Package 'FoldGO'

April 12, 2022

Type Package

Title Package for Fold-specific GO Terms Recognition

Version 1.12.0

Author Daniil Wiebe <daniil.wiebe@gmail.com> [aut, cre]

Maintainer Daniil Wiebe <daniil.wiebe@gmail.com>

Description FoldGO is a package designed to annotate gene sets derived from expression experiments and identify fold-change-specific GO terms.

Depends R (>= 4.0)

License GPL-3

Encoding UTF-8

LazyData true

biocViews DifferentialExpression, GeneExpression, GO, Software

RoxygenNote 6.1.0

Suggests knitr, rmarkdown, devtools, kableExtra

VignetteBuilder knitr

Imports topGO (>= 2.30.1), ggplot2 (>= 2.2.1), tidyr (>= 0.8.0), stats, methods

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

git_branch RELEASE_3_14

git_last_commit 7f8ce76

git_last_commit_date 2021-10-26

Date/Publication 2022-04-12

R topics documented:

`examdata_bg` *Background sets of genes used in examples*

Description

We used genes from two datasets in examples:

1. RNA-seq experiment on auxin treatment of Arabidopsis thaliana roots (degenes)
2. RNA-seq experiment on mRNA differential expression in LNCaP cells expressing the wild-type androgen receptor (AR-WT) or the ligand-independent AR-V7 splice variant (degenes_hum)

Usage

```
bggenes

bggenes_hum
```

Format

A vector containing the GeneIDs with 18039 and 38238 for A. thaliana and H. sapiens correspondingly

Source

1. A. thaliana and auxin: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE97258
2. H. sapiens LNCap AR-V7: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE71334

Examples

```
# load background genes from A. thaliana RNA-seq experiment
data("bggenes")
```

examdata_degs *Differential expressed genes used in examples*

Description

We used two datasets in examples:

1. RNA-seq experiment on auxin treatment of Arabidopsis thaliana roots (degenes)
2. RNA-seq experiment on mRNA differential expression in LNCaP cells expressing the wild-type androgen receptor (AR-WT) or the ligand-independent AR-V7 splice variant (degenes_hum)

Usage

```
degenes

degenes_hum
```

Format

A dataframes with 4 variables and 789 and 2079 for A. thaliana and H. sapiens correspondingly, where colnames are:

GeneID Gene identifier

FC fold-change value

pval p-value

qval Benjamini-Yekutieli adjusted p-value

Source

1. A. thaliana and auxin: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE97258
2. H. sapiens LNCap AR-V7: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE71334

Examples

```
# load degenes from RNA-seq experiment on auxin treatment of Arabidopsis thaliana roots
data("degenes")
```

examdata_objs *Precompiled objects of GeneGroups, FuncAnnotGroups classes used in examples*

Description

up_groups object of GeneGroups class compiled from up-regulated genes from rna-seq experiment on auxin treatment of Arabidopsis thaliana roots

down_groups object of GeneGroups class compiled from down-regulated genes from rna-seq experiment on auxin treatment of Arabidopsis thaliana roots

up_annotobj object of FuncAnnotGroups class compiled from lists of up-regulated genes from rna-seq experiment on auxin treatment of Arabidopsis thaliana roots

up_annotobj object of FuncAnnotGroups class compiled from lists of down-regulated genes from rna-seq experiment on auxin treatment of Arabidopsis thaliana roots

Usage

```
up_groups

down_groups

up_annotobj

down_annotobj
```

Format

up_groups object of GeneGroups class

down_groups object of GeneGroups class

up_annotobj object of FuncAnnotGroupsTopGO class

up_annotobj object of FuncAnnotGroupsTopGO class

Examples

```
# load GeneGroups object with up-regulated genes from rna-seq experiment on auxin treatment
# of Arabidopsis thaliana roots
data("up_groups")
# load FuncAnnotGroups object compiled from lists of up-regulated genes
# from rna-seq experiment on auxin treatment of Arabidopsis thaliana roots
data("up_annotobj")
```

`fagroupstopgo_class` *S4 class for FuncAnnotGroupsTopGO object*

Description

This function conducts functional enrichment analysis for sets of genes generated by `GeneGroups` function.

Constructor

`FuncAnnotGroupsTopGO(groups,namespace,customAnnot,annot,bggenes,...)`, where:

`groups` - object of `GeneGroups` class

`namespace` - character string specifing GO namespace ("BP", "MF" or "CC")

`customAnnot` - Use if `mapping` argument is set to "custom". It can be an object generated by `GAFReader` or `readGAF` function from mgsa package or list which has GO term ids as keys and character vectors contain gene ids as values.

`annot` - from TopGO manual: These functions are used to compile a list of GO terms such that each element in the list is a character vector containing all the gene identifiers that are mapped to the respective GO term.

`bggenes` - vector contains background set of genes

... - other parameters:

`genesannot` - minimal number of genes annotated to a term in the annotation. 1 by default

`algorithm` - from TopGO manual: character string specifing which algorithm to use. The algorithms are shown by the topGO whichAlgorithms() function. "classic" by default

`statistic` - from TopGO manual: character string specifing which test to use. The statistical tests are shown by the topGO whichTests() function. "fisher" by default

`mapping` - from TopGO manual: character string specifieng the name of the Bioconductor package containing the gene mappings for a specific organism. For example: mapping = "org.Hs.eg.db". "custom" by default

`ID` - from TopGO manual: character string specifing the gene identifier to use. Currently only the following identifiers can be used: c("entrez", "genbank", "alias", "ensembl", "symbol", "genename", "unigene")

Accessors

In the code examples below `object` is an object of FuncAnnotGroupsTopGO class

`getResultList(object)` - returns list of functional annotation result tables

Examples

```
# read .gaf file (in this example gaf file with annotation for \emph{A.thaliana} is used)
library(topGO)
gaf_path <- system.file("extdata", "gene_association.tair.lzma",
                        package = "FoldGO", mustWork = TRUE)
# read gaf file and convert annoitation in the list format
# contains GO term id's as keys and Gene ID's as values
gaf <- GAFReader(file = gaf_path, geneid_col = 10)
# split DEG genes into quantiles
gene_groups <- GeneGroups(degenes, 2)
# run enrichment test
annotobj <- FuncAnnotGroupsTopGO(gene_groups,"BP", customAnnot = gaf,
                                 annot = topGO::annFUN.GO2genes,
                                 bggenes = bggenes, padjmethod = "BH",
                                 qitborder = 10, genesannot = 1)

# get results of functional enrichment analysis in a tabular form:
getResultList(annotobj)
```

---

foldspectest_class *FoldSpecTest S4 class*

Description

FoldSpecTest object calculates test on fold-specificity and stores all resulting data needed for further analysis. It takes object which is instance of subclass of AnnotGroups class (e.g. FuncAnnotGroupsTopGO class) as a minimal set of input parameters. For more details see Constructor section.

Constructor

`FoldSpecTest(annotgroups,fdrstep1,fdrstep2,padjmethod,fisher_alternative)`, where:

`annotgroups` - object of FuncAnnotGroups class

`fdrstep1` - FDR threshold for 1 step of fold-specificty recognition procedure

`fdrstep2` - FDR threshold for 2 step of fold-specificty recognition procedure

`padjmethod` - method for multiple testing correction (to see all possible methods print: p.adjust.methods) Benjamini-Hochberg by default

`fisher_alternative` - indicates the alternative hypothesis and must be one of "two.sided", "greater" or "less". You can specify just the initial letter. Only used in the 2 by 2 case.

Accessors

In the code examples below `object` is an object of FoldSpecTest class

`getFStable(object)` - returns dataframe with fold-change-specific terms and related data

`getNFStable(object)` - returns dataframe with not fold-change-specific terms and related data

`getResultTable(object)` - returns dataframe with both fold-change-specific and not fold-change-specific terms

`getWholeIntName(object)` - returns name of largest fold-change interval (DEGs interval)

Examples

```
# FoldSpecTest function requires only object of FuncAnnotGroups class as a
# minimal set of parameters. In the example up_annotobj is an object of FuncAnnotGroups class
# compiled from lists of up-regulated genes from rna-seq experiment on auxin treatment
# of Arabidopsis thaliana roots [FoldGO::up_annotobj].
FoldSpecTest(up_annotobj)

# FoldSpecTest function with custom parameters
fs_up <- FoldSpecTest(up_annotobj, fdrstep1 = 0.2, fdrstep2 = 0.01, padjmethod = "BY")

# get dataframe with fold-change-specific terms
getFStable(fs_up)

# get dataframe with not fold-change-specific terms
getNFStable(fs_up)

# get dataframe with both fold-change-specific and not fold-change-specific terms
getResultTable(fs_up)

# get name of largest fold-change interval (DEGs interval)
getWholeIntName(fs_up)
```

`gafreader_class` *S4 class for GAFReader object*

Description

Parser for annotation presented in GAF file format (.gaf). GAFReader function returns object which contains as a dataframe annotation as it presented in initial file. Via GAFReader accessor method one can retrieve annotations as list GO term id's as keys and Gene ID's as values and version of file (see Accessors section).

Constructor

`GAFReader(file = gaf_path,geneid_col = 10)`, where:

`file` - full path to annotation file

`geneid_col` - index of column with Gene ID (2 by default)

Accessors

In the code examples below `object` is an object of GAFReader class

`getVersion(object)` - returns version of GAF file

`getAnnotation(object)` - returns annotation from GAF file in form of GO ids - Gene ids list

Methods

In the code examples below `object` is an object of GAFReader class

`getAnnotation(object)` - Convert annotation to list contains GO term id's as keys and Gene ID's as values

Examples

```
# read .gaf file (in this example gaf file with annotation for \emph{A.thaliana} is used)
# object returned by \code{\link{GAFReader}} can be used by
# \code{\link{FuncAnnotGroupsTopGO}} function.
gaf_path <- system.file("extdata", "gene_association.tair.lzma",
                        package = "FoldGO", mustWork = TRUE)
gaf <- GAFReader(file = gaf_path, geneid_col = 10)
# get version of file
getVersion(gaf)
# get annoitation in the list format contains GO term id's as keys and Gene ID's as values
getAnnotation(gaf)
```

---

genegroups_class *S4 class for Gene Groups*

Description

This function splits gene list into quantiles and generates all unions of neighbouring quantiles. It takes dataframe with genes ID's and fold values, number of quantiles and logical variable which must set to TRUE if fold values are presented in logarithmic scale, otherwise it must be set to FALSE value (TRUE by default) as parameters.

Constructor

`GeneGroups(inputtable,quannumber,logfold)`, where:

`inputtable` - dataframe contains initial set of genes gene ID's in the first row and corresponding fold change values in the second row

`quannumber` - number of quantiles (e.g. 2,3,4...)

`logfold` - TRUE if fold values are presented in log scale, otherwise is FALSE

Accessors

In the code examples below `object` is an object of GeneGroups class

`getGroups(object)` - returns list of gene sets for each quatile and all combinations

`getWholeIntName(object)` - returns name of the interval containing all differentially expressed genes

`getQuanNumber(object)` - returns number of quantiles

`getIntNames(object)` - returns vector of intervals names

`getRegType(object)` - returns regulation type

Examples

```
# split initial gene set into quantiles
gene_groups <- GeneGroups(degenes, 6)
# get list of gene sets for each quatile and all combinations
getGroups(gene_groups)
# get name of the interval containing all differentially expressed genes
getWholeIntName(gene_groups)
# get number of quantiles
getQuanNumber(gene_groups)
# get vector of intervals names
getIntNames(gene_groups)
# get regulation type
getRegType(gene_groups)
```

---

getAnnotation *Get annotation derived from annotation file*

Description

This method allows to retrieve annotation from `MgsaSets` or `GAFReader` class object in form of list contains GO term id's as keys and Gene ID's as values

Usage

```
getAnnotation(object)

## S4 method for signature 'AnnotationReader'
getAnnotation(object)

## S4 method for signature 'list'
getAnnotation(object)

## S4 method for signature 'MgsaSets'
getAnnotation(object)

## S4 method for signature '`NULL`'
getAnnotation(object)
```

Arguments

`object`	- Object of `mgsa` package `MgsaSets` class or `FoldGO` `GAFReader` class

Value

list contains GO term id's as keys and Gene ID's as values

Examples

```
## Not run:
gaf_path <- system.file("extdata", "gene_association.tair.lzma",
                            package = "FoldGO", mustWork = TRUE)
gaf <- GAFReader(file = gaf_path, geneid_col = 10)
getAnnotation(gaf)

## End(Not run)
```

`getWholeIntName` *getWholeIntName S4 method*

Description

This method returns name of the interval containing all differentially expressed genes. It can be applied to objects of GeneGroups and FoldSpecTest classes

Arguments

`object`	Object of GeneGroups or FoldSpecTest class

See Also

`FoldSpecTest` `GeneGroups`

Examples

```
# GeneGroups class object example
gene_groups <- GeneGroups(degenes, 6)
getWholeIntName(gene_groups)
# FoldSpecTest class object example
fs_up <- FoldSpecTest(up_annotobj)
getWholeIntName(fs_up)
```

`plot,FoldSpecTest,ANY-method`
Fold-change specific GO Profile chart plotting

Description

Fold-change specific GO Profile chart plotting

Usage

```
## S4 method for signature 'FoldSpecTest,ANY'
plot(x, y, x_text_size = 10)
```

Arguments

x	- object of S4 FoldSpecTest class with up-regulated genes
y	- object of S4 FoldSpecTest class with down-regulated genes
x_text_size	- x axis labels size

Value

- Fold-change specific GO Profile plot

Examples

```
# calculate fold-specificity test for up-regulated genes
up_fs <- FoldSpecTest(up_annotobj)
# calculate fold-specificity test for down-regulated genes
down_fs <- FoldSpecTest(down_annotobj)
plot(up_fs, down_fs)
```

---

rna_seq_data *Data from rna-seq experiment on auxin treatment of Arabidopsis thaliana roots*

Description

A dataset containing the GeneIDs and corresponding fold-change values.

Usage

```
rna_seq_data
```

Format

A data frame with 18039 rows and 4 variables:

GeneID Gene identifier

FC fold-change value

pval p-value

qval Benjamini-Yekutieli adjusted p-value

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE97258

Index