdef")
res_macrophage_IFNg_vs_naive$SYMBOL <-
create_link_PubMed(res_macrophage_IFNg_vs_naive$SYMBOL)

create_link_UniProt

Description
Link to UniProt database

Usage
create_link_UniProt(val)

Arguments
val Character, the gene symbol

Value
HTML for an action button
Examples

```r
create_link_UniProt("Oct4")

data(res_de_macrophage, package = "mosdef")
res_macrophage_IFNg_vs_naive$SYMBOL <-
  create_link_UniProt(res_macrophage_IFNg_vs_naive$SYMBOL)
```

---

**deresult_to_df**

Generate a table from the DESeq2 results

**Description**

Generate a tidy table with the results of DESeq2

**Usage**

```r
deresult_to_df(res_de, FDR = NULL)
```

**Arguments**

- `res_de` - An object containing the results of the Differential Expression analysis workflow (e.g., DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.
- `FDR` - Numeric value, specifying the significance level for thresholding adjusted p-values. Defaults to NULL, which would return the full set of results without performing any subsetting based on FDR.

**Value**

A tidy data.frame with the results from differential expression, sorted by adjusted p-value. If FDR is specified, the table contains only genes with adjusted p-value smaller than the value.

**Examples**

```r
library("DESeq2")
library("macrophage")
data(res_de_macrophage, package = "mosdef")
head(res_macrophage_IFNg_vs_naive)
res_df <- deresult_to_df(res_macrophage_IFNg_vs_naive)
head(res_df)
```
**de_table_painter**  
*DE table painter*

**Description**

Beautifying the aspect and looks of a DE results table

**Usage**

```r
de_table_painter(
    res_de,
    rounding_digits = NULL,
    signif_digits = NULL,
    up_DE_color = "darkred",
    down_DE_color = "navyblue",
    logfc_column = "log2FoldChange",
    basemean_column = "baseMean",
    lfcse_column = "lfcSE",
    stat_column = "stat",
    pvalue_column = "pvalue",
    padj_column = "padj"
)
```

**Arguments**

- **res_de**
  
  An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework. Or a data frame obtained from such an object through `dresult_to_df()`

- **rounding_digits**
  
  Numeric value, specifying the number of digits to round the numeric values of the DE table (except the p-values)

- **signif_digits**
  
  Numeric value, specifying the number of significant digits to display for the p-values in the DE table

- **up_DE_color**
  
  Character string, specifying the color to use for coloring the bar of upregulated genes.

- **down_DE_color**
  
  Character string, specifying the color to use for coloring the bar of downregulated genes.

- **logfc_column**
  
  Character string, defining the name of the column in which to find the log2 fold change.

- **basemean_column**
  
  Character string, defining the name of the column in which to find the average expression value.

- **lfcse_column**
  
  Character string, defining the name of the column in which to find the standard error of the log2 fold change.
**Details**

Feeding on the classical results of DE workflows, this function formats and tries to prettify the representation of the key values in it.

**Value**

A **datatable** object, ready to be rendered as a widget inside an analysis Rmarkdown report.

**Examples**

```r
data(res_de_macrophage, package = "mosdef")
de_table_painter(res_macrophage_IFNg_vs_naive,
                 rounding_digits = 3,
                 signif_digits = 5)

## It is also possible to pass the "buttonified" table,
res_df_small <- deresult_to_df(res_macrophage_IFNg_vs_naive)[1:100, ]

buttonified_df <- buttonifier(res_df_small,
                               create_buttons_to = c("NCBI", "HPA"),
                               ens_col = "id",
                               ens_species = "Homo_sapiens",
                               output_format = "DF"
)

de_table_painter(buttonified_df,
                 rounding_digits = 3,
                 signif_digits = 5)
```

---

**de_volcano**

*Generates a volcano plot using ggplot2*

**Description**

This function generates a base volcanoplot for differentially expressed genes that can then be expanded upon using further ggplot functions.
Usage

```r
de_volcano(
    res_de,
    mapping = "org.Mm.eg.db",
    logfc_cutoff = 1,
    FDR = 0.05,
    labeled_genes = 30
)
```

Arguments

- `res_de`: An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.
- `mapping`: Which org.XX.eg.db package to use for annotation - select according to the species
- `logfc_cutoff`: A numeric value that sets the cutoff for the xintercept argument of ggplot
- `FDR`: The pvalue threshold to us for counting genes as de and therefore also where to draw the line in the plot. Default is 0.05
- `labeled_genes`: A numeric value describing the amount of genes to be labeled. This uses the Top(x) highest differentially expressed genes

Value

A ggplot2 volcano plot object that can be extended upon by the user

Examples

```r
library("ggplot2")
library("RColorBrewer")
library("ggrepel")
library("DESeq2")
library("org.Hs.eg.db")
data(res_de_macrophage, package = "mosdef")

p <- de_volcano(res_macrophage_IFNg_vs_naive,
    logfc_cutoff = 1,
    labeled_genes = 20,
    mapping = "org.Hs.eg.db"
)

p
```
geneinfo_to_html  Information on a gene

Description

Assembles information, in HTML format, regarding a gene symbol identifier

Usage

geneinfo_to_html(gene_id, res_de = NULL, col_to_use = "SYMBOL")

Arguments

gene_id  Character specifying the gene identifier for which to retrieve information

res_de  An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework. If not provided, the experiment-related information is not shown, and only some generic info on the identifier is displayed. The information about the gene is retrieved by matching on the SYMBOL column, which should be provided in res_de.

col_to_use  The column of your res_de object containing the gene symbols. Default is "SYMBOL”

Details

Creates links to the NCBI and the GeneCards databases

Value

HTML content related to a gene identifier, to be displayed in web applications (or inserted in Rmd documents)

Examples

geneinfo_to_html("ACTB")
geneinfo_to_html("Pf4")
gene_plot

Plot expression values for a gene

Description

Plot expression values (e.g. normalized counts) for a gene of interest, grouped by experimental group(s) of interest.

Usage

gene_plot(
  de_container,
  gene,
  intgroup = "condition",
  assay = "counts",
  annotation_obj = NULL,
  normalized = TRUE,
  transform = TRUE,
  labels_display = TRUE,
  labels_repel = TRUE,
  plot_type = "auto",
  return_data = FALSE
)

Arguments

de_container An object containing the data for a Differential Expression workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqDataSet object, normally obtained after running your data through the DESeq2 framework.

gene Character, specifies the identifier of the feature (gene) to be plotted

intgroup A character vector of names in colData(de_container) to use for grouping. Note: the vector components should be categorical variables.

assay Character, specifies with assay of the de_container object to use for reading out the expression values. Defaults to "counts".

annotation_obj A data.frame object with the feature annotation information, with at least two columns, gene_id and gene_name.

normalized Logical value, whether the expression values should be normalized by their size factor. Defaults to TRUE, applies when assay is "counts"

transform Logical value, corresponding whether to have log scale y-axis or not. Defaults to TRUE.

labels_display Logical value. Whether to display the labels of samples, defaults to TRUE.

labels_repel Logical value. Whether to use ggrepel’s functions to place labels; defaults to TRUE.
plot_type  Character, one of "auto", "jitteronly", "boxplot", "violin", or "sina". Defines the type of geom_ to be used for plotting. Defaults to auto, which in turn chooses one of the layers according to the number of samples in the smallest group defined via intgroup.

return_data  Logical, whether the function should just return the data.frame of expression values and covariates for custom plotting. Defaults to FALSE.

Details

The result of this function can be fed directly to plotly::ggplotly() for interactive visualization, instead of the static ggplot viz.

Value

A ggplot object

Examples

```r
library("macrophage")
library("DESeq2")
library("org.Hs.eg.db")

# dds object
data(gse, package = "macrophage")
dds_macrophage <- DESeqDataSet(gse, design = ~ line + condition)
rownames(dds_macrophage) <- substr(rownames(dds_macrophage), 1, 15)
keep <- rowSums(counts(dds_macrophage) >= 10) >= 6
dds_macrophage <- dds_macrophage[keep, ]
# dds_macrophage <- DESeq(dds_macrophage)

# annotation object
anno_df <- data.frame(
gen.id = rownames(dds_macrophage),
gen.name = mapIds(org.Hs.eg.db,
  keys = rownames(dds_macrophage),
  column = "SYMBOL",
  keytype = "ENSEMBL"
),
stringsAsFactors = FALSE,
row.names = rownames(dds_macrophage)
)

gene_plot(
de_container = dds_macrophage,
gene = "ENSG00000125347",
intgroup = "condition",
annotation_obj = anno_df
)
```
Description
Extract expression values, with the possibility to select other assay slots

Usage
get_expr_values(
  de_container,
  gene,
  intgroup,
  assay = "counts",
  normalized = TRUE
)

Arguments
- **de_container**: An object containing the data for a Differential Expression workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqDataSet object, normally obtained after running your data through the DESeq2 framework.
- **gene**: Character, specifies the identifier of the feature (gene) to be extracted
- **intgroup**: A character vector of names in colData(de_container) to use for grouping.
- **assay**: Character, specifies with assay of the de_container object to use for reading out the expression values. Defaults to "counts".
- **normalized**: Logical value, whether the expression values should be normalized by their size factor. Defaults to TRUE, applies when assay is "counts"

Value
A tidy data.frame with the expression values and covariates for further processing

Examples
library("macrophage")
library("DESeq2")
library("org.Hs.eg.db")
library("AnnotationDbi")

# dds object
data(gse, package = "macrophage")
dds_macrophage <- DESeqDataSet(gse, design = ~ line + condition)
rownames(dds_macrophage) <- substr(rownames(dds_macrophage), 1, 15)
keep <- rowSums(counts(dds_macrophage) >= 10) >= 6
dds_macrophage <- dds_macrophage[keep, ]
# dds_macrophage <- DESeq(dds_macrophage)
df_exp <- get_expr_values(
    de_container = dds_macrophage,
    gene = "ENSG00000125347",
    intgroup = "condition"
)
head(df_exp)

---

### go_to_html

**Information on a Gene Ontology identifier**

**Description**

Assembles information, in HTML format, regarding a Gene Ontology identifier

**Usage**

```r
go_to_html(go_id, res_enrich = NULL)
```

**Arguments**

- `go_id` Character, specifying the GeneOntology identifier for which to retrieve information
- `res_enrich` A `data.frame` object, storing the result of the functional enrichment analysis. If not provided, the experiment-related information is not shown, and only some generic info on the identifier is displayed.

**Details**

Also creates a link to the AmiGO database

**Value**

HTML content related to a GeneOntology identifier, to be displayed in web applications (or inserted in Rmd documents)

**Examples**

```r
go_to_html("GO:0002250")
go_to_html("GO:0043368")
```
go_volcano

Generates a volcano plot using ggplot2. This function generates a base volcano plot highlighting genes associated with a certain GOterm that can then be expanded upon using further ggplot functions.

Description

Generates a volcano plot using ggplot2. This function generates a base volcano plot highlighting genes associated with a certain GOterm that can then be expanded upon using further ggplot functions.

Usage

```r
go_volcano(
  res_de,
  res_enrich,
  mapping = "org.Hs.eg.db",
  term_index,
  logfc_cutoff = 1,
  FDR = 0.05,
  col_to_use = NULL,
  enrich_col = "genes",
  gene_col_separator = ",",
  down_col = "black",
  up_col = "black",
  highlight_col = "tomato",
  n_overlaps = 20
)
```

Arguments

- `res_de`: An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.
- `res_enrich`: A enrichment result object created by for example using `run_topGO()`.
- `mapping`: Which org.XX.eg.db package to use for annotation - select according to the species.
- `term_index`: The location (row) of your GO term of interest in your enrichment result.
- `logfc_cutoff`: A numeric value that sets the cutoff for the xintercept argument of ggplot.
- `FDR`: The pvalue threshold to us for counting genes as de and therefore also where to draw the line in the plot. Default is 0.05.
- `col_to_use`: The column in your differential expression results containing your gene symbols. If you don’t have one it is created automatically.
- `enrich_col`: column name from your res_enrich where the genes associated with your GOterm are stored (for example see the `run_topGO()` result in mosdef).
gene_col_separator
The separator used to split the genes. If you used topGO or goseq this is a "," which is the default. (For an example see the run_topGO() result in mosdef) If you used clusterProfiler this has to be set to "/". (For example see the run_cluPro() result in mosdef)

down_col
The colour for your downregulated genes, default is "gray"

up_col
The colour for your upregulated genes, default is "gray"

highlight_col
The colour for the genes associated with your GOterm default is "tomato"

n_overlaps
Number of overlaps ggrepel is supposed to allow when labeling (for more info check ggrepel documentation)

Value
A ggplot2 volcano plot object that can be extended upon by the user

Examples

library("org.Hs.eg.db")

data(res_de_macrophage, package = "mosdef")
data(res_enrich_macrophage_topGO, package = "mosdef")

p <- go_volcano(
  res_macrophage_IFNg_vs_naive,
  res_enrich = res_enrich_macrophage_topGO,
  term_index = 1,
  logfc_cutoff = 1,
  mapping = "org.Hs.eg.db",
  n_overlaps = 20
)

p

map_to_color
Maps numeric values to color values

Description
Maps numeric continuous values to values in a color palette

Usage
map_to_color(x, pal, symmetric = TRUE, limits = NULL)
Arguments

- **x**: A character vector of numeric values (e.g., log2FoldChange values) to be converted to a vector of colors.
- **pal**: A vector of characters specifying the definition of colors for the palette, e.g., obtained via `RColorBrewer::brewer.pal()`.
- **symmetric**: Logical value, whether to return a palette which is symmetrical with respect to the minimum and maximum values - "respecting" the zero. Defaults to `TRUE`.
- **limits**: A vector containing the limits of the values to be mapped. If not specified, defaults to the range of values in the `x` vector.

Value

A vector of colors, each corresponding to an element in the original vector.

Examples

```r
a <- 1:9
pal <- RColorBrewer::brewer.pal(9, "Set1")
map_to_color(a, pal)
plot(a, col = map_to_color(a, pal), pch = 20, cex = 4)

b <- 1:50
pal2 <- grDevices::colorRampPalette(
  RColorBrewer::brewer.pal(name = "RdYlBu", 11)
)(50)
plot(b, col = map_to_color(b, pal2), pch = 20, cex = 3)
```

**mosdef-pkg**

**mosdef**: mostly useful differential expression functions

**Description**

mostly useful differential expression functions

**Details**

This package provides functionality to run a number of tasks in the differential expression analysis workflow. This encompasses the most widely used steps, from running various enrichment analysis tools with a unified interface to creating plots and beautifying table components linking to external websites and databases. This streamlines the generation of comprehensive analysis reports.

**Author(s)**

**Maintainer**: Federico Marini <marinif@uni-mainz.de> (ORCID)

Authors:

- Leon Dammer <lc.dammer@gmail.com> (ORCID)
mosdef_de_container_check

A function checking if your de_container contains everything you need

Usage

mosdef_de_container_check(de_container, verbose = FALSE)

Arguments

de_container An object containing the data for a Differential Expression workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqDataSet object, normally obtained after running your data through the DESeq2 framework.

verbose Logical, whether to add messages telling the user which steps were taken.

Value

An invisible NULL after performing the checks

Examples

library("macrophage")
library("DESeq2")
data(gse, package = "macrophage")

dds_macrophage <- DESeqDataSet(gse, design = ~ line + condition)
rownames(dds_macrophage) <- substr(rownames(dds_macrophage), 1, 15)
keep <- rowSums(counts(dds_macrophage) >= 10) >= 6
dds_macrophage <- dds_macrophage[keep, ]
# dds_macrophage <- DESeq(dds_macrophage)
mosdef_de_container_check(dds_macrophage)
mosdef_res_check  

Description

A function checking if your res_de contains everything you need

Usage

mosdef_res_check(res_de, verbose = FALSE)

Arguments

res_de  
An object containing the results of the Differential Expression analysis workflow  
(e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object  
created using the DESeq2 framework.

verbose  
Logical, whether to add messages telling the user which steps were taken

Value

An invisible NULL after performing the checks

Examples

data(res_de_macrophage, package = "mosdef")
mosdef_res_check(res_macrophage_IFNg_vs_naive)

plot_ma  

MA-plot from base means and log fold changes

Description

MA-plot from base means and log fold changes, in the ggplot2 framework, with additional support  
to annotate genes if provided.

Usage

plot_ma(  
res_de,  
FDR = 0.05,  
point_alpha = 0.2,  
sig_color = "red",  
annotation_obj = NULL,  
draw_y0 = TRUE,  
hlines = NULL,  
)
plot_ma

title = NULL,
xlab = "mean of normalized counts - log10 scale",
ylim = NULL,
add_rug = TRUE,
intgenes = NULL,
intgenes_color = "steelblue",
labels_intgenes = TRUE,
labels_repel = TRUE
)

Arguments

res_de  An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.

FDR  Numeric value, the significance level for thresholding adjusted p-values

point_alpha  Alpha transparency value for the points (0 = transparent, 1 = opaque)

sig_color  Color to use to mark differentially expressed genes. Defaults to red

annotation_obj  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. Optional

draw_y0  Logical, whether to draw the horizontal line at y=0. Defaults to TRUE.

hlines  The y coordinate (in absolute value) where to draw horizontal lines, optional

title  A title for the plot, optional

xlab  X axis label, defaults to "mean of normalized counts - log10 scale"

ylim  Vector of two numeric values, Y axis limits to restrict the view

add_rug  Logical, whether to add rug plots in the margins

intgenes  Vector of genes of interest. Gene symbols if a symbol column is provided in res_de, or else the identifiers specified in the row names

intgenes_color  The color to use to mark the genes on the main plot.

labels_intgenes  Logical, whether to add the gene identifiers/names close to the marked plots

labels_repel  Logical, whether to use ggrepel::geom_text_repel for placing the labels on the features to mark

Details

The genes of interest are to be provided as gene symbols if a symbol column is provided in res_de, or else by using the identifiers specified in the row names

Value

An object created by ggplot
Examples

```r
data(res_de_macrophage, package = "mosdef")

plot_ma(res_macrophage_IFNg_vs_naive, FDR = 0.05, hlines = 1)

plot_ma(res_macrophage_IFNg_vs_naive,
        FDR = 0.1,
        intgenes = c(
            "ENSG00000103196", # CRISPLD2
            "ENSG00000120129", # DUSP1
            "ENSG00000163884", # KLF15
            "ENSG00000179094" # PER1
        )
)
```

res_enrich_macrophage_cluPro

A sample enrichment object

Description

A sample enrichment object, generated in the mosdef and clusterProfiler framework

Format

An enrichResult object

Details

This enrichment object is on the data from the macrophage package. Specifically, this set of enrichment results was created using the Biological Process ontology, mapping the gene identifiers through the org.Hs.eg.db package.

Source

Details on how this object has been created are included in the create_mosdef_data.R script, included in the (installed) inst/scripts folder of the mosdef package. This is also available at https://github.com/imbeimainz/mosdef/blob/devel/inst/scripts/create_mosdef_data.R

References


See Also

res_macrophage_IFNg_vs_naive
res_enrich_macrophage_goseq

A sample enrichment object

Description
A sample enrichment object, generated in the mosdef and goseq framework

Format
A data.frame object

Details
This enrichment object is on the data from the macrophage package
Specifically, this set of enrichment results was created using the Biological Process ontology, mapping the gene symbol identifiers through the org.Hs.eg.db package - the gene length information is retrieved by the internal routines of goseq.

Source
Details on how this object has been created are included in the create_mosdef_data.R script, included in the (installed) inst/scripts folder of the mosdef package. This is also available at https://github.com/imbeimainz/mosdef/blob/devel/inst/scripts/create_mosdef_data.R

References

See Also
res_macrophage_IFNg_vs_naive

res_enrich_macrophage_topGO

A sample enrichment object

Description
A sample enrichment object, generated in the mosdef and topGO framework

Format
A data.frame object
Details

This enrichment object is on the data from the macrophage package.
Specifically, this set of enrichment results was created using the Biological Process ontology, mapping the gene symbol identifiers through the org.Hs.eg.db package.

Source

Details on how this object has been created are included in the create_mosdef_data.R script, included in the (installed) inst/scripts folder of the mosdef package. This is also available at https://github.com/imbeimainz/mosdef/blob/devel/inst/scripts/create_mosdef_data.R

References


See Also

res_macrophage_IFNg_vs_naive
References


---

run_cluPro

Extract functional terms enriched in the DE genes, based on clusterProfiler

Description

A wrapper for extracting functional GO terms enriched in a list of (DE) genes, based on the algorithm and the implementation in the clusterProfiler package

Usage

```r
run_cluPro(
  de_container = NULL, 
  res_de = NULL, 
  de_genes = NULL, 
  bg_genes = NULL, 
  top_de = NULL, 
  FDR_threshold = 0.05, 
  min_counts = 0, 
  mapping = "org.Hs.eg.db", 
  de_type = "up_and_down", 
  keyType = "SYMBOL", 
  verbose = TRUE, 
  ...
)
```

Arguments

- **de_container**: An object containing the data for a Differential Expression workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqDataSet object, normally obtained after running your data through the DESeq2 framework.
- **res_de**: An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.
- **de_genes**: A vector of (differentially expressed) genes
- **bg_genes**: A vector of background genes, e.g. all (expressed) genes in the assays
- **top_de**: numeric, how many of the top differentially expressed genes to use for the enrichment analysis. Attempts to reduce redundancy. Assumes the data is sorted by p adj (default in DESeq2).
- **FDR_threshold**: The pvalue threshold to us for counting genes as de. Default is 0.05
run_cluPro

- **min_counts**: numeric, min number of counts a gene needs to have to be included in the gene-set that the de genes are compared to. Default is 0, recommended only for advanced users.

- **mapping**: Which *org.XX.eg.db* package to use for annotation - select according to the species.

- **de_type**: One of: 'up', 'down', or 'up_and_down' Which genes to use for GOterm calculations.

- **keyType**: Gene format to input into enrichGO from clusterProfiler. If res_de and de_container are used use "SYMBOL" for more information check the enrichGO documentation.

- **verbose**: Logical, whether to add messages telling the user which steps were taken.

Further parameters to use for the *clusterProfiler::enrichGO()* function from clusterProfiler.

**Value**

A table containing the computed GO Terms and related enrichment scores.

**See Also**

*clusterProfiler::enrichGO()* for the underlying method.

Other Enrichment functions: *run_goseq()*, *run_topGO()*

**Examples**

```r
library("macrophage")
library("DESeq2")
data(gse, package = "macrophage")

dds_macrophage <- DESeqDataSet(gse, design = ~ line + condition)
rownames(dds_macrophage) <- substr(rownames(dds_macrophage), 1, 15)
keep <- rowSums(counts(dds_macrophage) >= 10) >= 6
dds_macrophage <- dds_macrophage[keep, ]
dds_macrophage <- DESeq(dds_macrophage)
data(res_de_macrophage, package = "mosdef")

library("AnnotationDbi")
library("org.Hs.eg.db")
library("clusterProfiler")
CluProde_macrophage <- run_cluPro(
  res_de = res_macrophage_IFNg_vs_naive,
  de_container = dds_macrophage,
  mapping = "org.Hs.eg.db"
)
```
run_goseq

Extract functional terms enriched in the DE genes, based on goseq

Description

A wrapper for extracting functional GO terms enriched in a list of (DE) genes, based on the algorithm and the implementation in the goseq package.

Usage

```r
run_goseq(
  de_container = NULL,
  res_de = NULL,
  de_genes = NULL,
  bg_genes = NULL,
  top_de = NULL,
  FDR_threshold = 0.05,
  min_counts = 0,
  genome = "hg38",
  id = "ensGene",
  de_type = "up_and_down",
  testCats = c("GO:BP", "GO:MF", "GO:CC"),
  mapping = "org.Hs.eh.db",
  add_gene_to_terms = TRUE,
  verbose = TRUE
)
```

Arguments

- **de_container**: An object containing the data for a Differential Expression workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqDataSet object, normally obtained after running your data through the DESeq2 framework.
- **res_de**: An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.
- **de_genes**: A vector of (differentially expressed) genes
- **bg_genes**: A vector of background genes, e.g. all (expressed) genes in the assays
- **top_de**: numeric, how many of the top differentially expressed genes to use for the enrichment analysis. Attempts to reduce redundancy. Assumes the data is sorted by padj (default in DESeq2).
- **FDR_threshold**: The pvalue threshold to us for counting genes as de. Default is 0.05
- **min_counts**: numeric, min number of counts a gene needs to have to be included in the gene-set that the de genes are compared to. Default is 0, recommended only for advanced users.
run_goseq

A string identifying the genome that genes refer to, as in the `goseq::goseq()` function

id

A string identifying the gene identifier used by genes, as in the `goseq::goseq()` function

de_type

One of: 'up', 'down', or 'up_and_down' Which genes to use for GOterm calculations: upregulated, downregulated or both

testCats

A vector specifying which categories to test for overrepresentation amongst DE genes - can be any combination of "GO:CC", "GO:BP", "GO:MF" & "KEGG"

mapping

Character string, named as the `org.XX.eg.db` package which should be available in Bioconductor

add_gene_to_terms

Logical, whether to add a column with all genes annotated to each GO term

verbose

Logical, whether to add messages telling the user which steps were taken

Details

Note: the feature length retrieval is based on the `goseq::goseq()` function, and requires that the corresponding TxDb packages are installed and available

Value

A table containing the computed GO Terms and related enrichment scores

See Also

`goseq::goseq()` for the underlying method

Other Enrichment functions: `run_cluPro()`, `run_topGO()`

Examples

```r
library("macrophage")
library("DESeq2")
data(gse, package = "macrophage")

dds_macrophage <- DESeqDataSet(gse, design = ~ line + condition)
rownames(dds_macrophage) <- substr(rownames(dds_macrophage), 1, 15)
keep <- rowSums(counts(dds_macrophage) >= 10) >= 6
dds_macrophage <- dds_macrophage[keep, ]
dds_macrophage <- DESeq(dds_macrophage)

data(res_de_macrophage, package = "mosdef")
res_de <- res_macrophage_IFNg_vs_naive
mygo <- run_goseq(
  res_de = res_macrophage_IFNg_vs_naive,
  de_container = dds_macrophage,
  mapping = "org.Hs.eg.db",
  testCats = "GO:BP",
  add_gene_to_terms = TRUE
)
```
head(mygo)

run_topGO

---

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

```r
run_topGO(
  de_container = NULL,
  res_de = NULL,
  de_genes = NULL,
  bg_genes = NULL,
  top_de = NULL,
  FDR_threshold = 0.05,
  min_counts = 0,
  ontology = "BP",
  annot = annFUN.org,
  mapping = "org.Mm.eg.db",
  gene_id = "symbol",
  full_names_in_rows = TRUE,
  add_gene_to_terms = TRUE,
  de_type = "up_and_down",
  topGO_method2 = "elim",
  do_padj = FALSE,
  verbose = TRUE
)
```

Arguments

- `de_container` An object containing the data for a Differential Expression workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqDataSet object, normally obtained after running your data through the DESeq2 framework.
- `res_de` An object containing the results of the Differential Expression analysis workflow (e.g. DESeq2, edgeR or limma). Currently, this can be a DESeqResults object created using the DESeq2 framework.
- `de_genes` A vector of (differentially expressed) genes
- `bg_genes` A vector of background genes, e.g. all (expressed) genes in the assays
- `top_de` numeric, how many of the top differentially expressed genes to use for the enrichment analysis. Attempts to reduce redundancy. Assumes the data is sorted by padj (default in DESeq2).
run_topGO

FDR_threshold  The pvalue threshold to us for counting genes as de. Default is 0.05
min_counts    numeric, min number of counts a gene needs to have to be included in the gene-set that the de genes are compared to. Default is 0, recommended only for advanced users.
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 package to use for annotation - select according to the species
gene_id       Which format the genes are provided. Defaults to symbol, could also be entrez or ENSEMBL
full_names_in_rows Logical, whether to display or not the full names for the GO terms
add_gene_to_terms Logical, whether to add a column with all genes annotated to each GO term
de_type       One of: 'up', 'down', or 'up_and_down' Which genes to use for GOnet calculations: upregulated, downregulated or both
topGO_method2 Character, specifying which of the methods implemented by topGO should be used, in addition to the classic algorithm. Defaults to elim.
do_padj       Logical, whether to perform the adjustment on the p-values from the specific topGO method, based on the FDR correction. Defaults to FALSE, since the assumption of independent hypotheses is somewhat violated by the intrinsic DAG-structure of the Gene Ontology Terms
verbose       Logical, whether to add messages telling the user which steps were taken

Details

Allowed values assumed by the topGO_method2 parameter are one of the following: elim, weight, weight01, lea, parentchild. For more details on this, please refer to the original documentation of the topGO package itself

Value

A table containing the computed GO Terms and related enrichment scores

See Also

topGO::topGOdata-class() and topGO::runTest() for the class objects and underlying methods

Other Enrichment functions: run_cluPro(), run_goseq()

Examples

library("macrophage")
library("DESeq2")
data(gse, package = "macrophage")
dds_macrophage <- DESeqDataSet(gse, design = ~ line + condition)
rownames(dds_macrophage) <- substr(rownames(dds_macrophage), 1, 15)
keep <- rowSums(counts(dds_macrophage) >= 10) >= 6
dds_macrophage <- dds_macrophage[keep, ]
dds_macrophage <- DESeq(dds_macrophage)
data(res_de_macrophage, package = "mosdef")

library("AnnotationDbi")
library("org.Hs.eg.db")
library("topGO")
topgoDE_macrophage <- run_topGO(
  de_container = dds_macrophage,
  res_de = res_macrophage_IFNg_vs naive,
  ontology = "BP",
  mapping = "org.Hs.eg.db",
  gene_id = "symbol",
)

---

**styleColorBar_divergent**

*Style DT color bars*

**Description**

Style DT color bars for values that diverge from 0.

**Usage**

```
styleColorBar_divergent(data, color_pos, color_neg)
```

**Arguments**

- `data` The numeric vector whose range will be used for scaling the table data from 0-100 before being represented as color bars. A vector of length 2 is acceptable here for specifying a range possibly wider or narrower than the range of the table data itself.
- `color_pos` The color of the bars for the positive values
- `color_neg` The color of the bars for the negative values

**Details**

This function draws background color bars behind table cells in a column, width the width of bars being proportional to the column values *and* the color dependent on the sign of the value.

A typical usage is for values such as log2FoldChange for tables resulting from differential expression analysis. Still, the functionality of this can be quickly generalized to other cases - see in the examples.
The code of this function is heavily inspired from styleColorBar, and borrows at full hands from an excellent post on StackOverflow - https://stackoverflow.com/questions/33521828/stylecolorbar-center-and-shift-left-right-dependent-on-sign/33524422#33524422

Value

This function generates JavaScript and CSS code from the values specified in R, to be used in DT tables formatting.

Examples

# With a very simple data frame

```r
simplest_df <- data.frame(
  a = c(rep("a", 9)),
  value = c(-4, -3, -2, -1, 0, 1, 2, 3, 4)
)
```

```r
library("DT")
DT::datatable(simplest_df) |>
  formatStyle(
    "value",
    background = styleColorBar_divergent(
      simplest_df$value,
      scales::alpha("forestgreen", 0.4),
      scales::alpha("gold", 0.4)
    ),
    backgroundSize = "100% 90%",
    backgroundRepeat = "no-repeat",
    backgroundPosition = "center"
  )
```
Index

* Enrichment functions
  - run_cluPro, 28
  - run_goseq, 30
  - run_topGO, 32
* internal
  - mosdef-pkg, 21
  - .info_enrichrun, 3
  - buttonifier, 3
  - clusterProfiler::enrichGO(), 29
  - create_link_dbPTM, 4
  - create_link_ENSEMBL, 5
  - create_link_GeneCards, 6
  - create_link_GO, 6
  - create_link_GTEX, 7
  - create_link_HPA, 7
  - create_link_NCBI, 8
  - create_link_GTex, 9
  - create_link_PubMed, 9
  - create_link_UniProt, 9
  - de_table_painter, 11
  - de_volcano, 12
  - deresult_to_df, 10
  - deresult_to_df(), 11
  - DT::datatable(), 4
  - gene_plot, 15
  - geneinfo_to_html, 14
  - get_expr_values, 17
  - go_to_html, 18
  - go_volcano, 19
  - goseq::goseq(), 31
  - map_to_color, 20
  - mosdef (mosdef-pkg), 21
  - mosdef-package (mosdef-pkg), 21
  - mosdef-pkg, 21
  - mosdef_de_container_check, 22
  - mosdef_res_check, 23
  - plot_ma, 23
  - plotly::ggplotly(), 16
  - RColorBrewer::brewer.pal(), 21
  - res_enrich_macrophage_cluPro, 25
  - res_enrich_macrophage_goseq, 26
  - res_enrich_macrophage_topGO, 26
  - res_macrophage_IFNg_vs_naive, 25–27, 27
  - run_cluPro, 28, 31, 33
  - run_cluPro(), 20
  - run_goseq, 29, 30, 33
  - run_topGO, 29, 31, 32
  - run_topGO(), 19, 20
  - styleColorBar_divergent, 34
  - topGO::runTest(), 33