# Package ‘ideal’

March 27, 2024

**Type**  Package  
**Title**  Interactive Differential Expression AnaLysis  
**Version**  1.26.0  
**Date**  2023-04-28  
**Description**  This package provides functions for an Interactive Differential Expression AnaLysis of RNA-sequencing datasets, to extract quickly and effectively information downstream the step of differential expression. A Shiny application encapsulates the whole package.  
**License**  MIT + file LICENSE  
**LazyData**  TRUE  
**Depends**  topGO  
**Imports**  DESeq2, SummarizedExperiment, GenomicRanges, IRanges, S4Vectors, ggplot2 (>= 2.0.0), heatmaply, plotly, pheatmap, pcaExplorer, IHW, gplots, UpSetR, goseq, stringr, dplyr, limma, GOstats, GO.db, AnnotationDbi, shiny (>= 0.12.0), shinydashboard, shinyBS, DT, rentrez, rintrojs, ggrepel, knitr, rmarkdown, shinyAce, BiocParallel, grDevices, base64enc, methods  
**Suggests**  testthat, BiocStyle, markdown, airway, org.Hs.eg.db, TxDb.Hsapiens.UCSC.hg38.knownGene, DEFormats, edgeR  
**BugReports**  https://github.com/federicomarini/ideal/issues  
**biocViews**  ImmunoOncology, GeneExpression, DifferentialExpression, RNASeq, Sequencing, Visualization, QualityControl, GUI, GeneSetEnrichment, ReportWriting, ShinyApps  
**VignetteBuilder**  knitr  
**RoxygenNote**  7.2.3  
**Encoding**  UTF-8
**deseqresult2DEgenes**

Generate a tidy table with the DE genes from the results of DESeq

**Description**

Generate a tidy table with the DE genes from the results of DESeq

**Usage**

deseqresult2DEgenes(deseqresult, FDR = 0.05)

**Arguments**

- **deseqresult** A `DESeqResults` object
- **FDR** Numeric value, the significance level for thresholding adjusted p-values

**Value**

A "tidy" data.frame with only genes marked as differentially expressed
Examples

```r
# with simulated data...
library(DESeq2)
dds <- DESeq2::makeExampleDESeqDataSet(n=100, m=8, betaSD=2)
dds <- DESeq(dds)
res <- results(dds)
deseqresult2DEgenes(res)
```

**deseqresult2tbl**

Generate a tidy table with the results of DESeq

**Description**

Generate a tidy table with the results of DESeq

**Usage**

`deseqresult2tbl(deseqresult)`

**Arguments**

- `deseqresult`: A `DESeqResults` object

**Value**

A "tidy" data frame with all genes

**Examples**

```r
# with simulated data...
library(DESeq2)
dds <- DESeq2::makeExampleDESeqDataSet(n=100, m=8, betaSD=1)
dds <- DESeq2::DESeq(dds)
res <- DESeq2::results(dds)
deseqresult2tbl(res)
```

**ggplotCounts**

Plot normalized counts for a gene

**Description**

Plot for normalized counts of a single gene, with jittered points superimposed on the boxplot
Usage

```r
ggplotCounts(
  dds,
  gene,
  intgroup = "condition",
  annotation_obj = NULL,
  transform = TRUE,
  labels_repel = TRUE
)
```

Arguments

- **dds**: A `DESeqDataSet` object.
- **gene**: A character, specifying the name of the gene to plot.
- **intgroup**: Interesting groups: a character vector of names in `colData(dds)` to use for grouping.
- **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.
- **transform**: Logical value, corresponding whether to have log scale y-axis or not. Defaults to `TRUE`.
- **labels_repel**: Logical value. Whether to use `ggrepel`'s functions to place labels; defaults to `TRUE`.

Details

Note: this function relies on the `plotCounts` function of DESeq2, therefore pseudocounts of 0.5 are added to each point.

Value

An object created by `ggplot`

Examples

```r
library(airway)
data(airway)
airway

dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
  colData = colData(airway),
  design = ~ cell + dex)

ggplotCounts(dds_airway,
  gene = "ENSG00000103196", # CRISPLD2 in the original publication
  intgroup = "dex")
```
**goseqTable**

*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
goseqTable(
  de.genes,
  assayed.genes,
  genome = "hg38",
  id = "ensGene",
  testCats = c("GO:BP", "GO:MF", "GO:CC"),
  FDR_GO_cutoff = 1,
  nTop = 200,
  orgDbPkg = "org.Hs.eg.db",
  addGeneToTerms = TRUE
)
```

**Arguments**

- **de.genes**: A vector of (differentially expressed) genes
- **assayed.genes**: A vector of background genes, e.g. all (expressed) genes in the assays
- **genome**: A string identifying the genome that genes refer to, as in the `goseq` function
- **id**: A string identifying the gene identifier used by genes, as in the `goseq` function
- **testCats**: A vector specifying which categories to test for over representation amongst DE genes - can be any combination of "GO:CC", "GO:BP", "GO:MF" & "KEGG"
- **FDR_GO_cutoff**: Numeric value for subsetting the results
- **nTop**: Number of categories to extract, and optionally process for adding genes to the respective terms
- **orgDbPkg**: Character string, named as the `org.XX.eg.db` package which should be available in Bioconductor
- **addGeneToTerms**: Logical, whether to add a column with all genes annotated to each GO term

**Details**

Note: the feature length retrieval is based on the `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
Examples

```r
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
  colData = colData(airway),
  design = ~ cell + dex
)
.dds_airway <- DESeq2::DESeq(dds_airway)
res_airway <- DESeq2::results(dds_airway)

res_subset <- deseqresult2DEgenes(res_airway)[1:100, ]
myde <- res_subset$id
myassayed <- rownames(res_airway)
mygo <- goseqTable(myde, myassayed,
  testCats = "GO:BP",
  addGeneToTerms = FALSE
)
head(mygo)
```

## Not run:

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

---

ideal

### ideal: Interactive Differential Expression Analysis

Description

ideal makes differential expression analysis interactive, easy and reproducible. This function launches the main application included in the package.

Usage

```r
ideal(
  dds_obj = NULL,
  res_obj = NULL,
  annotation_obj = NULL,
  countmatrix = NULL,
  expdesign = NULL,
  gene_signatures = NULL
)
```

Arguments

- **dds_obj** A `DESeqDataSet` object. If not provided, then a `countmatrix` and a `expdesign` 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
ideal

res_obj A **DESeqResults** object. If not provided, it can be computed during the execution of the application

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. If not provided, it can be constructed during the execution via the org.eg.XX.db packages - these need to be installed

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

extdesign 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

gene_signatures A list of vectors, one for each pathway/signature. This is for example the output of the **read_gmt** function. The provided object can also be replaced during runtime in the dedicated upload widget.

**Value**

A Shiny App is launched for interactive data exploration and differential expression analysis

**Examples**

```r
# with simulated data...
library(DESeq2)
dds <- DESeq2::makeExampleDESeqDataSet(n = 100, m = 8)
cm <- counts(dds)
cd <- colData(dds)

# with the well known airway package...
library(airway)
data(airway)
airway

dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
  colData = colData(airway),
  design = ~ cell + dex
)
## Not run:
ideal()
ideal(dds)
ideal(dds_airway)

dds_airway <- DESeq2::DESeq(dds_airway)
res_airway <- DESeq2::results(dds_airway)
ideal(dds_airway, res_airway)
## End(Not run)
```
ideal-pkg  

*ideal: Interactive Differential Expression Analysis*

---

**Description**

ideal makes differential expression analysis interactive, easy and reproducible. The analysis of RNA-seq datasets is guided by the Shiny app as main component of the package, which also provides a wide set of functions to efficiently extract information from the existing data. The app can be also deployed on a Shiny server, to allow its usage without any installation on the user’s side.

**Details**

ideal makes differential expression analysis interactive, easy and reproducible. The analysis of RNA-seq datasets is guided by the Shiny app as main component of the package, which also provides a wide set of functions to efficiently extract information from the existing data. The app can be also deployed on a Shiny server, to allow its usage without any installation on the user’s side.

**Author(s)**

Federico Marini <marinif@uni-mainz.de>, 2016-2017
Maintainer: Federico Marini <marinif@uni-mainz.de>

---

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**

```r
plot_ma(
  res_obj,
  FDR = 0.05,
  point_alpha = 0.2,
  sig_color = "red",
  annotation_obj = NULL,
  draw_y0 = TRUE,
  hlines = NULL,
  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_obj**: A `DESeqResults` object
- **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_obj`, 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 `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_obj`, or else by using the identifiers specified in the row names.

Value

An object created by `ggplot`

Examples

```r
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
  colData = colData(airway),
  design = ~ cell + dex)
```
## plot_volcano

Volcano plot for log fold changes and log p-values

### Description

Volcano plot for log fold changes and log p-values in the ggplot2 framework, with additional support to annotate genes if provided.

### Usage

```r
plot_volcano(
  res_obj,
  FDR = 0.05,
  ylim_up = NULL,
  vlines = NULL,
  title = NULL,
  intgenes = NULL,
  intgenes_color = "steelblue",
  labels_intgenes = TRUE,
  labels_repel = TRUE
)
```
Arguments

res_obj  A DESeqResults object
FDR      Numeric value, the significance level for thresholding adjusted p-values
ylim_up  Numeric value, Y axis upper limits to restrict the view
vlines   The x coordinate (in absolute value) where to draw vertical lines, optional
title    A title for the plot, optional
intgenes Vector of genes of interest. Gene symbols if a symbol column is provided in res_obj, 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 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_obj, or else using the identifiers specified in the row names

Value

An object created by ggplot

Examples

library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
  colData = colData(airway),
  design = ~ cell + dex
)

# subsetting for quicker run, ignore the next two commands if regularly using the function
gene_subset <- c(
    "ENSG00000103196", # CRISPLD2
    "ENSG00000120129", # DUSP1
    "ENSG00000163884", # KLF15
    "ENSG00000179094", # PER1
)
rownames(dds_airway)[rep(c(rep(FALSE, 99), TRUE), length.out = nrow(dds_airway))]

dds_airway <- dds_airway[gene_subset,]

dds_airway <- DESeq2::DESeq(dds_airway)
res_airway <- DESeq2::results(dds_airway)

plot_volcano(res_airway)
**read_gmt**  
*Read in a GMT file*

**Description**

Returns a list of pathways from a GMT file.

**Usage**

```r
read_gmt(gmtfile)
```

**Arguments**

- `gmtfile`: A character value, containing the location of the GMT formatted file. It can also be a file found online.

**Value**

A list of vectors, one for each pathway in the GMT file.

**Examples**

```r
# this example reads in the freely available pathways from wikipathways
## Not run:
mysigs <- read_gmt(
    "http://data.wikipathways.org/20180910/gmt/wikipathways-20180910-gmt-Homo_sapiens.gmt"
)
head(mysigs)
# see how the gene identifiers are encoded as ENTREZ id

## End(Not run)
```

**sepguesser**  
*Make an educated guess on the separator character*

**Description**

This function tries to guess which separator was used in a text delimited file.

**Usage**

```r
sepguesser(file, sep_list = c("," , "\t", ";", " "))
```

**Arguments**

- `file`: The name of the file which the data are to be read from.
- `sep_list`: A vector containing the candidates for being identified as separators. Defaults to `c("," , "\t", ";", " ")`. 
Value
A character value, corresponding to the guessed separator. One of "," (comma), "\t" (tab), ";" (semicolon), "," (whitespace)

Examples
sepguesser(system.file("extdata/design_commas.txt", package = "ideal"))
sepguesser(system.file("extdata/design_semicolons.txt", package = "ideal"))
sepguesser(system.file("extdata/design_spaces.txt", package = "ideal"))
mysep <- sepguesser(system.file("extdata/design_tabs.txt", package = "ideal"))
# to be used for reading in the same file, without having to specify the sep

---

**sig_heatmap**

*Plot a heatmap of the gene signature on the data*

Description
Plot a heatmap for the selected gene signature on the provided data, with the possibility to compactly display also DE only genes

Usage
```
sig_heatmap(
  vst_data,
  my_signature,
  res_data = NULL,
  FDR = 0.05,
  de_only = FALSE,
  annovec,
  title = "",
  cluster_rows = TRUE,
  cluster_cols = FALSE,
  anno_colData = NULL,
  center_mean = TRUE,
  scale_row = FALSE
)
```

Arguments
- **vst_data**: A `DESeqTransform` object - usually the variance stabilized transformed data, which will be used to extract the expression values
- **my_signature**: A character vector, usually named, containing the genes which compose the gene signature
- **res_data**: A `DESeqResults` object. If not provided, it can be computed during the execution of the application
FDR  Numeric value between 0 and 1, the False Discovery Rate

de_only Logical, whether to display only DE genes belonging to the pathway - defaults to FALSE

annovec A named character vector, with the corresponding annotation across IDs

title Character, title for the heatmap

cluster_rows Logical, whether to cluster rows - defaults to TRUE

cluster_cols Logical, whether to cluster column - defaults to FALSE. Recommended to be set to TRUE if de_only is also set to TRUE

anno_colData Character vector, specifying the elements of the colData information to be displayed as a decoration of the heatmap. Can be a vector of any length, as long as these names are included as colData. Defaults to NULL, which would plot no annotation on the samples.

center_mean Logical, whether to perform mean centering on the expression values. Defaults to TRUE, as it improves the general readability of the heatmap

scale_row Logical, whether to perform row-based standardization of the expression values

Value

A plot based on the pheatmap function

Examples

# with the well known airway package...
library(airway)
data(airway)
airway
dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
  colData = colData(airway),
  design = ~ cell + dex
)
## Not run:
dds_airway <- DESeq2::DESeq(dds_airway)
res_airway <- DESeq2::results(dds_airway)
vst_airway <- DESeq2::vst(dds_airway)
library(org.Hs.eg.db)
annovec <- mapIds(org.Hs.eg.db, rownames(dds_airway), "ENTREZID", "ENSEMBL")
mysignatures <- read_gmt("http://data.wikipathways.org/20190210/gmt/wikipathways-20190210-gmt-Homo_sapiens.gmt")
mysignature_name <- "Lung fibrosis%WikiPathways_20190210%WP3624%Homo sapiens"
library(pheatmap)
sig_heatmap(vst_airway,
  mysignatures[[mysignature_name]],
  res_data = res_airway,
  de_only = TRUE,
  annovec = annovec,
  title = mysignature_name,
  cluster_cols = TRUE
)
## Description

Combine data from a typical DESeq2 run

## Usage

```r
wrapup_for_iSEE(dds, res)
```

## Arguments

- `dds` A `DESeqDataSet` object.
- `res` A `DESeqResults` object.

## Details

Combines the DESeqDataSet input and DESeqResults into a SummarizedExperiment object, which can be readily explored with iSEE.

A typical usage would be after running the DESeq2 pipeline as specified in one of the workflows which include this package, e.g. in the context of the ideal package.

## Value

A SummarizedExperiment object, with raw counts, normalized counts, and variance-stabilizing transformed counts in the assay slots; and with colData and rowData extracted from the corresponding input parameters

## Examples

```r
# with simulated data...
library(DESeq2)
dds <- DESeq2::makeExampleDESeqDataSet(n = 10000, m = 8)
dds <- DESeq(dds)
res <- results(dds)
se <- wrapup_for_iSEE(dds, res)

# library(iSEE)
# iSEE(se)
```

```r
## Not run:

# or with the well known airway package...
library(airway)
data(airway)
airway
.dds_airway <- DESeq2::DESeqDataSetFromMatrix(assay(airway),
```
colData = colData(airway),
design = ~ cell + dex

dds_airway <- DESeq2::DESeq(dds_airway)
res_airway <- DESeq2::results(dds_airway)
se_airway <- wrapup_for_iSEE(dds_airway, res_airway)
# iSEE(se_airway)

## End(Not run)
Index

DESeqDataSet, 4, 6, 15
deseqresult2DEgenes, 2
deseqresult2tbl, 3
DESeqResults, 2, 3, 7, 9, 11, 13, 15
DESeqTransform, 13

ggplotCounts, 3
goseq, 5
goseqTable, 5

ideal, 6
ideal-pkg, 8

plot_ma, 8
plot_volcano, 10
plotCounts, 4

read_gmt, 7, 12

sepguesser, 12
sig_heatmap, 13

wrapup_for_iSEE, 15