Package 'meshr'

March 9, 2015

Title Tools for conducting enrichment analysis of MeSH

Description A set of annotation maps describing the entire MeSH assembled using data from MeSH

Version 1.2.5

Author Itoshi Nikaido, Koki Tsuyuzaki, Gota Morota

Maintainer Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

Depends R (>= 3.0.1), fdrtool, Category, BiocGenerics, methods,cummeRbund, org.Hs.eg.db, MeSH.db, MeSH.AOR.db, MeSH.PCR.db,MeSHDbi, org.MeSH.Hsa.db, org.MeSH.

Imports

Suggests

License Artistic-2.0

biocViews AnnotationData, FunctionalAnnotation, Bioinformatics,Statistics, Annotation, MultipleComparisons

R topics documented:

meshr-package *Enrichment analysis for MeSH terms.*

Description

meshr package conducts a MeSH enrichment analysis employing gene-MeSH annotation data. A hypergeometric test accounting for a multiple tesing correction is used to find significantly enriched MeSH terms.

Details

Package:	meshr
Version:	1.2.4
Date:	1-12-2015
biocViews:	AnnotationData, FunctionalAnnotation, Bioinformatics, Statistics, Annotation, MultipleComparisons
Depends:	R (>= 3.0.1), cummeRbund, org.Hs.eg.db, fdrtool, Category, BiocGenerics, methods, MeSH.db, MeSH.AOE
Imports:	
Suggests:	
License:	Artistic-2.0

Index:

`meshHyperGTest performs a hypergeometric statistical test.`

Further information is available in the vignettes.

Author(s)

Gota Morota, Koki Tsuyuzaki, Takeru Nakazato, Itoshi Nikaido

Maintainer: Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

See Also

`MeSHHyperGParams-class`, `MeSHHyperGResult-class`, `meshHyperGTest`

Examples

```
ls("package:meshr")
```

---

category *A function to return the name of MeSH category*

Description

This function returns the name of MeSH category.

Usage

```
category(r)
category(r) <- value
```

Arguments

r	An object containing annotation information.
value	The annotation information to set on object.

Author(s)

Koki Tsuyuzaki

---

database *A function to return the name of MeSH database*

Description

This function returns the name of MeSH database.

Usage

```
database(r)
database(r) <- value
```

Arguments

r	An object containing annotation information.
value	The annotation information to set on object.

Author(s)

Koki Tsuyuzaki

---

geneid.cummeRbund *Test data of significant differentially expressed genes used in cummeRbund package.*

Description

This RNA-Seq data were taken from three samples, "iPS", "hESC", and "Fibroblasts". We first create two objects of gene sets, i.e., selected and universal genes, by comparing significantly regulated genes between iPS and hESC under the significance level of 0.05 by getSig method in **cummeRbund** package. 303 genes were finally choosed and 104 of them were differentially expressed.

Usage

```
data(geneid.cummeRbund)
```

Source

http://www.bioconductor.org/packages/release/bioc/vignettes/cummeRbund/inst/doc/cummeRbund-manual.pdf

See Also

sig.geneid.cummeRbund.

Examples

```
data(geneid.cummeRbund)
names(geneid.cummeRbund)

## This data is also available by following scripts.
if(interactive()){
library(cummeRbund)
library(org.Hs.eg.db)
cuff <- readCufflinks(dir = system.file("extdata", package = "cummeRbund"))

gene.symbols <- annotation(genes(cuff))[,4]
mySigGeneIds <- getSig(cuff,x='hESC',y='iPS',alpha=0.05,level='genes')
mySigGenes <- getGenes(cuff,mySigGeneIds)

sig.gene.symbols <- annotation(mySigGenes)[,4]
gene.symbols <- gene.symbols[!is.na(gene.symbols)]
sig.gene.symbols <- sig.gene.symbols[!is.na(sig.gene.symbols)]

geneid.cummeRbund <- select(org.Hs.eg.db, keys=gene.symbols, keytype="SYMBOL", columns="ENTREZID")
sig.geneid.cummeRbund <- select(org.Hs.eg.db, keys=sig.gene.symbols, keytype="SYMBOL", columns="ENTREZID")

na.index1 <- which(is.na(geneid.cummeRbund[,2]))
for (i in na.index1){
s <- unlist(strsplit(as.character(geneid.cummeRbund[i,][1]), ","))[1]
sym <- get(s, org.Hs.egALIAS2EG)[1]
geneid.cummeRbund[i,2] <- as.integer(sym)
}

na.index2 <- which(is.na(sig.geneid.cummeRbund[,2]))
for (i in na.index2){
s <- unlist(strsplit(as.character(sig.geneid.cummeRbund[i,][1]), ","))[1]
sym <- get(s, org.Hs.egALIAS2EG)[1]
sig.geneid.cummeRbund[i,2] <- as.integer(sym)
}

geneid.cummeRbund <- geneid.cummeRbund[!duplicated(geneid.cummeRbund[,2]), ]
sig.geneid.cummeRbund <- sig.geneid.cummeRbund[!duplicated(sig.geneid.cummeRbund[,2]), ]
}
```

MeSHHyperGParams-class

Class "MeSHHyperGParams"

Description

A parameter class for representing all parameters needed for running the 'meshHyperGTest' method with one of the MeSH categories ("Anatomy", "Organisms", "Diseases", "Chemicals and Drugs", "Analytical, Diagnostic and Therapeutic Techniques and Equipment", "Psychiatry and Psychology", "Phenomena and Processes", "Disciplines and Occupations", "Anthropology, Education, Sociology and Social Phenomena", "Technology and Food and Beverages", "Humanities", "Information Science", "Persons", "Health Care", "Publication Type", "Geographical Locations").

Objects from the Class

Objects can be created by calls of the form `new("MeSHHyperGParams", ...)`.

Slots

`geneIds`: Object of class `"ANY"`: A vector of gene identifiers. Numeric and character vectors are probably the only things that make sense. These are the gene ids for the selected gene set.

`universeGeneIds`: Object of class `"ANY"`: A vector of gene ids in the same format as `geneIds` defining a subset of the gene ids on the chip that will be used as the universe for the hypergeometric calculation.

`annotation`: A string giving the name of the gene-MeSH annotation package like MeSH.XXX.eg.db.

`category`: A string giving the name of the MeSH category like A, B, C, D, ...and so on.

`database`: A string giving the name of the MeSH database like gendoo, gene2pubmed, ...and so on.

`pvalueCutoff`: A numeric values between zero and one used as a p-value or FDR cutoff for hypergeometric test depending on `pAdjust`. The default is set to 0.05.

`pAdjust`: A string which can be one of the Benjamini-Hochberg procedure (a.k.a. q-value) ("BH"), Q-value ("QV"), empirical Bayes method ("lFDR"), and unadjusted p-value ("none") for multiple testing correction.

Methods

`geneIds(p)`, `geneIds(p) <- value` Accessor methods for the geneIds.

`universeGeneIds(p)`, `universeGeneIds(p) <- value` Accessor methods for the geneIds.

`annotation(p)`, `annotation(p) <- value` Accessor methods for the gene-MeSH annotation data.

`pAdjust(p)` An accessor method for the choice of a method for multiple testing correction.

`pvalueCutoff(p)` An accessor method for the choice of a threshold when conducting enrichment analysis.

Author(s)

Gota Morota, Koki Tsuyuzaki, Takeru Nakazato, Itoshi Nikaido

Maintainer: Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

See Also

`meshr-package`, `MeSHHyperGResult-class`, `meshHyperGTest`, `category`, `database`

MeSHHyperGResult-class

Class "MeSHHyperGResult"

Description

This class represents the results of a test for overrepresentation of MeSH terms among genes in a selected gene set based upon the Hypergeometric distribution.

For details on extracting information from this object, please read the documentation in the MeSHHyperGParams-class.

Objects from the Class

Objects can be created by calls of the form `new("MeSHHyperGResult", ...)`.

Slots

`meshCategory`: Object of class `"character"` representing the category of MeSH terms tested.

`meshAnnotation`: Object of class `"character"`. The name of the annotation data used in the analysis.

`meshDatabase`: Object of class `"character"`. The name of the database used in the analysis.

`ORA`: Object of class `"data.frame"`. MeSH IDs, MeSH Terms, P-value, and other statistics is returned.

Methods

meshCategory `signature(r = "MeSHHyperGResult")`: Returns the MeSH category used in the analysis.

meshAnnotation `signature(r = "MeSHHyperGResult")`: Returns the name of the annotation data used in the analysis.

meshDatabase `signature(r = "MeSHHyperGResult")`: Returns the name of the database used in the analysis.

meshIds `signature(r = "MeSHHyperGResult")`: Returns the character vector of the MeSH IDs identified as significant in the analysis.

meshTerms `signature(r = "MeSHHyperGResult")`: Returns the character vector of the MeSH terms identified as significant in the analysis.

pvalues `signature(r = "MeSHHyperGResult")`: Returns the associated p-values of significantly enriched MeSH terms.

summary `signature(r = "MeSHHyperGResult")`: Returns a `data.frame` summarizing the test result. Optional arguments `pvalue` and `categorySize` allow specification of maximum p-value and minimum categorySize, respectively. Optional argument `htmlLinks` is a logical value indicating whether to add HTML links (useful in conjunction with xtables print method with `type` set to `"html"`).

show `signature(object = "MeSHHyperGResult")`: Return a short description of the result.

save.pdf `signature(object = "MeSHHyperGResult")`: Return PDF files corresponding PMCID. This function is available only when using gene2pubmed as database

Author(s)

Gota Morota, Koki Tsuyuzaki, Takeru Nakazato, Itoshi Nikaido

Maintainer: Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

See Also

meshr-package, MeSHHyperGParams-class, meshHyperGTest

---

meshHyperGTest *Hypergeometric Tests for MeSH term association*

Description

Given a MeSHHyperGParams object containing a set of selected and background gene IDs, and gene-MeSH annotation data of interest, meshHyperGTest performs Hypergeomtric test for over-representation of each MeSH term accouting for the multiple testing correction.

Arguments

p	A MeSHHyperGParams object

Details

For details on creating MeSHHyperGParams object, please read the documentation in the MeSHHyperGParams-class.

Value

A MeSHHyperGResult object.

Author(s)

Gota Morota, Koki Tsuyuzaki, Takeru Nakazato, Itoshi Nikaido

Maintainer: Koki Tsuyuzaki <k.t.the-answer@hotmail.co.jp>

See Also

meshr-package, MeSHHyperGParams-class, MeSHHyperGResult-class

Examples

```
data(geneid.cummeRbund)
data(sig.geneid.cummeRbund)

meshParams <- new("MeSHHyperGParams", geneIds=sig.geneid.cummeRbund[,2], universeGeneIds=geneid.cummeRbund[

meshR <- meshHyperGTest(meshParams)
```

PMCID *PUBMEDID - PMCID correspondence*

Description

PUBMEDID - PMCID correspondence. This data is used by save.pdf function

Usage

```
data(PMCID)
```

Examples

```
data(PMCID)
names(PMCID)
```

sig.geneid.cummeRbund *Test data of significant differentially expressed genes used in cummeRbund package.*

Description

This RNA-Seq data were taken from three samples, "iPS", "hESC", and "Fibroblasts". We first create two objects of gene sets, i.e., selected and universal genes, by comparing significantly regulated genes between iPS and hESC under the significance level of 0.05 by getSig method in **cummeRbund** package. 303 genes were finally choosed and 104 of them were differentially expressed.

Usage

```
data(sig.geneid.cummeRbund)
```

Source

http://www.bioconductor.org/packages/release/bioc/vignettes/cummeRbund/inst/doc/cummeRbund-manual.pdf

See Also

geneid.cummeRbund.

Examples

```
data(sig.geneid.cummeRbund)
names(sig.geneid.cummeRbund)

## This data is also available by following scripts.
if(interactive()){
library(cummeRbund)
library(org.Hs.eg.db)
cuff <- readCufflinks(dir = system.file("extdata", package = "cummeRbund"))
```

```
gene.symbols <- annotation(genes(cuff))[,4]
mySigGeneIds <- getSig(cuff,x='hESC',y='iPS',alpha=0.05,level='genes')
mySigGenes <- getGenes(cuff,mySigGeneIds)

sig.gene.symbols <- annotation(mySigGenes)[,4]
gene.symbols <- gene.symbols[!is.na(gene.symbols)]
sig.gene.symbols <- sig.gene.symbols[!is.na(sig.gene.symbols)]

geneid.cummeRbund <- select(org.Hs.eg.db, keys=gene.symbols, keytype="SYMBOL", columns="ENTREZID")
sig.geneid.cummeRbund <- select(org.Hs.eg.db, keys=sig.gene.symbols, keytype="SYMBOL", columns="ENTREZID")

na.index1 <- which(is.na(geneid.cummeRbund[,2]))
for (i in na.index1){
s <- unlist(strsplit(as.character(geneid.cummeRbund[i,][1]), ","))[1]
sym <- get(s, org.Hs.egALIAS2EG)[1]
geneid.cummeRbund[i,2] <- as.integer(sym)
}

na.index2 <- which(is.na(sig.geneid.cummeRbund[,2]))
for (i in na.index2){
s <- unlist(strsplit(as.character(sig.geneid.cummeRbund[i,][1]), ","))[1]
sym <- get(s, org.Hs.egALIAS2EG)[1]
sig.geneid.cummeRbund[i,2] <- as.integer(sym)
}

geneid.cummeRbund <- geneid.cummeRbund[!duplicated(geneid.cummeRbund[,2]), ]
sig.geneid.cummeRbund <- sig.geneid.cummeRbund[!duplicated(sig.geneid.cummeRbund[,2]), ]
}
```

Index