Package ‘GeneAnswers’
April 25, 2017

Type Package
Title Integrated Interpretation of Genes
Version 2.18.0
Date 2014-10-1

Depends R (>= 3.0.0), igraph, RCurl, annotate, Biobase (>= 1.12.0),
   methods, XML, RSQLite, MASS, Heatplus, RColorBrewer
Imports RBGL, annotate, downloader
Suggests GO.db, KEGG.db, reactome.db, biomaRt, AnnotationDbi,
   org.Hs.eg.db, org.Rn.eg.db, org.Mm.eg.db, org.Dm.eg.db, graph

Author Lei Huang, Gang Feng, Pan Du, Tian Xia, Xishu Wang, Jing, Wen, Warren Kibbe and Simon Lin
Maintainer Lei Huang <lhuang7@uchicago.edu> and Gang Feng <gilbertfeng@gmail.com>

Description GeneAnswers provides an integrated tool for biological or medical interpreta-
   tion of the given one or more groups of genes by means of statistical test.

License LGPL (>= 2)

biocViews Infrastructure, DataRepresentation, Visualization,
   GraphsAndNetworks

LazyLoad yes

NeedsCompilation no

R topics documented:

GeneAnswers-package .............................................. 3
buildNet ................................................................. 4
caBIO2entrez ............................................................ 6
categoryNet ............................................................... 7
chartPlots ................................................................. 8
DmIALite ................................................................. 9
DO ......................................................................... 9
DOLite ................................................................. 10
DOLiteTerm ............................................................. 10
drawTable ............................................................... 11
entrez2caBIO .......................................................... 12
geneAnnotationHeatmap ............................................. 13
GeneAnswers-class ................................................... 14
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>geneAnswersBuilder</td>
</tr>
<tr>
<td>geneAnswersChartPlots</td>
</tr>
<tr>
<td>geneAnswersConceptNet</td>
</tr>
<tr>
<td>geneAnswersConceptRelation</td>
</tr>
<tr>
<td>geneAnswersConcepts</td>
</tr>
<tr>
<td>geneAnswersHeatmap</td>
</tr>
<tr>
<td>geneAnswersHomoMapping</td>
</tr>
<tr>
<td>geneAnswersReadable</td>
</tr>
<tr>
<td>geneAnswersSort</td>
</tr>
<tr>
<td>geneConceptNet</td>
</tr>
<tr>
<td>getcaBIOPATHList</td>
</tr>
<tr>
<td>getcaBIOPATHTerms</td>
</tr>
<tr>
<td>getCategoryList</td>
</tr>
<tr>
<td>getCategoryTerms</td>
</tr>
<tr>
<td>getConceptTable</td>
</tr>
<tr>
<td>getConnectedGraph</td>
</tr>
<tr>
<td>getDOLiteTerms</td>
</tr>
<tr>
<td>getGOList</td>
</tr>
<tr>
<td>getHomoGeneIDs</td>
</tr>
<tr>
<td>getListGIF</td>
</tr>
<tr>
<td>getMultiLayerGraphIDs</td>
</tr>
<tr>
<td>getNextGOIDs</td>
</tr>
<tr>
<td>getPathList</td>
</tr>
<tr>
<td>getPathTerms</td>
</tr>
<tr>
<td>getREACTOME.PATHList</td>
</tr>
<tr>
<td>getREACTOME.PATHTerms</td>
</tr>
<tr>
<td>getSingleLayerGraphIDs</td>
</tr>
<tr>
<td>getSymbols</td>
</tr>
<tr>
<td>getTotalGeneNumber</td>
</tr>
<tr>
<td>groupReport</td>
</tr>
<tr>
<td>HsIALite</td>
</tr>
<tr>
<td>humanExpr</td>
</tr>
<tr>
<td>humanGeneInput</td>
</tr>
<tr>
<td>MmIALite</td>
</tr>
<tr>
<td>mouseExpr</td>
</tr>
<tr>
<td>mouseGeneInput</td>
</tr>
<tr>
<td>RnIALite</td>
</tr>
<tr>
<td>sampleGroupsData</td>
</tr>
<tr>
<td>searchEntrez</td>
</tr>
<tr>
<td>topcaBIO.PATH</td>
</tr>
<tr>
<td>topcaBIO.PATHGenes</td>
</tr>
<tr>
<td>topCategory</td>
</tr>
<tr>
<td>topCategoryGenes</td>
</tr>
<tr>
<td>topDOLITE</td>
</tr>
<tr>
<td>topDOLITEGenes</td>
</tr>
<tr>
<td>topGO</td>
</tr>
<tr>
<td>topGOGenes</td>
</tr>
<tr>
<td>topPATH</td>
</tr>
<tr>
<td>topPATHGenes</td>
</tr>
<tr>
<td>topREACTOME.PATH</td>
</tr>
<tr>
<td>topREACTOME.PATHGenes</td>
</tr>
</tbody>
</table>
**Description**

GeneAnswers provide an integrated tool for biological or medical interpretation of the given one or more groups of genes by means of statistical test.

**Details**

<table>
<thead>
<tr>
<th>Package: GeneAnswers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type: Package</td>
</tr>
<tr>
<td>Version: 1.6.0</td>
</tr>
<tr>
<td>Date: 2010-10-14</td>
</tr>
<tr>
<td>License: LGPL version 2 or newer</td>
</tr>
</tbody>
</table>

**Author(s)**

Gang Feng, Lei Huang, Pan Du, Tian Xia, Xishu Wang, Jing, Wen, Warren Kibbe and Simon Lin

Maintainer: Gang Feng <g-feng@northwestern.edu> and Lei Huang <lhuang7@uchicago.edu>

**References**


**Examples**

```r
data('humanExpr')
data('humanGeneInput')
x <- geneAnswersBuilder(humanGeneInput, 'org.Hs.eg.db', categoryType='GO.BP', testType='hyperG', pvalueT=0.01, FDR.correct=TRUE, geneExpressionProfile=humanExpr)
class(x)
```
buildNet | build and display a network for given IDs and interaction Matrix

Description

A function to build and display a network for given IDs and interaction Matrix with specified filtered IDs.

Usage

```r
buildNet(graphIDs, idType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction', 'Customized'), edgeM=NULL, layers=1, filterGraphIDs=NULL, filterLayer=0, annLib=c('org.Hs.eg.db', 'org.Mm.eg.db', 'org.Rn.eg.db', 'org.Dm.eg.db', 'customized'), output=c('interactive', 'fixed'), netMode=c('layer', 'connection'), vertexSize=NULL, edgeColor=NULL, colorMap=NULL, zeroColorIndex=NULL, matchMode=c('absolute', 'relative'), label=TRUE, directed=FALSE, direction=c('up', 'down', 'both'), showModeForNodes=c('nodes', 'filters'), verbose=TRUE, readable=TRUE, labelSize=1, labelColor='#666666', ...)
```

Arguments

- `graphIDs`: a character vector for given IDs
- `idType`: type of IDs, could be one of 'GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction' and 'Customized'
- `edgeM`: a 2-column Matrix representing a network
- `layers`: an integer, specify how many layers will be retrieved.
- `filterGraphIDs`: a character vector for filtered IDs or a 2- or 3-column matrix for extra values.
- `filterLayer`: an integer, specify where filterGraphIDs are applied.
- `annLib`: type for annotation library, 'org.Hs.eg.db', 'org.Mm.eg.db', 'org.Rn.eg.db', 'org.Dm.eg.db' and 'customized'. For 'customized', edgeM is necessary
- `output`: type to specify output figure types
- `netMode`: type to show network, see details
- `vertexSize`: an integer, the size of vertices in the network, default is NULL
- `edgeColor`: a R compatible color type, the color of edges in the network, default is NULL
- `colorMap`: a R compatible color character vector, or NULL by embedded color scheme.
- `zeroColorIndex`: index of color corresponding to zero, see details
- `matchMode`: the mode of values matching colors, valid only if inputValue is not NULL, see details
- `label`: logic, specify whether put labels for non-given nodes in the network.
- `directed`: logic, the network is a directed or not
- `direction`: search direction, it could be 'up', 'down' and 'both'. Valid for directed network only.
- `showModeForNodes`: type, the show mode for nodes on the network, only valid if filterGraphIDs is not NULL, see details
- `verbose`: logic, specify to show information or not.
- `readable`: logic, specify whether show IDs or Terms/Names for nodes
- `labelSize`: an integer, the size of label for nodes
- `labelColor`: an R compatible color, default is #666666
- `...`: other parameters used by 'getCategoryTerms'
Details

Currently, if idType is 'GO', 'GO.BP', 'GO.CC' or 'GO.MF', edgeM will be ignore. edgeM is a 2-column matrix. For directional connection, the direction is from column 1 elements to column 2 elements. For non-directional connection, each connection should be reversely presented twice, one is from column 1 element to column 2 element, while another is from column 2 element to column 1 element. In other words, non-directional connection is considered as two reverse directional connections.

filterGraphIDs are applied only at the filterLayer and more outer layers. This means the nodes between the filterLayer layer and the most external layer belong to the filterGraphIDs. The nodes between given graphIDs and the (filterLayer-1) layer are or are not from filterGraphIDs, but those nodes not in filterGraphIDs should be able to be finally connected by given graphIDs and filter-GraphIDs.

There are two type of color matching methods. 'absolute' means, given zeroColorIndex that is color index in the colorMap for value 0, any value more than 0 will be matched to color between zeroColorIndex and the last one in colorMap based on the ratio of the value to the maximum of the inputValue, while the value less than 0 will be matched to color between the first color in colorMap and zeroColorIndex, also based on the ratio of the value to the minimum of the inputValue.

showModeForNodes stands for, if the filterGraphIDs is not NULL, some or all of filterGraphIDs could be nodes for given IDs multiple search. If it is set to 'nodes', it means only the values of nodes in the display network will be used to match color by matchMode. For 'filters', it means the values of all filter nodes will be used to match color. If values for color of nodes in the network are not large, while the maximum of color of filter nodes is large, it is recommended to set to 'nodes', or it is difficult to see difference for the nodes. For comparing two networks, for example, one is up-search and another is down-search for the same IDs, it is better to set to 'absolute' for easy comparisons.

There are two types of output figures. "Fixed" means a network will be drawn on a regular R canvas, while "interactive" will generate a tck/tk canvas. Users can adjust nodes on it by mouse.

If the filterGraphIDs is a ID vector. The filterGraphIDs nodes will be black, others will be white. If filterGraphIDs is a 2- or 3-column matrix, the 1st column is filter IDs and 2nd column is for color of nodes. If the 3rd column is available, it is for size of nodes.

There are two types of netMode. 'layer' means size of nodes will be smaller and smaller for more and more external layers. And also color of edges change for different layers. 'connection' mode just distinguish direct or indirect connection. The size of the given IDs the largest. However, if filterGraphIDs is a 3-column matrix, the size of nodes will be determined by the 3rd column of filterGraphIDs.

The graphIDs nodes are yellow circled solid dots. Color depends on colorMap and filterGraphIDs 2nd column. If no value available, all given graphIDs filterGraphIDs nodes are black, others are white.

Value

invisibly return a list containing elements to represent a network.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

See Also

generateCategoryTerms

Examples

```r
require(GeneAnswers)
example(GeneAnswers)
filterM <- cbind(rownames(getEnrichmentInfo(x)), -log2(getEnrichmentInfo(x)[,7]), getEnrichmentInfo(x)[,1])
## Not run: buildNet(rownames(getEnrichmentInfo(x))[6:9], layers=5, filterGraphIDs=filterM, filterLayer=3, directed=TRUE, output=
## Not run: buildNet(rownames(getEnrichmentInfo(x))[200:204], layers=2, filterGraphIDs=filterM, filterLayer=1, directed=TRUE, output=
## Not run: buildNet(rownames(getEnrichmentInfo(x))[6:9], layers=3, filterGraphIDs=filterM[,1:2], filterLayer=3)
```

---

cabIO2entrez

map cabIO gene IDs to Entrez gene IDs

Description

Function to map the given cabIO gene IDs to the Entrez gene IDs. This function is not supported starting this version due to the termination of the cabBig project.

Usage

```r
cabIO2entrez(cabIOIds)
```

Arguments

- `cabIOIds` an cabIOIds gene IDs vector

Value

return a Entrez genes ID list, names of the list are the given cabIO gene IDs and elements are Entrez gene IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

```r
## Not run: cabIO2entrez(c('2933', '7326'))
```
categoryNet

Plot Category Links

Description

Function to plot a linkages of specified categories.

Usage

categoryNet(catGenesList, centroidSize=NULL, output=c('fixed','interactive'))

Arguments

catGenesList  a list of categories.
centroidSize  a numeric vector to specify the size of concept nodes. If NULL, all of concept nodes are represented as the same size solid circles.
output  type to specify output figure types.

Details

catGenesList is a list of categories. Each element contains the genes in the corresponding category, respectively. And the names of the list are categories. If centroidSize is a numeric vector, its values are mapped to the categories in the catGenesList sequentially.

Value

A category linkage is generated.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

help

Examples

input <- list('cat1'=c(1,4,2,5), 'cat2'=c(3,5,8,9), 'cat3'=c(2,4,5,9), 'cat4'=c(1,5,3))
## Not run: categoryNet(input)
Description

Make pie chart and bar plot based on the given data frame.

Usage

chartPlots(x, chartType = c("pieChart", "barPlot", "all"), specifiedCols = c("genes in Category"),

Arguments

x an data frame to be used for pie chart and box plot
chartType plot type, "pieChart", "barPlot" or both could be specified.
specifiedCols the column will be used to be represented.
top number to specify how many first categories will be drawn.
newWindow logic, determine whether draw on a new canvas.
... additional arguments passed to piechart or barplot.

Details

chartType could be pie chart, bar plot or both (parameter is "all"). specifiedCols is the column that will be used to plot. It could be column name or number. If chartType is set to 'all', the barplot will be drawn on a new canvas whatever newWindow is set to TRUE or FALSE.

Value

A pie chart and/or barplot are generated depends on specification.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

x <- matrix(c(6,9,3,30,13,2,15,20), nrow = 4, ncol=2, byrow=FALSE,
dimnames = list(c("group1", "group2", "group3", "group4"),
c("value1", "value2")))
chartPlots(x, chartType='all', specifiedCol = "value2", top = 3)
**DmIALite**

*Fly gene interaction matrix*

**Description**
Preprocessed fly gene interaction matrix

**Usage**
```
data(DmIALite)
```

**Details**
a 4-column matrix containing fly interacted genes and evidences

**References**

**Examples**
```
data(DmIALite)
DmIALite[1:4,]
```

---

**DO**

*Several data objects related with DO (Disease Ontology) and its mapping to genes*

**Description**
Several data objects related with DO (Disease Ontology) and its mapping to genes

**Usage**
```
data(Do)
```

**Details**
The data file "DO.rda" includes five datasets:
- **DO.graph.gene**: a graphNEL object, which shows the ontology relations of DO
- **DO.graph.closure.gene**: a graphNEL object, whose edges represent the link between a DO term and its offspring ontology terms. Only the DO terms with gene mappings were included.
- **DO2gene.map**: a list show the mapping from DOIDs to genes
- **gene2DO.map**: a list show the mapping from genes to DOIDs
- **DO.terms**: a named character vector. Its names are DOIDs and elements are DO.terms
Examples

```r
data(DO)

datasets <- c("DO.graph.gene", "DO.graph.closure.gene", "DO2gene.map", "gene2DO.map", "DO.terms")
# check the existence of these datasets:
sapply(datasets, exists)
```

---

**DOLite**

*Disease Ontology Annotation List*

---

**Description**

Disease Ontology Annotation List

**Usage**

data(DOLite)

**Details**

a standard list, whose names are DOLite IDs and each element contains the gene Entrez IDs belonging to the corresponding DOLite IDs.

**Source**

~~ reference to a publication or URL from which the data were obtained ~~

**References**


**Examples**

data(DOLite)
DOLite[1:2]

---

**DOLiteTerm**

*Disease Ontology Annotation Vector*

---

**Description**

Disease Ontology Annotation Vector

**Usage**

data(DOLiteTerm)
**drawTable**

**Details**

a character vector, where names are DOLite IDs and elements are Terms

**Source**

~~ reference to a publication or URL from which the data were obtained ~~

**References**


**Examples**

data(DOLiteTerm)
DOLiteTerm[1:10]

drawTable

---

**Concept-Gene Networking Plotting**

**Description**

A function to generate a multigroup concepts-genes table

**Usage**

drawTable(dataMatrix, topCat=10, heatMap=TRUE, matrixOfHeatmap=NULL, clusterTable=c('geneNum', 'p'), addRowLabel=TRUE, cex.axis=c(1.1, 0.9), reverseOfCluster=FALSE, xGridLine=FALSE, colorBar=TRUE, newWindow=TRUE, endOfColBar=c('> 0.01', 'Minimum of p values'), heatMapColor=c('#00ff00', '#ffffff'), canvasWidth=NULL, canvasHeight=NULL, ...)

**Arguments**

- `dataMatrix`: a top concepts-genes matrix generated by `getConceptTable`
- `topCat`: number to specify how many top concepts-genes analysis will show.
- `heatMap`: logic, determine whether the multiple group concepts-genes table is presented by heatmap.
- `matrixOfHeatmap`: NULL or a concepts-genes matrix generated by `getConceptTable`, which is used to show enrichment test significance for each concept.
- `clusterTable`: cluster data to specify which type of values will be used for cluster.
- `methodOfCluster`: cluster method
- `mar`: marginal parameter for table, please see `par`
- `addRowLabel`: logic, whether add row names
- `cex.axis`: font size parameter for table, please see `par`
- `reverseOfCluster`: logic, whether reverse the cluster order.
- `xGridLine`: logic, whether add horizontal line in table or not
Function to map the given Entrez gene IDs to the caBIO gene IDs. This function is not supported starting this version due to the termination of the caBig project.

Usage

```
entrez2caBIO(lls)
```
geneAnnotationHeatmap

Arguments

- **lls**: an Entrez gene IDs vector

Value

- return a caBIO genes ID list, names of the list are Entrez gene IDs and elements are caBIO gene IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

```r
## Not run: entrez2caBIO(c('1647', '596'))
```

---

**geneAnnotationHeatmap**  
*Make a concept-gene cross tabulation*

Description

Function to make a concept-gene cross tabulation

Usage

```r
geneAnnotationHeatmap(annotationList, dataMatrix = NULL, addGeneLabel = TRUE, colorMap = c("#000000", "#FFFFFF"), sortBy = c(0.8, 0.8), mapType = c("table", "heatmap"), displayAll=FALSE, symmetry=FALSE, colorBar=FALSE, colorBarLabel=NULL)
```

Arguments

- **annotationList**: a list of annotation to gene mapping.
- **dataMatrix**: a 2-dimensional numeric matrix. If it is provided, it will be plot side by side with the annotation heatmap.
- **addGeneLabel**: logic, indicate whether add gene labels
- **colorMap**: vector to specify color map of the two-color annotation heatmap
- **sortBy**: string to specify whether to sort the annotation matrix by row, column, both row and column or none of them
- **standardize.data**: logic, specify whether to standardize the dataMatrix by row~
- **colorMap.data**: string to specify color map of the dataMatrix heatmap
- **showGeneMax**: an integer, the maximum of gene number to show genes id or symbol on the heatmap
- **sortBy.data**: string to specify whether to sort the dataMatrix by row, column, both row and column or none of them
- **mar**: integer vector to speicify margin of the plot
GeneAnswers-class

Details

This function basically generates two maps in one canvas. Left side is a heatmap based on given expression matrix. Right side is a concept-gene map, which could be represented as two-color heatmap or table, depends on parameter "mapType".

Value

The function will generate a map without return value.

Author(s)

Pan Du, Gang Feng and Simon Lin

References


Examples

```r
a <- list(group1 = c('a','b','c','d','f'), group2= c('b','d','e','a','g','h'))
b <- matrix(rnorm(48), nrow=8,ncol=6)
rownames(b) <- tolower(LETTERS[1:8])
colnames(b) <- c('ctrl1', 'ctrl2', 'ctrl3', 'treat1', 'treat2', 'treat3')
## Not run: geneAnnotationHeatmap(a,dataMatrix=b)
```

GeneAnswers-class

Class GeneAnswers: contain and describe the relationship between given gene data and specified category

Description

This is a class representation of the relationship between given gene data and specified category.

Creating Objects

Objects can be created using the function `geneAnswersBuilder`.
Slots

Slot specific to GeneAnswers:

geneInput: a data frame containing gene Entrez IDs with or without any values. Current version only supports gene Entrez IDs. The values could be foldChange, p value, or other values. These data can be used for concept-gene network. Genes with positive values will be represented as red nodes, while negative value genes are green nodes.
testType: statistical test method. Current version supports hypergeometric test to test relationship between genes and specified categories.
pvalueT: the cutoff value of statistical test. Any categories will not be reported if the p value is more than the cutoff.
genesInCategory: a list containing genes belonging to categories. The names of the list are categories.
geneExprProfile: a data frame to store gene expression data. If not available, it could be NULL.
anLib: annotation database used for statistical test.
categoryType: functional or medical category used for statistical test.
enrichmentInfo: a data frame containing filtered categories with statistical results by specified pvalueT.

Methods

Class-specific methods:

geneInput(GeneAnswers): Access the geneInput slot of GeneAnswers object.
testType(GeneAnswers): Access the testType slot of GeneAnswers object.
pvalueT(GeneAnswers): Access the pvalueT slot of GeneAnswers object.
genesInCategory(GeneAnswers): Access the genesInCategory slot of GeneAnswers object.
geneExprProfile(GeneAnswers): Access the geneExprProfile slot of GeneAnswers object.
anLib(GeneAnswers): Access the anLib slot of GeneAnswers object.
categoryType(GeneAnswers): Access the categoryType slot of GeneAnswers object.
enrichmentInfo(GeneAnswers): Access the enrichmentInfo slot of GeneAnswers object.

setGeneInput(GeneAnswers, geneInput): Assign the geneInput slot of GeneAnswers object.
setTestType(GeneAnswers, type=c('hyperG', 'none')): Assign the testType slot of GeneAnswers object.
setPValueT(GeneAnswers, pvalueT): Assign the pvalueT slot of GeneAnswers object.
setGenesInCategory(GeneAnswers, genesInCategory): Assign the genesInCategory slot of GeneAnswers object.
setGeneExprProfile(GeneAnswers, geneExprProfile): Assign the geneExprProfile slot of GeneAnswers object.
setAnLib(GeneAnswers, anLib): Assign the anLib slot of GeneAnswers object.
setCategoryType(GeneAnswers, type=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'DOLITE', 'KEGG', 'REACTOME', 'CABIO.PATH', 'User defiend')): Assign the categoryType slot of GeneAnswers object.
setEnrichmentInfo(GeneAnswers, enrichmentInfo): Assign the enrichmentInfo slot of GeneAnswers object.

summary(GeneAnswers): Briefly summarize the information of GeneAnswers object and show contents of GeneAnswers object.
show(GeneAnswers): Briefly show contents of GeneAnswers object.
Author(s)
Gang Feng, Pan Du and Simon Lin

References

See Also
geneAnswersBuilder

Examples
```r
data('humanExpr')
data('humanGeneInput')
x <- geneAnswersBuilder(humanGeneInput, 'org.Hs.eg.db', categoryType='GO.BP', testType='hyperG', pvalueT=0.01, FDR.correct=TRUE, geneExpressionProfile=humanExpr)
class(x)
```

geneAnswersBuilder Build an object of a GeneAnswers class

Description
A function to build an object of a GeneAnswers class based on given information.

Usage
geneAnswersBuilder(geneInput, annotationLib, categoryType = NULL, testType = c("hyperG", "none"), known=TRUE, ... geneExpressionProfile = NULL, categorySubsetIDs = NULL, pvalueT = 0.01, FDR.correction = FALSE, verbose=TRUE, sortBy=c("pvalue", "geneNum", "foldChange", "oddsRatio", "correctedPvalue", "none"), ...)

Arguments
geneInput a dataframe containing gene IDs and possible values associated with given gene IDs.
annotationLib name of given annotation library file or user provided annotation list.
categoryType name of given annotation category or NULL for user provided annotation list.
testType name of enrichment test.
known logic, specify only known annotation gene enrichment test.
totalGeneNumber number of total genes to perform hypergeometric test.
geneExpressionProfile data frame containing gene expression file or NULL.
categorySubsetIDs a character vector of user-specified subset of categories to be tested.
pvalueT p-value threshold of the enrichment test.
FDR.correction logic, indicating if FDR correction of the enrichment test p-value is performed or not.
verbose logic, display current building stage.
sortBy sorted type
... additional arguments passed to getGOList.
**geneAnswersBuilder**

**Details**

As the input of geneAnswersBuilder, geneInput could be a character vector (Gene Entrez ID vector), a matrix or a dataframe. For the matrix and dataframe, the first column is for Gene Entrez IDs, while other columns could be any interested values that could be used to represent gene expression direction for generating concepts-genes network. Rownames are not necessary.

annotationLib could be Disease Ontology library, Entrez annotation libraries for a specie, such as 'org.Hs.eg.db'. Current version supports "org.Ag.eg.db", "org.Bt.eg.db", "org.Ce.eg.db", "org.Cf.eg.db", "org.Dm.eg.db", "org.Dr.eg.db", "org.EcK12.eg.db", "org.EcStrainsakai.eg.db", "org.Gg.eg.db", "org.Hs.eg.db", "org.Mm.eg.db", "org.Mmu.eg.db", "org.Pt.eg.db", "org.Rn.eg.db", "org.Ss.eg.db", "org.XL.eg.db", "org.At.tair.db", "org.Pf.plasmo.db" and "org.Sc.sgd.db". User can also use own annotation library. User’s annotation library should be a list. Each element in this list is a vector of genes for a user-specified category. Names of this annotation list are categories’ names.

categoryType could be "GO", "GO.BP", "GO.CC", "GO.MF", "DOLITE", 'KEGG', "REACTOME.PATH" and "CABIO.PATH". "GO.BP" only test biological process Gene Ontology terms, "GO.CC" for cellular components, "GO.MF" for molecular functions, "GO" for all of these three categories, "KEGG" for all KEGG pathways, and "REACTOME.PATH" for all REACTOME pathways. For user provided annotation library, it should be NULL in most cases.

"caBIO.PATH", which was for NCI-Nature curated, Biocarta and Reactome, is not supported starting this version because of the termination of caBig project.

If known is set to TRUE, the enrichment test only considers the genes with annotation. If FALSE, the total number of genes in that species will be returned. If user has own annotationLib, totalGeneNumber should be an integer, or one of "anopheles", "arabidopsis", "bovine", "worm", "canine", "fly", "zebrafish", "ecoliStrainsakai", "chicken", "human", "mouse", "rhesus", "malaria", "chimp", "rat", "yeast", "pig" and "xenopus". NULL only works when "known" is set TRUE. geneAnswersBuilder will automatically assign the corresponding value to totalGeneNumber. User can get total gene numbers by `getTotalGeneNumber`, too.

sortBy could be one of "geneNum", "pvalue", "foldChange", "oddsRatio", "correctedPvalue" and "none". Default value is 'pvalue'.

**Value**

A GeneAnswers class containing geneInput, enrichmentInfo, etc.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**


**See Also**

`getTotalGeneNumber`

**Examples**

data('humanExpr')
data('humanGeneInput')
x <- geneAnswersBuilder(humanGeneInput, 'org.Hs.eg.db', categoryType='GO.BP', testType='h', pvalueT=0.1, FDR.correct=TRUE)
class(x)
geneAnswersChartPlots  
Make pie chart and bar plot

Description
Make pie chart and bar plot for given GeneAnswers instance

Usage
geneAnswersChartPlots(x, chartType=c("pieChart", "barPlot", "all"), sortBy = c('geneNum', 'pvalue'), newWindow=TRUE, ...)

Arguments
x  a GeneAnswers instance
chartType  plot type, "pieChart", "barPlot" or both could be specified.
sortBy  the column will be used to be represented.
newWindow  logic, determine whether draw on a new canvas.
...  additional arguments passed to piechart or barplot.

Details
chartType could be pie chart, bar plot or both (parameter is "all"). specifiedCols is the column of enrichmentInfo that will be used to plot. It could be one of 'genes in Category', 'p value' or 'fdr p value'. If chartType is set to 'all', the barplot will be drawn on a new canvas whatever newWindow is set to TRUE or FALSE.

Value
A pie chart and/or barplot are generated depends on specification.

Author(s)
Gang Feng, Pan Du and Simon Lin

References

See Also
chartPlots

Examples
example(GeneAnswers)
## Not run: geneAnswersChartPlots(x)
Description

A function to generate a concept-gene network by given gene information

Usage

geneAnswersConceptNet(x, colorValueColumn = NULL, centroidSize = c("pvalue", "geneNum", "foldChange", "oddsRatio", "correctedPvalue"), showCats = c(1:5), geneLayer = 1, edgeM = NULL, catTerm = FALSE, geneSymbol = FALSE, catID = FALSE, nameLength = "all", ...)

Arguments

x a GeneAnswers instance.
colorValueColumn number or column name of geneInput slot to specify the colors of leaves
centroidSize type to represent the size of concepts.
output output type of final output.
showCats a numeric or string vector specified categories
geneLayer an integer, specify how many layers of genes connecting to concepts
edgeM a 2-column Matrix representing a network
catTerm a logic value to specify whether mapping category IDs to category names
geneSymbol a logic value to specify whether mapping gene IDs to gene symbols
catID a logic value to specify whether show category IDs when catTerm is set to TRUE
nameLength show how many first letters for long term names, 'all' for full name
... other parameters used by 'geneConceptNet'

Details

colorValueColumn specifies which column of the geneInput of the GeneAnswers instance is used for color of nodes. centroidSize could be one of "geneNum", "pvalue", "foldChange", "oddsRatio", "correctedPvalue". Each one defines to which the size of concept dot is proportional geneNum: number of genes connecting to the concept pvalue: p value of enrichment test foldChange: fold of gene overrepresent in concepts oddsRatio: odds ratio of enrichment test correctedPvalue: adjusted p value of enrichment test output defines whether the final figure is interactive or not. Interactive figure calls igraph package to generate a tcl/tk canvas. Fixed figure is a non-interactive png figure. None will not output any figure but a list. See details in geneConceptNet.

Value

One concept-gene figure is generated. It could be a R figure or tcltk figure depends on how the user set parameter output.

Author(s)

Gang Feng, Pan Du and Simon Lin
geneAnswersConceptRelation

References


See Also

geneConceptNet, getMultiLayerGraphIDs

Examples

e.example(GeneAnswers)
## Not run: geneAnswersConceptNet(x, colorValueColumn='foldChange', centroidSize='pvalue', output='interactive')

geneAnswersConceptRelation

Display a network related to given concepts for a GeneAnswers instance

Description

A function to display a network related to given concepts of a GeneAnswer instance

Usage

geneAnswersConceptRelation(x, showCats=c(1:5), conceptsIDs=NULL, directed=TRUE, direction=c('up', 'down', 'both'), catTerm=TRUE, catID=FALSE, nameLength='all', ...)

Arguments

x a GeneAnswers instance
showCats a numeric or string vector specified categories
conceptsIDs a vector or a data frame or matrix containing possible relative concepts, see details
directed logic, the network is a directed or not
direction search direction, it could be 'up', 'down' and 'both'. Valid for directed network only.
catTerm a logic value to specify whether mapping category IDs to category names
catID a logic value to specify whether show category IDs when catTerm is set to TRUE
nameLength show how many first letters for long term names, 'all' for full name
... other parameters used by 'getConnectedGraph'

Details

categoriesIDs could be a character vector or a data frame or a matrix. As a character vector, it is a group of concept IDs or names depending on the given GeneAnswers instance, which are used to be a group of filters to draw a network relative to given concepts specified by showCats. When it is a data frame or matrix, it could be a 2- or 3-column data frame or matrix. The column 2 is always used to be represent nodes color, while the 3rd column is for size of nodes if available.
Value

return a invisible list representing the network.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

geneAnswersConcepts

tools

Examples

```r
require(GeneAnswers)
example(GeneAnswers)
## Not run: geneAnswersConceptRelation(x, UP=FALSE, directed=TRUE, netMode='connection')
```

geneAnswersConcepts Concept-Gene Networking Plotting

Description

A function to generate a concept-gene network by given gene information

Usage

geneAnswersConcepts(x, centroidSize=c('geneNum', 'pvalue', 'foldChange', 'oddsRatio', 'correctedPvalue'), output=c('fixed', 'interactive'), showCats=1:5, catTerm=FALSE, catID=FALSE)

Arguments

- **x**: a GeneAnswers instance.
- **centroidSize**: type to represent the size of concepts.
- **output**: output type of final output.
- **showCats**: a numeric or string vector specified categories.
- **catTerm**: a logic value to specify whether mapping category IDs to category names.
- **catID**: a logic value to specify whether show category IDs when catTerm is set to TRUE

Details

centroidSize could be one of "geneNum", "pvalue", "foldChange", "oddsRatio", "correctedPvalue". Each one defines to which the size of concept dot is proportional geneNum: number of genes connecting to the concept pvalue: p value of enrichment test foldChange: fold of gene overrepresent in concepts oddsRatio: odds ratio of enrichment test correctedPvalue: adjusted p value of enrichment test output defines whether the final figure is interactive or not. Interactive figure calls igraph package to generate a tcl/tk canvas. Fixed figure is a non-interactive png figure.
geneAnswersHeatmap

Description

A function to generate specified Concept-Gene Tabulates

Usage

geneAnswersHeatmap(x, showCats = c(1:5), catTerm = FALSE, geneSymbol = FALSE, catID=FALSE, nameLength='all', showAllGenes=FALSE, ...)

Arguments

x an instance of GeneAnswers objects
showCats a numeric or string vector specified categories
catTerm a logic value to specify whether mapping category IDs to category names
geneSymbol a logic value to specify whether mapping gene IDs to gene symbols
catID a logic value to specify whether show category IDs when catTerm is set to TRUE
nameLength show how many first letters for long term names, 'all' for full name
showAllGenes logic, show all genes in the heatmap or not
... other parameters used by geneAnnotationHeatmap

Details

This function generates concept-gene tabulates for an input GeneAnswers instance. The concept-gene tabulates contain two maps. Left side is a heatmap based on given expression matrix. Right side is a concept-gene map, which could be represented as two-color heatmap or table.
**geneAnswersHomoMapping**

**Value**

The function will generate a map without return value.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**


**See Also**

geneAnnotationHeatmap

**Examples**

```r
eample(GeneAnswers)
## Not run: geneAnswersHeatmap(x, catTerm=TRUE, geneSymbol=TRUE)
```

---

**geneAnswersHomoMapping**

Mapping homogenes for a GeneAnswers instance

**Description**

A function to mapping homogenes in all of slots of a GeneAnswer instance

**Usage**

```r
geneAnswersHomoMapping(x, species = c("human", "rat", "mouse", "fly"), speciesL = c("human", "rat")
```

**Arguments**

- `x` : a GeneAnswers instance
- `species` : species of the current genes
- `speciesL` : species of the mapped genes
- `mappingMethod` : mapping method, see details
- `filterGenes` : a gene symbol vector to filter genes
- `verbose` : logical, show current stage or not

**Details**

There are two mapping methods supported by current version. "direct" only works between human and mouse because most of human gene symbols are capitalized and only the first letter is uppercase for those homogenes in mouse. Another way is by means of package "biomaRt" ,which contains more information while the network connection is necessary to access biomaRt online server. Since two methods are based on different mechanisms, it is highly recommended to employ same method during mapping. Each method might introduce more homogenes, so users can remove ones that do not belong to original genes by optional "filterGeneList".


Value
return a mapped GeneAnswers instance

Author(s)
Gang Feng, Pan Du and Simon Lin

References

See Also
geneAnswersReadable

testHomoGeneIDs

Examples
example(GeneAnswers)
## Not run: geneAnswersHomoMapping(x, species='human', speciesL='mouse', mappingMethod='direct')

geneAnswersReadable Make GeneAnswers Instance readable

Description
a function to mapping category IDs and gene IDs to names and symbols.

Usage
geneAnswersReadable(x, catTerm = TRUE, geneSymbol = TRUE, strict = FALSE, verbose=TRUE, missing=c('name', 'keep', 'remove'), ...)

Arguments
x a GeneAnswers instance containing category IDs and geneIDs
catTerm logic value to determine whether mapping category IDs to names
geneSymbol logic value to determine whether mapping gene IDs to symbols
strict logic value to determine whether interrupt conversion if NA is introduced.
verbose logical, show current stage or not
missing type of handling NA mapping.
... other parameters used by getCategoryTerms

Details
Conversion could stop if NA is introduced and strict is set to TRUE. There are three types of parameters for variable 'missing'. 'name' means the NA mapping values are replaced by their names. 'keep' means all of NA values are kept. 'remove' means all of NA values are removed. Occasionally, Reactome uses the same name for species-mixed pathways based on in vivo and in vitro experiments, so we highly recommend to set addID as TRUE for Reactome test.
geneAnswersSort

Value
return a GeneAnswers instance with category names and/or gene symbols.

Author(s)
Gang Feng, Pan Du and Simon Lin

References

See Also
getSymbols, getCategoryTerms

Examples
e.example(GeneAnswers)
x.x <- geneAnswersReadable(x)

geneAnswersSort  Sort enrichmentInfo of a GeneAnswers instance

Description
a function to sort enrichmentInfo data frame in GeneAnswers objects.

Usage
geneAnswersSort(x, sortBy = c("geneNum", "pvalue", "foldChange", "oddsRatio", "correctedPvalue"))

Arguments
x  a GeneAnswers instance
sortBy  sorted type

Details
sortBy could be one of "geneNum", "pvalue", "foldChange", "oddsRatio" and "correctedPvalue".

Value
return a new GeneAnswers instance with sorted by the specified type.

Author(s)
Gang Feng, Pan Du and Simon Lin

References
See Also
GeneAnswers-class

Examples

```r
eexample(GeneAnswers)
xx <- geneAnswersSort(x, sortBy='correctedPvalue')
```

geneConceptNet Generate Concept-gene network

Description
Function to generate concept-gene network based on given list.

Usage

geneConceptNet(inputList, lengthOfRoots=NULL, inputValue = NULL, centroidSize = "geneNum", output = c("fixed", "interactive", "none"), colorMap=NULL, bgColor="#ffffff", matchMode=c(absolute, relative), zeroColorIndex=NULL, verbose=FALSE, symmetry=TRUE)

Arguments

inputList a character list to generate concept-gene network. Names of the list are concepts.
lengthOfRoots an integer, how many first elements could be root nodes.
inputValue NULL or a numeric vector to be used for color of nodes.
centroidSize 'geneNum' or a numeric vector to specify the size of concept nodes.
output type to specify output figure types
colorMap a R compatible color character vector, or NULL by embedded color scheme.
bgColor a R compatible color, default is '#ffffff' (white)
matchMode the mode of values matching colors, valid only if inputValue is not NULL, see details
zeroColorIndex index of color corresponding to zero, see details
verbose logic, determine whether show messages
symmetry logic, determine whether positive and negative values use the same color level.

Details

The color of gene nodes could be specified by inputValue. Its length should be same as the total number of unique genes in inputList. There are two type of color matching methods. 'absolute' means, given zeroColorIndex that is color index in the colorMap for value 0, any value more than 0 will be matched to color between zeroColorIndex and the last one in colorMap based on the ratio of the value to the maximum of the inputValue, while the value less than 0 will be matched to color between the first color in colorMap and zeroColorIndex, also based on the ratio of the value to the minimum of the inputValue. 'relative' means, set the first and last colors in colorMap to minimum and maximum of the inputValue, respectively, then any value between them will be mapped. If colorMap is set to NULL, the default color scheme will be applied. If the matching method is 'absolute', the color of 0 or the median of inputValue for 'relative' method, is set by bgColor, default value is '#ffffff' (white). The most positive value is represented as '#ff0000' (red), '#00ff00' (green) for the most negative value.

There are two types of output figures. "Fixed" means a network will be drawn on a regular R canvas, while "interactive" will generate a tck/tk canvas. Users can adjust nodes on it by mouse. "none" means no graphics output and return the attributes of vertices and edges.
getcaBIOPATHList

Value

A concept-gene network is generated. A 3-element (1st one: igraph object; 2nd one: a dataframe for vertices attributes; 3rd one: a dataframe for edge attributes) list is returned when output is set to "none".

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

```r
input <- list('ele01'=c('Aa', 'Bb'), 'ele02'=c('Bb', 'Cc', 'dd'))
## Not run: geneConceptNet(input)
```

getcaBIOPATHList

Retrieve caBIO path categories containing given genes

Description

Function to retrieve caBIO pathway IDs containing the given genes. This function is not supported starting this version due to the termination of the caBig project.

Usage

getcaBIOPATHList(lls)

Arguments

lls an Entrez gene IDs vector

Details

The given gene IDs should be Entrez gene IDs. And the return list also only contains Entrez gene IDs besides caBIO pathway IDs.

Value

return an Entrez genes ID list, names of the list are caBIO pathway IDs and elements are Entrez gene IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

getcaBIOPATHTerms

See Also
getcCategoryList

Examples

```r
## Not run: a <- getcaBIOPATHList('1647')
## Not run: length(a)
```

getcBIOPATHTerms Get Pathway names of given REACTOME PATH_DB IDs

Description

Function to map given caBIO pathway IDs to Pathway names. This function is not supported starting this version due to the termination of the caBig project.

Usage

getcBIOPATHTerms(caBIOPATHIDs)

Arguments

caBIOPATHIDs a caBIO pathway IDs vector

Details

caBIO(Cancer Bioinformatics Infrastructure Objects, https://cabig.nci.nih.gov/tools/cabio) integrates three pathway databases from NCI-Nature curated, Biocarta and Reactome. Therefore, terms could be same from different databases and the source library is added the end of each term.

Value

return the caBIO pathway terms of given caBIO pathway IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

```r
## Not run: getcaBIOPATHTerms(c('7622', '289', '7173'))
```
**getCategoryList**

**Retrieve categories containing given genes**

**Description**

Function to retrieve specified category IDs containing given genes.

**Usage**

```r
getCategoryList(geneVector, lib, categoryType)
```

**Arguments**

- `geneVector` : an Entrez gene IDs vector
- `lib` : annotation library to be used to retrieve categories terms.
- `categoryType` : type of category

**Details**

The current version only supports Bioconductor team maintained annotation libraries, like 'org.Bt.eg.db', 'org.Ce.eg.db', 'org.Cf.eg.edu', 'org.Dm.eg.db', 'org.Dr.eg.db', 'org.EcK12.eg.db', 'org.EcSakai.eg.db', 'org.Gg.eg.db', 'org.Hs.eg.db', 'org.Mm.eg.db', 'org.Rn.eg.db' and 'org.Ss.eg.db'.

**Value**

return a category list, names of the list are category IDs and elements are genes IDs.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**


**Examples**

```r
getCategoryList(c('56458', '16590'), 'org.Mm.eg.db', 'PATH')
```
**getCategoryTerms**

Mapping Category IDs to Terms

**Description**
Function to map category IDs to category terms.

**Usage**

```
getCategoryTerms(catIDs, catType, strict = FALSE, missing=c('name', 'keep', 'remove'), nameLength='all', addID=FALSE)
```

**Arguments**

- `catIDs` a character vector containing category IDs
- `catType` type of category
- `strict` logic value to stop conversion if NA is introduced.
- `missing` type of handling NA mapping.
- `nameLength` show how many first letters for long term names, 'all' for full name
- `addID` logic, add term IDs following term names or not

**Details**
The current version only supports 'GO', 'DOLITE', 'KEGG', 'REACTOME.PATH' and 'CABIO.PATH'. There are three types of parameters for variable 'missing'. 'name' means the NA mapping values are replaced by their names. 'keep' means all of NA values are kept. 'remove' means all of NA values are removed.

**Value**
return category terms of given category IDs.

**Author(s)**
Gang Feng, Pan Du and Simon Lin

**References**

**Examples**
```
getCategoryTerms(c("04640", "05221", "05215"), catType='KEGG')
```
**getConceptTable**

Generate top concepts-genes table

**Description**

Function to generate a top concepts-genes table based on a given GeneAnswers instance list.

**Usage**

```r
getConceptTable(gAList, topCat=10, items=c('both', 'geneNum', 'pvalue'), sortBy = c('pvalue', 'geneNum', 'foldChange', 'oddsRatio', 'correctedPvalue'), catTerm=TRUE, strict=FALSE)
```

**Arguments**

- `gAList`: a GeneAnswers instance list
- `topCat`: a numeric or string vector specified categories
- `items`: specify the contents in cells, see details
- `sortBy`: sorted type
- `catTerm`: a logic value to specify whether mapping category IDs to category names
- `strict`: logic value to stop conversion if NA is introduced.

**Details**

A list containing two top concepts-genes tables is generated. The first table consists of gene amounts and enrichment test p values if `items` is set to "both". Only gene amounts are kept if `items` is set to "geneNum" or enrichment test p values if it is set to "p values", while the second table contains enrichment test p values.

**Value**

return a concepts-genes matrix list.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**


**See Also**

geneAnswersBuilder

**Examples**

data(sampleGroupsData)
gAKEGGL <- lapply(sampleGroupsData, geneAnswersBuilder, org.Hs.eg.db', categoryType="KEGG", pvalueT=0.1, verbose=FALSE)
output<- getConceptTable(gAKEGGL)
getConnectedGraph

*build and display a network for given IDs*

## Description

A function to build and display a network by different show types for given IDs and interaction Matrix with specified filtered IDs.

## Usage

```r
getConnectedGraph(graphIDs, idType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction', 'Customized'),
limitedLayers=FALSE, layers=6, treeMergeFilter=FALSE, searchAll=FALSE, showAllNodes=FALSE, directed=FALSE, direction=c('up', 'down', 'both'), filterGraphIDs=NULL, filterLayer=1, verbose=TRUE, ...)
```

## Arguments

- **graphIDs**: a character vector for given IDs
- **idType**: type of IDs, could be one of 'GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction' and 'Customized'
- **edgeM**: a 2-column Matrix representing a network
- **limitedLayers**: logic, user specified layers to stop search
- **layers**: an integer, specify how many layers will be retrieved.
- **treeMergeFilter**: logic, determine whether apply filterGraphIDs during searching a merged tree, see details
- **searchAll**: logic, determine whether search all nodes
- **showAllNodes**: logic, determine whether show all nodes based on searching result
- **directed**: logic, the network is a directed or not
- **direction**: search direction, it could be 'up', 'down' and 'both'. Valid for directed network only.
- **filterGraphIDs**: a character vector for filtered IDs or a 2- or 3-column matrix for extra values.
- **filterLayer**: an integer, specify where filterGraphIDs are applied.
- **verbose**: logic, specify to show information or not.
- **...**: other parameters used by 'buildNet'

## Details

Currently, if idType is 'GO', 'GO.BP', 'GO.CC' or 'GO.MF', edgeM will be ignore.

edgeM is a 2-column matrix. For directional connection, the direction is from column 1 elements to column 2 elements. For non-directional connection, each connection should be reversely presented twice, one is from column 1 element to column 2 element, while another is from column 2 element to column 1 element. In other words, non-directional connection is considered as two reverse directional connections.

filterGraphIDs are applied only at the filterLayer and more outer layers. This means the nodes between the filterLayer layer and the most external layer belong to the filterGraphIDs. The nodes between given graphIDs and the (filterLayer-1) layer are or are not from filterGraphIDs, but those nodes not in filterGraphIDs should be able to be finally connected by given graphIDs and filterGraphIDs.
getDOLiteTerms

The function at first searches a merged tree based on given IDs. During searching, filterGraphIDs could be applied if 'treeMergeFilter' is set to TRUE. If a merged tree is found, searching process stops unless 'searchAll' is set to TRUE. However, 'limitedLayers' is set to TRUE, searching process also stops when searching layers reach 'layers'. Only all filterGraphIDs specified nodes as well as given nodes will be displayed if 'showAllNodes' is set to FALSE, or all connected nodes will be displayed.

See buildnet for network layout.

**Value**

invisibly return a list containing elements to represent a network.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**


**See Also**

buildNet

**Examples**

```r
require(GeneAnswers)
example(GeneAnswers)
filterM <- cbind(rownames(getEnrichmentInfo(x)), -log2(getEnrichmentInfo(x)[,7]), getEnrichmentInfo(x)[,1])
## Not run: getConnectedGraph(rownames(getEnrichmentInfo(x))[c(1:5)], filterGraphIDs=filterM, output='fixed')
```

---

getchLiteTerms

*Get DOLite Terms of Given DOLite IDs*

**Description**

function to map DOLite IDs to DOLite Terms

**Usage**

getDOLiteTerms(DOLiteIDs)

**Arguments**

- **DOLiteIDs** a character vector containing DOLite IDs

**Value**

return a DOLite term vector based on given DOLite IDs.
getGOList

**Author(s)**
Gang Feng, Pan Du and Simon Lin

**References**

**See Also**
getCategoryTerms

**Examples**
```
data('DOLiteTerm')
getDOLiteTerms(c('DOLite:25', 'DOLite:142'))
```

---

**Description**
Get GO list of given genes

**Usage**
```
getGOList(geneVector, lib, GOCat = c("ALL", "BP", "CC", "MF"), level = 1)
```

**Arguments**
- `geneVector`: a character vector containing entrez IDs
- `lib`: annotation library
- `GOCat`: type of Gene Ontology
- `level`: positive integer to specify how many levels GO IDs will be removed.

**Details**
User can specify which subtype of GO can be kept. "ALL" means all of subtypes are kept. Gene Ontology is a tree-like structure. Level can be used to remove top noncritical GO IDs.

**Value**
return a GO list, whose names are GO IDs. Elements are gene entrez IDs belonging to the corresponding GO categories.

**Author(s)**
Gang Feng, Pan Du and Simon Lin
getHomoGeneIDs

References


See Also

getCategoryList

Examples

```r
a <- getGOList(c('56458', '16590'), 'org.Mm.eg.db', GOCat='BP', level=2)
length(a)
```

---

**getHomoGeneIDs**

Get homologous genes of given genes

**Description**

Map given gene IDs to homologous gene IDs.

**Usage**

```r
getHomoGeneIDs(oriGeneIDs, species = c("human", "rat", "mouse", "yeast", "fly"), speciesL = c("human", "rat", "mouse", "yeast", "fly"), mappingMethod = c("direct", "biomaRt", "none"))
```

**Arguments**

- `oriGeneIDs`: a given entrez gene IDs
- `species`: species of the current genes
- `speciesL`: species of the mapped genes
- `mappingMethod`: mapping method, see details

**Details**

There are two mapping methods supported by current version. "direct" only works between human and mouse because most of human gene symbols are capitalized and only the first letter is uppercase for those homogenes in mouse. Another way is by means of package "biomaRt" which contains more information while the network connection is necessary to access biomaRt online server.

**Value**

return homologous gene IDs of given genes

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**

Examples

glist <- c("MRPS35", "NBL1", "PSMD14", "PGK1", "SMC4", "SLC16A1", "CAV1")
## Not run: getListGIF(glist=glist)
## Not run: getListGIF(glist=glist, output="tmp2.png", background="black", type="riflevel")
getMultiLayerGraphIDs

**getMultiLayerGraphIDs** retrieve multilayer interacted nodes for given IDs and interaction Matrix

**Description**
A function to retrieve multilayer interacted nodes for given IDs and interaction Matrix with specified filtered IDs.

**Usage**

```
getMultiLayerGraphIDs(graphIDs, idType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction', 'Customized'), edgeM=NULL, layers=1, filterGraphIDs=NULL, filterLayer=0, UP=TRUE, directed=FALSE, verbose=TRUE)
```

**Arguments**
- **graphIDs**: a character vector for given IDs
- **idType**: type of IDs, could be one of 'GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction' and 'Customized'
- **edgeM**: a 2-column Matrix representing a network
- **layers**: an integer, specify how many layers will be retrieved.
- **filterGraphIDs**: a character vector for filtered IDs
- **filterLayer**: an integer, specify where filterGraphIDs are applied.
- **UP**: logic, determine search Parents or Children. Only valid for directed relation.
- **directed**: logic, the network is a directed or not
- **verbose**: logic, specify to show information or not.

**Details**
Currently, if idType is 'GO', 'GO.BP', 'GO.CC' or 'GO.MF', edgeM will be ignore. edgeM is a 2-column matrix. For directional connection, the direction is from column 1 elements to column 2 elements. For non-directional connection, each connection should be reversely presented twice, one is from column 1 element to column 2 element, while another is from column 2 element to column 1 element. In other words, non-directional connection is considered as two reverse directional connections. filterGraphIDs are applied only at the filterLayer and more outer layers. This means the nodes between the filterLayer layer and the most external layer belong to the filterGraphIDs. The nodes between given graphIDs and the (filterLayer-1) layer are or are not from filterGraphIDs, but those nodes not in filterGraphIDs should be able to be finally connected by given graphIDs and filterGraphIDs.

**Value**
return a list containing elements to represent a network. The first element is a logic value, TRUE means no more connection between the most external layer nodes and other nodes. The second element is a list of layer-length. If the 1st element is FALSE, the length of 2nd element should be (layers + 1). And starting from the 3rd elements, the remaining elements construct a network.

**Author(s)**
Gang Feng, Pan Du and Simon Lin
getNextGOIDs

References


See Also
g getNextGOIDs

Examples

require(GeneAnswers)
example(GeneAnswers)
getNextGOIDs(rownames(getEnrichmentInfo(x))[5:6], UP=FALSE)

Description

A function to retrieve parents or children GO IDs for given IDs with specified filtered IDs.

Usage

g getNextGOIDs(GOIDs, GOType=c(‘GO’, ‘GO.BP’, ‘GO.CC’, ‘GO.MF’), remove=TRUE, filterGOIDs=NULL, UP=TRUE)

Arguments

GOIDs a character GO ID vector
GOType type of GO IDs, ‘GO’, ‘GO.BP’, ‘GO.CC’ and ‘GO.MF’
remove logic, remove the empty GOIDs in the return values
filterGOIDs a character vector for filtered GO IDs
UP logic, determine search Parents or Children.

Details

filterGraphIDs is used to only keep nodes in filterGraphIDs.

Value

return a GO IDs list representing a network.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

g getNextGOIDs(c(‘GO:0050794’, ‘GO:0034960’))
**getPATHList**

Retrieves KEGG categories containing given genes.

**Description**

Function to retrieve KEGG category IDs containing given genes.

**Usage**

```r
getPATHList(geneVector, lib)
```

**Arguments**

- `geneVector`: an Entrez gene IDs vector
- `lib`: annotation library to be used to retrieve KEGG IDs.

**Details**

The current version only supports Bioconductor team maintained annotation libraries, like `org.Bt.eg.db`, `org.Ce.eg.db`, `org.Cf.eg.edu`, `org.Dm.eg.db`, `org.Dr.eg.db`, `org.EcK12.eg.db`, `org.EcSakai.eg.db`, `org.Gg.eg.db`, `org.Hs.eg.db`, `org.Mm.eg.db`, `org.Rn.eg.db` and `org.Ss.eg.db`.

**Value**

return a KEGG genes ID list, names of the list are KEGG IDs and elements are genes IDs.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**


**See Also**

`getCategoryList`

**Examples**

```r
a <- getPATHList(c('56458', '16590'), 'org.Mm.eg.db')
length(a)
```
getPATHTerms  Get Pathway names of given KEGG IDs

Description
Function to map given KEGG IDs to Pathway names.

Usage
getPATHTerms(pathIDs)

Arguments
pathIDs  a KEGG IDs vector

Value
return a KEGG pathway terms of given KEGG IDs.

Author(s)
Gang Feng, Pan Du and Simon Lin

References

See Also
getchCategoryTerms

Examples
getPATHTerms(c('04916', '05221'))

getREACTOMEPATHList  Retrieve REACTOME path categories containing given genes

Description
Function to retrieve REACTOME path_db IDs containing given genes.

Usage
getREACTOMEPATHList(geneVector)

Arguments
geneVector  an Entrez gene IDs vector
getREACTOMEPATHTerms

Details

geneVector should be a vector of Entrez IDs.

Value

return a REACTOME genes ID list, names of the list are REACTOME path IDs and elements are gene IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

getCategoryList

Examples

## Not run: a <- getREACTOMEPATHList(c('8772', '1017'))
## Not run: length(a)

getREACTOMEPATHTerms Get Pathway names of given REACTOME PATH_DB IDs

Description

Function to map given REACTOME PATH_DB IDs to Pathway names.

Usage

getREACTOMEPATHTerms(pathIDs, allowNA=TRUE)

Arguments

pathIDs a REACTOME PATHWAY IDs vector
allowNA logic, to determine whether change unrecognized term names or not

Value

return a REACTOME pathway terms of given REACTOME PATH_DB IDs. If the REACTOME service is not available, the function will stop.

Author(s)

Gang Feng, Pan Du and Simon Lin
**getSingleLayerGraphIDs**

**References**


**Examples**

```r
## Not run: getREACTOMEPATHTerms(c('174143', '453274'))
```

**Description**

A function to retrieve direct interacted nodes for given IDs and interaction Matrix with specified filtered IDs.

**Usage**

```r
getSingleLayerGraphIDs(graphIDs, edgeM, remove=TRUE, filterGraphIDs=NULL, directed=FALSE, UP=TRUE)
```

**Arguments**

- `graphIDs` a character vector for given IDs
- `edgeM` a 2-column Matrix representing connectionship
- `remove` logic, remove the non-connection graphIDs in the return values
- `filterGraphIDs` a character vector for filtered IDs
- `directed` logic, the network is a directed or not
- `UP` logic, determine search Parents or Children. Only valid for directed relation.

**Details**

des.Array is a 2-column matrix. For directional connection, the direction is from column 1 elements to column 2 elements. For non-directional connection, each connection should be reversely presented twice, one is from column 1 element to column 2 element, while another is from column 2 element to column 1 element. In other words, non-directional connection is considered as two reverse directional connections. filterGraphIDs is used to only keep nodes in filterGraphIDs.

**Value**

return a list representing a network.

**Author(s)**

Gang Feng, Pan Du and Simon Lin

**References**

getSymbols

Examples

m <- matrix(c('1', '4', '2', '6', '1', '5', '3', '7', '5', '2'), ncol=2, byrow=TRUE)
m
getSingleLayerGraphIDs(c('1', '2', '3'), m)

# if the connection is not directional, the connection between '5' and '2' will be missed without changing m.
m <- rbind(m, c('2', '5'))
getSingleLayerGraphIDs(c('1', '2', '3'), m)

getSymbols

Convert entrez gene IDs to gene symbols

Description

function to convert given entrez gene IDs to gene symbols.

Usage

getSymbols(geneIDs, data, strict = FALSE, missing=c('name', 'keep', 'remove'))

Arguments

geneIDs an Entrez gene IDs vector
data annotation library
strict logic value to stop conversion if NA is introduced.
missing type of handling NA mapping.

Value

return a gene symbols vector of given gene IDs. There are three types of parameters for variable 'missing'. 'name' means the NA mapping values are replaced by their names. 'keep' means all of NA values are kept. 'remove' means all of NA values are removed.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


Examples

require('org.Mm.eg.db')
getSymbols(c('11651', '11836'), 'org.Mm.eg.db')
**getDescription**

Obtain the total number of genes in the given annotation library

### Description

A function to Obtain the total number of genes in the given annotation library.

### Usage

```r
```  

### Arguments

- **categoryType**: name of given annotation category or NULL for user provided annotation list.
- **known**: logic, specify only known annotation gene enrichment test.
- **annotationLib**: name of given annotation library file or user provided annotation list.

### Details

categoryType could be one of "GO", "GO.BP", "GO.CC", "GO.MF", "DOLITE", "KEGG" and "REACTOME.PATH".

annotationLib could be one of "org.Ag.eg.db", "org.Bt.eg.db", "org.Ce.eg.db", "org.Cf.eg.db", "org.Dm.eg.db", "org.Dr.eg.db", "org.EcK12.eg.db", "org.EcSakai.eg.db", "org.Gg.eg.db", "org.Hs.eg.db", "org.Mm.eg.db", "org.Mmu.eg.db", "org.Pt.eg.db", "org.Rn.eg.db", "org.Ss.eg.db", "org.Xl.eg.db", "org.At.tair.db", "org.Pf.plasmo.db" and "org.Sc.sgd.db". However, if categoryType is set to "REACTOME.PATH", only 'org.At.tair.db'(516), 'org.Ce.eg.db'(627), 'org.Dm.eg.db'(686), 'org.Ec.K12.eg.db'(185), 'org.Ecc.Sakai.eg.db'(185), 'org.Gg.eg.db'(840), 'org.Hs.eg.db'(1019), 'org.Mm.eg.db'(900), 'org.Pf.plasmo.db'(308), 'org.Rn.eg.db'(883) and 'org.Sc.sgd.db'(473) are available. Since DOLITE is designed for human being, currently only 4051 genes are annotated in Disease Ontology. Other species could be mapped to homologous genes by getHomoGeneIDs.

If known is set to TRUE, the enrichment test only considers the genes with annotation. If FALSE, the total number of genes in that species will be returned.

### Value

A number of total genes.

### Author(s)

Gang Feng, Pan Du and Simon Lin

### References


### See Also

geneAnswersBuilder
Examples

getTotalGeneNumber(categoryType='GO.CC', annotationLib='org.Hs.eg.db')

---

**Description**

Function to generate a html format top multigroup Concepts-genes analysis report based on a given GeneAnswers instance list. Currently, this function only works at Mac platform and Java 1.6 or latest version is required.

**Usage**

groupReport(dataMatrix, gAList, topCat=10, methodOfCluster=c('mds', 'sort'), matrixOfHeatmap=NULL, fileName = "multiConceptsGenes.html", title='Multigroup Genes Concepts Analysis', catType=c('GO'), reverseOfCluster=FALSE, colorValueColumn = NULL, annLib=c('org.Hs.eg.db', 'org.Rn.eg.db', 'org.Mm.eg.db', 'org.Dm.eg.db'), bgColor="#ffffff", keepCytoscapeFiles=TRUE, wordleOn=FALSE, ...)

**Arguments**

dataMatrix a top concepts-genes matrix generated by getConceptTable.
gAList a GeneAnswers instance list.
topCat number to specify how many top concepts-genes analysis will show.
methodOfCluster cluster method
matrixOfHeatmap NULL or a concepts-genes matrix generated by getConceptTable, which is used to show enrichment test significance for each concept.
clusterTable cluster data to specify which type of values will be used for cluster.
catTerm logic, determine whether mapping category IDs to names
fileName output html file name
title output html title
catType category type, current version supports 'GO', 'KEGG', 'DOLITE', 'REACTOME.PATH', 'CABIO.PATH' and customized annotation libraries, 'Unknown'.
reverseOfCluster logic, whether reverse the cluster order.
colorValueColumn numbers or column names of geneInput slots of the given GeneAnswers instance list to specify the colors of leaves
annLib annotation library names, current version supports 'org.Hs.eg.db', 'org.Rn.eg.db', 'org.Mm.eg.db' and 'org.Dm.eg.db'.
nameLength show how many first letters for long term names, 'all' for full name, default value is 94.
addID logic, add term IDs following term names or not
interactive logic, determine whether network is interactive or not. Interactive network requires java and flash supports.
bgColor a R compatible color for html background color.

keepCytoscapeFiles logic, determine whether to keep cytoscape files if interactive is set to TRUE

wordleOn logic, determine whether a geneRif based word cloud is generated or not. This requires an internet connection.

... other parameters used by 'sort'

Details

In general, a html format top multigroup Concepts-genes analysis report is generated. It includes a multigroup concepts-genes table, several concepts-genes networks figures and a couple of tables containing genes and their information. colorValueColumn could be NULL, column name or a same length column-name vector as length of the given GeneAnswers instance list. No color for genes if it is NULL. All of GeneAnswers instances are applied color for genes based on the same column name if the length is one. Or the colors of genes in concepts-genes networks are based on the same length column-name vector. If catType is not set to 'Unknown', catTerm in function getConceptTable should be set to FALSE.

Value

no value returned

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

getConceptTable, drawTable

Examples

data(sampleGroupsData)
gAKEGGL <- lapply(sampleGroupsData, geneAnswersBuilder, org.Hs.eg.db', categoryType='KEGG', pvalueT=0.1, verbose=FALSE)
output<- getConceptTable(gAKEGGL, catTerm=FALSE, items='geneNum')
## Not run: groupReport(output[[1]], gAKEGGL, matrixOfHeatmap=output[[2]], clusterTable=NULL, fileName='KEGGtest.html')

HsIALite Human gene interaction matrix

Description

Preprocessed human gene interaction matrix

Usage

data(HsIALite)
humanExpr

Details

a 4-column matrix containing human interacted genes and evidences

References


Examples

data(HsIALite)
HsIALite[1:4,]

Data frame with 86 observations on the 6 variables.

This data frame is a part of expression profile from a human Illumina array experiment.

References


Examples

data(humanExpr)
humanExpr[1:10,]
**humanGeneInput**

*Example human gene data*

**Description**

An example of a group of human gene data.

**Usage**

`data(humanGeneInput)`

**Format**

A data frame with 86 observations. Column names are "Symbol", "foldChange" and "pValue". Row names are gene Entrez IDs. For general usage, row names of geneInput could be anything.

**Details**

Fold change could be negative, which means the ratio of treatment to control is less than 1 and the value is reciprocal of general fold change.

**References**


**Examples**

```r
data(humanGeneInput)
humanGeneInput[1:10,]
```

---

**MmIALite**

*Mouse gene interaction matrix*

**Description**

Preprocessed mouse gene interaction matrix

**Usage**

`data(MmIALite)`

**Details**

a 4-column matrix containing mouse interacted genes and evidences

**References**

mouseExpr

Examples
data(MmIALite)
MmIALite[1:4,]

Example mouse expression data

Description
Example data of mouse expression

Usage
data(mouseExpr)

Format
A data frame with 71 observations on the following 6 variables.

Details
This data frame is a part of expression profile from a mouse Illumina array experiment.

References

Examples
data(mouseExpr)
mouseExpr[1:10,]

mouseGeneInput

Example mouse gene data

Description
An example of a group of mouse gene data.

Usage
data(mouseGeneInput)

Format
A data frame with 71 observations. Column names are "Symbol", "foldChange" and "pValue". Row names are gene Entrez IDs. For general usage, row names of geneInput could be anything.
Details

Fold change could be negative, which means the ratio of treatment to control is less than 1 and the value is reciprocal of general fold change.

References


Examples

data(mouseGeneInput)
mouseGeneInput[1:10,]

data(RnIALite)
RnIALite[1:4,]
sampleGroupsData

Example human expression data

Description
An example data of human expression

Usage
data(sampleGroupsData)

Format
A data frame list containing genes and fold changes from 6 different comparisons.

Details
This data frame is a part of expression profile from a group of human Illumina array experiments.

References

Examples
data(sampleGroupsData)
head(sampleGroupsData)

searchEntrez
Search specified information from Entrez site

Description
A function to search Entrez website by one given keywords list.

Usage
searchEntrez(tagList, species = "human")

Arguments
tagList keyword list to search on Entrez.
species specie for search on Entrez.

Value
an Entrez ID list containing all of relative genes from Entrez database.
Author(s)
Pan Du, Gang Feng and Simon Lin

References

Examples
```r
tagList <- list(FSHR=c("FSHR", "Follicle stimulating hormone receptor"), apoptosis=c('apoptosis'))
## Not run: entrezList <- searchEntrez(tagList, species='mouse')
```

topcaBIO.PATH  

Present top CABIO.PATH enrichment test information

Description
Function to present top CABIO.PATH enrichmentInfo of given GeneAnswers instance. This function is not supported starting this version due to the termination of the caBig project.

Usage
topcaBIO.PATH(x, catTerm = TRUE, keepID=TRUE, ...)

Arguments
- **x**: a given GeneAnswers instance containing CABIO.PATH information
- **catTerm**: logic value to determine whether mapping to CABIO.PATH terms or not
- **keepID**: logic value to determine whether showing CABIO.PATH IDs or not
- ... other parameters to transfer to topCategory

Value
print necessary information on the screen and save into a specified file if request.

Author(s)
Gang Feng, Pan Du and Simon Lin

References

See Also
topCategory

Examples
```r
# x is a GeneAnswers instance with CABIO.PATH test
## Not run: topcaBIO.PATH(x, top=10)
```
Description

Function to present top CABIO.PATH enrichment test information with genes. This function is not supported starting this version due to the termination of the caBig project.

Usage

topcaBIO.PATHGenes(x, catTerm = TRUE, keepID = TRUE, geneSymbol = TRUE, ...)

Arguments

- **x**: a given GeneAnswers instance with CABIO.PATH test
- **catTerm**: logic value to determine whether mapping CABIO.PATH IDs to CABIO.PATH terms
- **keepID**: logic, to determine whether keep CABIO.PATH IDs
- **geneSymbol**: logic value to determine whether mapping gene Entrez IDs to gene symbols
- **...**: other parameters to transfer to topCategoryGenes

Details

See function topCategoryGenes help for details

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategoryGenes

Examples

```r
## Not run: topcaBIO.PATHGenes(x, geneSymbol = TRUE, orderby = 'pvalue', top = 10, topGenes = 'ALL', genesOrderBy = 'pValue')
```
Present top enrichment test information

Description
Function to present top enrichmentInfo of given GeneAnswers instance.

Usage
topCategory(inputX, orderby = c("geneNum", "pvalue", "foldChange", "oddsRatio", "correctedPvalue"), top = 5, file = FALSE, fileName = "topCategory.txt")

Arguments

inputX a given GeneAnswers instance
orderby type to sort enrichmentInfo slot
top integer to specify how many top rows to be presented
file logic value to determine whether save to a file
fileName string to specify file name, default file name is topCategory.txt

Details
orderby could be one of 'geneNum', 'pvalue', 'foldChange', 'oddsRatio' and 'correctedPvalue'.
top could be an integer or 'ALL'. The top former specified categories will be printed on screen
while only 30 categories will be displayed for 'ALL'. All categories can be saved in a specified file.

Value
print necessary information on the screen and save into a specified file if request.

Author(s)
Gang Feng, Pan Du and Simon Lin

References
bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples
# x is a GeneAnswers instance
## Not run: topCategory(x, orderby='pvalue')
topCategoryGenes

Description

Function to present top enrichmentInfo of given GeneAnswers instance with genes.

Usage

```r
topCategoryGenes(inputX, orderby = c("geneNum", "pvalue", "foldChange", "oddsRatio", "correctedPvalue"), top = 5, genesOrderBy = 1, decreasing = FALSE, topGenes = 5, file = FALSE, fileName = "topCategoryGenes.txt")
```

Arguments

- `inputX` a given GeneAnswers instance
- `orderby` type to sort enrichmentInfo slot
- `top` integer to specify how many top rows to be presented
- `genesOrderBy` integer or characters to specify gene ordered column.
- `decreasing` logic value to specify gene order is descending or not
- `topGenes` integer to specify how many top genes to be presented
- `file` logic value to determine whether save to a file
- `fileName` string to specify file name, default file name is topCategoryGenes.txt

Details

orderby could be one of 'geneNum', 'pvalue', 'foldChange', 'oddsRatio' and 'correctedPvalue'.
top could be an integer or 'ALL'. The top former specified categories will be printed on screen
while only 30 categories will be displayed for 'ALL'. All categories can be saved in a specified file.
topGenes is similar to top, but only top 5 genes will be displayed for 'ALL'. genesOrderBy could
be an integer to specify column to be sorted. It can also be the column name. If set to 'none', no
sorting for genes.

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```r
# x is a GeneAnswers instance
## Not run: topCategoryGenes(x, orderby='p')
```
topDOLITE  

Present top DOLITE enrichment test information

Description

Function to present top DOLITE enrichmentInfo of given GeneAnswers instance.

Usage

topDOLITE(x, catTerm=TRUE, keepID=TRUE, ...)

Arguments

- `x` a given GeneAnswers instance containing DOLITE information
- `catTerm` logic value to determine whether mapping to DOLITE terms or not
- `keepID` logic value to determine whether showing IDs or not
- `...` other parameters to transfer to topCategory

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategory

Examples

# x is a GeneAnswers instance with DOLITE test
## Not run: topDOLITE(x, top=10)
topDOLITEGenes

Present top DOLITE enrichment test information with genes

Description

Function to present top DOLITE enrichmentInfo of given GeneAnswers instance with genes.

Usage

topDOLITEGenes(x, catTerm = TRUE, keepID=TRUE, geneSymbol = TRUE, ...)

Arguments

x   a given GeneAnswers instance with DOLITE test
catTerm logic value to determine whether mapping DOLITE IDs to DOLITE terms
keepID logic, to determine whether keep DOLITE IDs
geneSymbol logic value to determine whether mapping gene Entrez IDs to gene symbols
... other parameters to transfer to topCategoryGenes

Details

See function topCategoryGenes help for details

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategoryGenes

Examples

##x is a GeneAnswers instance with DOLITE test
## Not run: topDOLITEGenes(x, geneSymbol=TRUE, orderby='pvalue', top=10, topGenes='ALL', genesOrderBy='pValue')
Description

Function to present top GO enrichment test information of given GeneAnswers instance.

Usage

topGO(x, catTerm = TRUE, keepID = TRUE, ...)

Arguments

x a given GeneAnswers instance containing GO test information
catTerm logic value to determine whether mapping to GO terms or not
keepID logic value to determine whether showing IDs or not
... other parameters to transfer to topCategory

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategory

Examples

# x is a GeneAnswers instance with GO test
## Not run: topGO(x, top=10)
topGOGenes

Present top GO enrichment test information with genes

Description

Function to present top GO enrichmentInfo of given GeneAnswers instance with genes.

Usage

topGOGenes(x, catTerm = TRUE, keepID=TRUE, geneSymbol = TRUE, ...)

Arguments

x a given GeneAnswers instance with GO test
catTerm logic value to determine whether mapping GO IDs to GO terms
keepID logic, to determine whether keep GO IDs
geneSymbol logic value to determine whether mapping gene Entrez IDs to gene symbols
... other parameters to transfer to topCategoryGenes

Details

See function topCategoryGenes help for details

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategoryGenes

Examples

##x is a GeneAnswers instance with GO test
## Not run: topGOGenes(xxx, geneSymbol=F, catTerm=F, orderby='p')
topPATH

Present top KEGG enrichment test information

Description

Function to present top KEGG enrichmentInfo of given GeneAnswers instance.

Usage

topPATH(x, catTerm = TRUE, keepID = TRUE, ...)

Arguments

- **x**: a given GeneAnswers instance containing KEGG information
- **catTerm**: logic value to determine whether mapping to DOLite terms or not
- **keepID**: logic value to determine whether showing IDs or not
- **...**: other parameters to transfer to topCategory

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategory

Examples

# x is a GeneAnswers instance with DOLite test
## Not run: topPATH(x, top=10)
**topPATHGenes**

Present top KEGG enrichment test information with genes

### Description

Function to present top KEGG enrichmentInfo of given GeneAnswers instance with genes.

### Usage

```r
topPATHGenes(x, catTerm = TRUE, keepID = TRUE, geneSymbol = TRUE, ...)
```

### Arguments

- `x`: a given GeneAnswers instance with KEGG test
- `catTerm`: logic value to determine whether mapping KEGG IDs to KEGG terms
- `keepID`: logic, to determine whether keep KEGG IDs
- `geneSymbol`: logic value to determine whether mapping gene Entrez IDs to gene symbols
- `...`: other parameters to transfer to topCategoryGenes

### Details

See function topCategoryGenes help for details

### Value

print necessary information on the screen and save into a specified file if request.

### Author(s)

Gang Feng, Pan Du and Simon Lin

### References


### See Also

~~objects to See Also as topCategoryGenes, ~~~

### Examples

```r
##x is a GeneAnswers instance with KEGG test
## Not run: topPATHGenes(x, geneSymbol=TRUE, orderby='genenum', top=6, topGenes=8, genesOrderBy='foldChange'
```
topREACTOME.PATH

Present top REACTOME.PATH enrichment test information

Description

Function to present top REACTOME.PATH enrichmentInfo of given GeneAnswers instance.

Usage

topREACTOME.PATH(x, catTerm = TRUE, keepID=TRUE, ...)

Arguments

x a given GeneAnswers instance containing REACTOME.PATH information
catTerm logic value to determine whether mapping to REACTOME.PATH terms or not
keepID logic value to determine whether showing REACTOME.PATH IDs or not
... other parameters to transfer to topCategory

Value

print necessary information on the screen and save into a specified file if request.

Author(s)

Gang Feng, Pan Du and Simon Lin

References


See Also

topCategory

Examples

# x is a GeneAnswers instance with REACTOME.PATH test
## Not run: topREACTOME.PATH(x, top=10)
**topREACTOME.PATHGenes**

Present top REACTOME.PATH enrichment test information with genes

### Description
Function to present top REACTOME.PATH enrichment Info of given GeneAnswers instance with genes.

### Usage

topREACTOME.PATHGenes(x, catTerm = TRUE, keepID=TRUE, geneSymbol = TRUE, ...)

### Arguments
- **x**
  a given GeneAnswers instance with REACTOME.PATH test
- **catTerm**
  logic value to determine whether mapping REACTOME.PATH IDs to REACTOME.PATH terms
- **keepID**
  logic, to determine whether keep REACTOME.PATH IDs
- **geneSymbol**
  logic value to determine whether mapping gene Entrez IDs to gene symbols
- **...**
  other parameters to transfer to topCategoryGenes

### Details
See function topCategoryGenes help for details

### Value
print necessary information on the screen and save into a specified file if request.

### Author(s)
Gang Feng, Pan Du and Simon Lin

### References

### See Also
topCategoryGenes

### Examples
```r
##x is a GeneAnswers instance with REACTOME.PATH test
## Not run: topREACTOME.PATHGenes(x, geneSymbol=TRUE, orderby='pvalue', top=10, topGenes='ALL', genesOrderBy=
```
Index

*Topic IO*
  topcaBIO.PATH, 52
topcaBIO.PATHGenes, 53
topCategory, 54
topCategoryGenes, 55
topDOLITE, 56
topDOLITEGenes, 57
topGO, 58
topGOGenes, 59
topPATH, 60
topPATHGenes, 61
topREACTOME.PATH, 62
topREACTOME.PATHGenes, 63

*Topic classes*
  GeneAnswers-class, 14

*Topic datasets*
  DmIALite, 9
  DO, 9
  DOLite, 10
  DOLiteTerm, 10
  HsIALite, 46
  humanExpr, 47
  humanGeneInput, 48
  MmIALite, 48
  mouseExpr, 49
  mouseGeneInput, 49
  RnIALite, 50
  sampleGroupsData, 51

*Topic methods*
  buildNet, 4
  caBIO2entrez, 6
categoryNet, 7
chartPlots, 8
drawTable, 11
tenzz2caBIO, 12
geneAnnotationHeatmap, 13
geneAnswersBuilder, 16
geneAnswersChartPlots, 18
geneAnswersConceptNet, 19
geneAnswersConceptRelation, 20
geneAnswersConcepts, 21
geneAnswersHeatmap, 22
geneAnswersHomoMapping, 23
geneAnswersReadable, 24
geneAnswersSort, 25
geneConceptNet, 26
getcaBIOPATHList, 27
getcaBIOPATHTerms, 28
categoryList, 29
categoryTerms, 30
getConceptTable, 31
categoryNet, 7
22
class:GeneAnswers (GeneAnswers-class), 14

DmIALite, 9
DO, 9
DOLite, 10
DOLiteTerm, 10
drawTable, 11

tenzz2caBIO, 12
gene2DO.map (DO), 9

GeneAnswers-package, 3
buildNet, 4, 33
caBIO2entrez, 6

categoryNet, 7

class:GeneAnswers (GeneAnswers-class), 14

DmIALite, 9
DO, 9
DOLite, 10
DOLiteTerm, 10
drawTable, 11

tenzz2caBIO, 12
gene2DO.map (DO), 9

GeneAnswers-package, 3
buildNet, 4, 33
caBIO2entrez, 6

categoryNet, 7

class:GeneAnswers (GeneAnswers-class), 14

DmIALite, 9
DO, 9
DOLite, 10
DOLiteTerm, 10
drawTable, 11

tenzz2caBIO, 12
gene2DO.map (DO), 9

GeneAnswers-package, 3
buildNet, 4, 33
caBIO2entrez, 6

categoryNet, 7

class:GeneAnswers (GeneAnswers-class), 14

DmIALite, 9
DO, 9
DOLite, 10
DOLiteTerm, 10
drawTable, 11

tenzz2caBIO, 12
gene2DO.map (DO), 9

GeneAnswers-package, 3
buildNet, 4, 33
caBIO2entrez, 6

categoryNet, 7

class:GeneAnswers (GeneAnswers-class), 14

DmIALite, 9
DO, 9
DOLite, 10
DOLiteTerm, 10
drawTable, 11

tenzz2caBIO, 12
gene2DO.map (DO), 9
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>geneAnnotationHeatmap</code></td>
<td>13, 23</td>
</tr>
<tr>
<td><code>GeneAnswers</code></td>
<td>3</td>
</tr>
<tr>
<td><code>GeneAnswers-class</code></td>
<td>14</td>
</tr>
<tr>
<td><code>geneAnswersBuilder</code></td>
<td>14, 16, 16, 31, 44</td>
</tr>
<tr>
<td><code>geneAnswersChartPlots</code></td>
<td>18</td>
</tr>
<tr>
<td><code>geneAnswersConceptNet</code></td>
<td>19</td>
</tr>
<tr>
<td><code>geneAnswersConceptRelation</code></td>
<td>20</td>
</tr>
<tr>
<td><code>geneAnswersConcepts</code></td>
<td>21</td>
</tr>
<tr>
<td><code>geneAnswersHeatmap</code></td>
<td>22</td>
</tr>
<tr>
<td><code>geneAnswersReadable</code></td>
<td>23</td>
</tr>
<tr>
<td><code>geneAnswersSort</code></td>
<td>25</td>
</tr>
<tr>
<td><code>getAnnLib(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getAnnLib,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getcaBIOPATHList</code></td>
<td>27</td>
</tr>
<tr>
<td><code>getcaBIOPATHTerms</code></td>
<td>28</td>
</tr>
<tr>
<td><code>getCategoryList</code></td>
<td>29, 35, 39, 41</td>
</tr>
<tr>
<td><code>getCategoryTerms</code></td>
<td>6, 24, 25, 30, 34, 40</td>
</tr>
<tr>
<td><code>getCategoryType(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getCategoryType,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getConceptTable</code></td>
<td>11, 12, 31, 45, 46</td>
</tr>
<tr>
<td><code>getConnectedGraph</code></td>
<td>21, 32</td>
</tr>
<tr>
<td><code>getDOLiteTerms</code></td>
<td>33</td>
</tr>
<tr>
<td><code>getEnrichmentInfo(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getEnrichmentInfo,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGeneExprProfile(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGeneExprProfile,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGeneInput(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGeneInput,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGenesInCategory(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGenesInCategory,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getGOList</code></td>
<td>16, 34</td>
</tr>
<tr>
<td><code>getHomoGeneIDs</code></td>
<td>24, 35, 44</td>
</tr>
<tr>
<td><code>getListGIF</code></td>
<td>36</td>
</tr>
<tr>
<td><code>getMultiLayerGraphIDs</code></td>
<td>20, 37</td>
</tr>
<tr>
<td><code>getNextGOIDs</code></td>
<td>38</td>
</tr>
<tr>
<td><code>getPATHList</code></td>
<td>39</td>
</tr>
<tr>
<td><code>PATHTerms</code></td>
<td>40</td>
</tr>
<tr>
<td><code>getPValueT(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getPValueT,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getReactomePATHList</code></td>
<td>40</td>
</tr>
<tr>
<td><code>getReactomePATHTerms</code></td>
<td>41</td>
</tr>
<tr>
<td><code>getSingleLayerGraphIDs</code></td>
<td>38, 42</td>
</tr>
<tr>
<td><code>getSymbols</code></td>
<td>25, 43</td>
</tr>
<tr>
<td><code>getType(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getType,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>getTotalGeneNumber</code></td>
<td>17, 44</td>
</tr>
<tr>
<td><code>groupReport</code></td>
<td>12, 45</td>
</tr>
<tr>
<td><code>help</code></td>
<td>7</td>
</tr>
<tr>
<td><code>HsIALite</code></td>
<td>46</td>
</tr>
<tr>
<td><code>humanExpr</code></td>
<td>47</td>
</tr>
<tr>
<td><code>humanGeneInput</code></td>
<td>48</td>
</tr>
<tr>
<td><code>MmIALite</code></td>
<td>48</td>
</tr>
<tr>
<td><code>mouseExpr</code></td>
<td>49</td>
</tr>
<tr>
<td><code>mouseGeneInput</code></td>
<td>49</td>
</tr>
<tr>
<td><code>par</code></td>
<td>11</td>
</tr>
<tr>
<td><code>RnIALite</code></td>
<td>50</td>
</tr>
<tr>
<td><code>sampleGroupsData</code></td>
<td>51</td>
</tr>
<tr>
<td><code>searchEntrez</code></td>
<td>51</td>
</tr>
<tr>
<td><code>setAnnLib(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setAnnLib,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setCategoryType(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setCategoryType,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setEnrichmentInfo(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setEnrichmentInfo,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setGeneExprProfile(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setGeneExprProfile,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setGeneInput(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setGeneInput,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setGenesInCategory(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setGenesInCategory,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setPValueT(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setPValueT,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setTestType(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>setTestType,GeneAnswers-method(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
<tr>
<td><code>show(GeneAnswers-class)</code></td>
<td>14</td>
</tr>
</tbody>
</table>
show, GeneAnswers-method (GeneAnswers-class), 14
summary (GeneAnswers-class), 14
summary, GeneAnswers-method (GeneAnswers-class), 14

topcaBIO.PATH, 52
topcaBIO.PATHGenes, 53
topCategory, 52, 54, 56, 58, 60, 62
topCategoryGenes, 53, 55, 57, 59, 61, 63
topDOLITE, 56
topDOLITEGenes, 57
topGO, 58
topGOGenes, 59
topPATH, 60
topPATHGenes, 61
topREACTOME.PATH, 62
topREACTOME.PATHGenes, 63