GeneAnswers

April 20, 2011

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

```
buildNet(graphIDs, idType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInteraction', 
  annLib=c('org.Hs.eg.db', 'org.Mm.eg.db', 'org.Rn.eg.db', 'org.Dm.eg.db', 'cust
vertexSize = NULL, edgeColor = NULL, colorMap=NULL, zeroColorIndex=NULL, matchMc
directed=FALSE, direction=c('up', 'down', 'both'), showModeForNodes=c('nodes', '
```

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 chacater 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.

`steric` logic, specify whether show 3D network. Because of igraph limited support for 3D, this function is just for fun!

`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 filterGraphIDs.

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 tothe 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

getCategoryTerms

Examples

```
require(GeneAnswers)
example(GeneAnswers)
filterM <- cbind(rownames(getEnrichmentInfo(x)), -log2(getEnrichmentInfo(x)[,7]), getEnri

## Not run: buildNet(rownames(getEnrichmentInfo(x))[6:9], layers=5, filterGraphIDs=filter
## Not run: buildNet(rownames(getEnrichmentInfo(x))[242:244], layers=2, filterGraphIDs=fi
## Not run: buildNet(rownames(getEnrichmentInfo(x))[6:9], layers=3, filterGraphIDs=filter
```

---

caBIO2entrez *map caBIO gene IDs to Entrez gene IDs*

Description

Function to map the given caBIO gene IDs to the Entrez gene IDs.

Usage

```
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
## Not run: caBIO2entrez(c('2933', '7326'))
```

topcaBIO.PATHGenes *Present top CABIO.PATH enrichment test information with genes*

Description

Function to present top CABIO.PATH enichmentInfo of given GeneAnswers instance with genes.

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategoryGenes`

Examples

```
##x is a GeneAnswers instance with CABIO.PATH test
## Not run: topcaBIO.PATHGenes(x, geneSymbol=TRUE, orderby='pvalue', top=10, topGenes='AL
```

`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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

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)
```

---

chartPlots *Pie Chart and Bar Plots*

Description

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

Usage

```
chartPlots(x, chartType = c("pieChart", "barPlot", "all"), specifiedCols = c("ge
```

Arguments

`x` a 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

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)
```

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

Du, P., Feng, G., Flatow, J., Song, J., Holko, M., Kibbe, W.A. and Lin, S.M., (2009) ’From disease ontology to disease-ontology lite: statistical methods to adapt a general-purpose ontology for the test of gene-ontology associations’, Bioinformatics 25(12):i63-8

Examples

```
data(DOLite)
DOLite[1:2]
```

DOLiteTerm *Disease Ontology Annotation Vector*

Description

Disease Ontology Annotation Vector

Usage

```
data(DOLiteTerm)
```

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

Du, P., Feng, G., Flatow, J., Song, J., Holko, M., Kibbe, W.A. and Lin, S.M., (2009) 'From disease ontology to disease-ontology lite: statistical methods to adapt a general-purpose ontology for the test of gene-ontology associations', Bioinformatics 25(12):i63-8

Examples

```
data(DOLiteTerm)
DOLiteTerm[1:10]
```

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

```
data(DO)

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

`drawTable` *Concept-Gene Networking Plotting*

Description

A function to generate a multigroup concepts-genes table

Usage

```
drawTable(dataMatrix, topCat=10, heatMap=TRUE, matrixOfHeatmap=NULL, clusterTabl
addRowLabel=TRUE, cex.axis=c(1.1, 0.9), reverseOfCluster=FALSE, xGridLine=FALSE,
```

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

`colorBar` logic, whether show color bar or not

`newWindow` logic, whether present table in current active window or not

`endOfColoBar` a character string for color bar.

`...` other parameters used by 'sort'

Details

an image based multigroup concepts-genes table is generated. If heatmap is on, the statistical significant cells are shaded by different level green. Specified top gene amounts are highlighted as red.

Value

No return value.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., ’A collection of bioconductor methods to visualize gene-list annotations’, BMC Research Notes 2010, 3:10

See Also

See Also as `getConceptTable`, `groupReport`

Examples

```
data(sampleGroupsData)
gAKEGGL <- lapply(sampleGroupsData, geneAnswersBuilder, 'org.Hs.eg.db', categoryType='KEG
#output<- getConceptTable(gAKEGGL, items='geneNum')
## Not run: drawTable(output[[1]], matrixOfHeatmap=output[[2]], mar=c(2,15,3,2), clusterT
```

`entrez2caBIO` *map Entrez gene IDs to caBIO gene IDs*

Description

Function to map the given Entrez gene IDs to the caBIO gene IDs.

Usage

```
entrez2caBIO(lls)
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., ’A collection of bioconductor methods to visualize gene-list annotations’, BMC Research Notes 2010, 3:10

Examples

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

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

Description

Function to make a concept-gene cross tabulation

Usage

```
geneAnnotationHeatmap(annotationList, dataMatrix = NULL, addGeneLabel = TRUE, cc
```

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

`cex.axis` integer vector to specify the character size of row and column labels

`mapType` string to specify concept-gene map type

`displayAll` logic, specify to show all of gene expression profile or remove redundant entries.

`symmetry` logic, indicate the values corresponding to two extreme colors are same if TURE.

`colorBar` logic, show colorbar or not

`colorBarLabel`
character vector to show color bar label.

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
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)
```

---

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("
```

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`.

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.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". 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, "caBIO.PATH" for NCI-Nature curated, Biocarta and REACTOME, which might not work with all available Entrez annotation libraries, please refer `getTotalGeneNumber` for details. For user provided annotation library, it should be NULL in most cases.

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", "ecolistraink12", "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, entrichmentInfo, etc.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getTotalGeneNumber`

Examples

```
data('humanExpr')
data('humanGeneInput')
x <- geneAnswersBuilder(humanGeneInput, 'org.Hs.eg.db', categoryType='GO.BP', testType='h
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('
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`chartPlots`

Examples

```
example(GeneAnswers)
## Not run: geneAnswersChartPlots(x)
```

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.

`annLib`**:** 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:

`getGeneInput(GeneAnswers)`**:** Access the geneInput slot of `GeneAnswers` object.

`getTestType(GeneAnswers)`**:** Access the testType slot of `GeneAnswers` object.

`getPValueT(GeneAnswers)`**:** Access the pvaluteT slot of `GeneAnswers` object.

`getGenesInCategory(GeneAnswers)`**:** Access the genesInCategory slot of `GeneAnswers` object.

`getGeneExprProfile(GeneAnswers)`**:** Access the geneExprProfile slot of `GeneAnswers` object.

`getAnnLib(GeneAnswers)`**:** Access the annLib slot of `GeneAnswers` object.

`getCategoryType(GeneAnswers)`**:** Access the categoryType slot of `GeneAnswers` object.

`getEnrichmentInfo(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 pvaluteT slot of `GeneAnswers` object.

`setGenesInCategory(GeneAnswers, genesInCategory)`: Assign the genesInCategory slot of `GeneAnswers` object.

`setGeneExprProfile(GeneAnswers, geneExprProfile)`: Assign the geneExprProfile slot of `GeneAnswers` object.

`setAnnLib(GeneAnswers, annLib)`: Assign the annLib slot of `GeneAnswers` object.

`setCategoryType(GeneAnswers, type=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'DOLITE', '`
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`geneAnswersBuilder`

Examples

```
data('humanExpr')
data('humanGeneInput')
x <- geneAnswersBuilder(humanGeneInput, 'org.Hs.eg.db', categoryType='GO.BP', testType='h
class(x)
```

geneAnswersConceptNet *Concept-Gene Networking Plotting*

Description

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

Usage

```
geneAnswersConceptNet(x, colorValueColumn = NULL, centroidSize = c("pvalue", "ge
```

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 cencept 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 tck/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

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

getMultiLayerGraphIDs, geneConceptNet

Examples

```
example(GeneAnswers)
## Not run: geneAnswersConceptNet(x, colorValueColumn='foldChange', centroidSize='pvalue'
```

---

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,
```

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

conceptsIDs 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getConnectedGraph`

Examples

```
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', 'oddsRa
```

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 cencept 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 tck/tk canvas. Fixed figure is a non-interactive png figure.

Value

One category-linkage 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

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`categoryNet`

Examples

```
example(GeneAnswers)
## Not run: geneAnswersConcepts(x, centroidSize='pvalue', output='interactive')
```

`geneAnswersHeatmap` *Generate Concept-Gene Tabulates*

Description

A function to generate specified Concept-Gene Tabulates

Usage

```
geneAnswersHeatmap(x, showCats = c(1:5), catTerm = FALSE, geneSymbol = FALSE, ca
```

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.

Value

The function will generate a map without return value.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`geneAnnotationHeatmap`

Examples

```
example(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

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

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

getHomoGeneIDs

Examples

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

GeneAnswers-package *Integrated Interpretation of Genes*

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

Package:	GeneAnswers
Type:	Package
Version:	1.6.0
Date:	2010-10-14
License:	LGPL version 2 or newer

Author(s)

Gang Feng, Pan Du, Tian Xia, Warren Kibbe and Simon Lin

Maintainer: Gang Feng <g-feng@northwestern.edu> and Pan Du <dupan@northwestern.edu>

References

1. Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

2. Du, P., Feng, G., Flatow, J., Song, J., Holko, M., Kibbe, W.A. and Lin, S.M., (2009) 'From disease ontology to disease-ontology lite: statistical methods to adapt a general-purpose ontology for the test of gene-ontology associations', Bioinformatics 25(12):i63-8

3. Osborne, J.D., Flatow, J., Holko, M., Lin, S.M., Kibbe, W.A., Zhu, L.J., Danila, M.I., Feng, G. and Chisholm, R.L., Annotating the human genome with Disease Ontology. BMC Genomics. 2009 Jul 7;10 Suppl 1:S6.

Examples

```
data('humanExpr')
data('humanGeneInput')
x <- geneAnswersBuilder(humanGeneInput, 'org.Hs.eg.db', categoryType='GO.BP', testType='h
class(x)
```

---

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, verbos
```

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. Occationally, 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 and caBIO test.

Value

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

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

getSymbols, getCategoryTerms

Examples

```
example(GeneAnswers)
xx <- 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", "c
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

GeneAnswers-class

Examples

```
example(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 =
```

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 tothe 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.

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

---

geneFunSummarize *Summarize gene functions (annotations) based collective annotation evidences associated with ontology terms*

Description

Summarize gene functions (annotations) based collective annotation evidences associated with ontology terms

Usage

```
geneFunSummarize(genes, gene2Onto, Onto2offspring, rmOntoID = c("DOID:4", "DOID:
```

Arguments

`genes` a vector of Entrez Gene IDs to do gene function summarization

`gene2Onto` a list gene 2 Ontology mapping (Ontology IDs, duplicated IDs are allowed, which is equivalent to multiple evidences with the same function)

`Onto2offspring` a list or graphNEL object shows the relations of a ontology id to all its offsprings

`rmOntoID` some ontology ids (like root id or too general ontology ids) can be pre-removed in the estimation

`p.value.th` the p-value threshold used to determine the significance of function enrichment

`fdr.adjust` the FDR estimation methods used to estimate the FDR of function enrichment

`minNumTh` the minimum number of evidences required to claim as significant enriched ontology term

`includeTestOnto` whether include the direct evidence of the testing ontology term itself

`directMapConstraint` whether only consider the ontology ids with direct evidence mappings

Value

The function return is a list with the same length of input "genes". Each element of the list is the summarization of a gene. The gene summarization is also a list with the following items

`allEvidence` all evidences (ontology terms) related with the testing gene

`sigOntoInfo` the significant ontology terms associated with the testing gene

`bestOntoInfo` the information of the most significant ontology term associated with the testing gene

The gene summarization object also includes the attributes of the input parameter settings.

Author(s)

Pan Du, Simon Lin, Gilbert Feng, Warren Kibbe

References

Pan Du, Simon Lin, Gilbert Feng, Warren Kibbe, "GeneRIFcompendiate: Ranked gene annotations using collective GeneRIF associations and ontology terms", under review

See Also

See Also `plotGeneFunSummary` and `simplifyGeneFunSummary`

Examples

```
data(DO)
## test all ontology terms related with gene PEBP1 (Entrez Gene ID: 5037)
geneSummary <- geneFunSummarize('5037', gene2DO.map, DO.graph.closure.gene, p.value.th=0.
## the p.values of all related ontology terms
pValue <- sapply(geneSummary[[1]]$sigOntoInfo, function(x) x$pValue)
pValue
## plot the relations of the summarized gene annotation
plotGeneFunSummary(geneSummary, onto.graph=DO.graph.gene, onto.graph.closure=DO.graph.clo
```

`getcaBIOPATHList` *Retrieve caBIO path categories containing given genes*

Description

Function to retrieve caBIO pathway IDss containing the given genes.

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 pahtway IDs and elements are Entrez gene IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getCategoryList`

Examples

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

`getcaBIOPATHTerms` *Get Pathway names of given REACTOME PATH_DB IDs*

Description

Function to map given caBIO pathway IDs to Pathway names.

Usage

```
getcaBIOPATHTerms(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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

`getCategoryList` *Retrieve categories containing given genes*

Description

Function to retrieve specified category IDs containing given genes.

Usage

```
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
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', 'rem
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

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

```
getConceptTable(gAList, topCat=10, items=c('both', 'geneNum', 'pvalue'),  sortBy
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

geneAnswersBuilder

Examples

```
data(sampleGroupsData)
gAKEGGL <- lapply(sampleGroupsData, geneAnswersBuilder, 'org.Hs.eg.db', categoryType='KEG
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

```
getConnectedGraph(graphIDs, idType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'GeneInter
searchAll=FALSE, showAllNodes=FALSE, directed=FALSE, direction=c('up', 'down', '
```

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 chacater 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.

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

buildNet

Examples

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

---

getDOLiteTerms   *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.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

getCategoryTerms

Examples

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

getGOList *Get GO list of given genes*

Description

Retrieve GO IDs based on given gene IDs.

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

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getCategoryList`

Examples

```
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

```
getHomoGeneIDs(oriGeneIDs, species = c("human", "rat", "mouse", "yeast", "fly"),
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
getHomoGeneIDs(c('56458', '16590'), species='m', speciesL='h', mappingMethod='direct')
```

`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', 'GeneI
```

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 chacater 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 FASLE, 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

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getSingleLayerGraphIDs`

Examples

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

`getNextGOIDs` *retrieve parents or children GO IDs for given GO IDs*

Description

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

Usage

```
getNextGOIDs(GOIDs, GOType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF'), remove=TRUE, filt
```

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 chacater 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
getNextGOIDs(c('GO:0050794','GO:0034960'))
```

`getPATHList` *Retrieve KEGG categories containing given genes*

Description

Function to retrieve KEGG category IDs containing given genes.

Usage

```
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getCategoryList`

Examples

```
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getCategoryTerms`

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, lib)
```

Arguments

`geneVector` an Entrez gene IDs vector

`lib` annotation library to be used to retrieve REACTOME path_db IDs 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'. If the REACTOME service is not available, the function will stop.

Value

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

Author(s)

Gang Feng, Pan Du and Simon Lin

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getCategoryList`

Examples

```
## Not run: a <- getREACTOMEPATHList(c('8772', '1017'), 'org.Hs.eg.db')
## 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(pathDBIDs, allowNA=TRUE)
```

Arguments

`pathDBIDs` a REACTOME PATH_DB 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

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

---

getSingleLayerGraphIDs

retrieve direct interacted nodes for given IDs and interaction Matrix

---

Description

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

Usage

```
getSingleLayerGraphIDs(graphIDs, edgeM, remove=TRUE, filterGraphIDs=NULL, UP=TRU
```

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 chacater vector for filtered IDs
`UP`	logic, determine search Parents or Children. Only valid for directed relation.

Details

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 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

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
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

`getTotalGeneNumber` *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

```
getTotalGeneNumber(categoryType=c('GO', 'GO.BP', 'GO.CC', 'GO.MF', 'DOLITE', 'KE
'org.Dr.eg.db', 'org.EcK12.eg.db', 'org.EcSakai.eg.db', 'org.Gg.eg.db', 'org.Hs.
'org.Ss.eg.db', 'org.Xl.eg.db', 'org.At.tair.db', 'org.Pf.plasmo.db', 'org.Sc.sg
```

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", "REACTOME.PATH" and "CABIO.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 caegoryType is set to "REACTOME.PATH", only 'org.At.tair.db'(516), 'org.Ce.eg.db'(627), 'org.Dm.eg.db'(686), 'org.EcK12.eg.db'(185), 'org.EcSakai.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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

geneAnswersBuilder

Examples

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

---

groupReport *Generate a multigroup Concepts-genes analysis report*

Description

Function to generate a html format top multigroup Concepts-genes analysis report based on a given GeneAnswers instance list.

Usage

```
groupReport(dataMatrix, gAList, topCat=10, methodOfCluster=c('mds', 'sort'), mat
fileName = "multiConceptsGenes.html", title='Multigroup Genes Concepts Analysis'
reverseOfCluster=FALSE,  colorValueColumn = NULL, annLib=c('org.Hs.eg.db', 'org.
bgColor='#ffffcc', keepCytoscapeFiles=TRUE, ...)
```

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 libarry 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

`...` 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 instantce 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`getConceptTable`, `drawTable`

Examples

```
data(sampleGroupsData)
gAKEGGL <- lapply(sampleGroupsData, geneAnswersBuilder, 'org.Hs.eg.db', categoryType='KEG
output<- getConceptTable(gAKEGGL, catTerm=FALSE, items='geneNum')
groupReport(output[[1]], gAKEGGL,  matrixOfHeatmap=output[[2]], clusterTable=NULL, fileNa
```

`humanExpr` *Example human expression data*

Description

An example data of human expression

Usage

```
data(humanExpr)
```

Format

A data frame with 86 observations on the 6 variables.

Details

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

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

`mouseExpr` *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 date frame is a part of expression profile from a mouse Illumina array experiment.

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
data(mouseGeneInput)
mouseGeneInput[1:10,]
```

`plotGeneFunSummary` *plot the summarized gene annotation (ontologies) of a gene*

Description

plot ontology graphs of the summarized gene annotation (ontologies), which is return by function `geneFunSummarize`

Usage

```
plotGeneFunSummary(geneFunSummarizeResult, onto.graph, selGene = NULL, onto.grap
```

Arguments

`geneFunSummarizeResult`
the output of `geneFunSummarize` function

`onto.graph` an ontology graph in graphNEL class, which keeps the graph of the entire ontology DAG. Each node of it is an ontology ID

`selGene` a subset of genes to plot ontology graph. If it is NULL, all plots of all genes in the "geneFunSummarizeResult" will be plotted.

`onto.graph.closure`
a flatten ontology graph in graphNEL class, whose edges represent offspring relationships.

`allOntoID.direct`
a vector of ontology IDs, which has direct gene mappings.

`fdr.adjust` FDR adjustment methods used in p-value estimation.

`showMiniSet` whether to high-light miniSet in the ontology plot

`highlightBest`
whether to high-light the most significant ontology term.

`miniSetPvalue`
threshold of miniSet p-value

`ID2Name` a named vector of ontology IDs to ontology terms mapping

`savePrefix` the filename prefix of the image files

`geneSymbol` whether use gene symbol in the file name.

`saveImage` whether save images as files.

`selectionMethod`
the method to select ontology nodes included in the plot graph

`lib` the library used to get gene symbols of the Entrez Gene ID

`...` other parameters used by function `plotOntologyGraph`

Value

Invisibly return a list of ontology subgraph (in graphGEL class), which correspond to the genes in the "geneFunSummarizeResult" (or "selGene" if provided as an input parameter).

Author(s)

Pan DU

See Also

See Also `plotGraph` and `plotOntologyGraph`

Examples

```
data(DO)
geneSummary <- geneFunSummarize('4267', gene2DO.map, DO.graph.closure.gene)[[1]]
plotGeneFunSummary(geneSummary['4267'], onto.graph=DO.graph.gene, onto.graph.closure=DO.g
```

plotGraph *plot and render a graphNEL object*

Description

plot and render a graphNEL object

Usage

```
plotGraph(graph, layoutAttrs = NULL, layoutNodeAttrs = NULL, layoutEdgeAttrs = N
```

Arguments

`graph` a graphNEL object includes the ontology subgraph to plot

`layoutAttrs` layoutAttrs used by the Rgraphviz function `layoutGraph`

`layoutNodeAttrs` layoutNodeAttrs used by the Rgraphviz function `layoutGraph`

`layoutEdgeAttrs` layoutEdgeAttrs used by the Rgraphviz function `layoutGraph`

`nodeRenderAttrs` nodeRenderAttrs used by the graph function `nodeRenderInfo`

`edgeRenderAttrs` edgeRenderAttrs used by the graph function `edgeRenderInfo`

Value

Invisibly return a list, which include following elements:

`graph` a graphNEL object includes the ontology subgraph to plot

`layoutAttrs` layoutAttrs used by the Rgraphviz function `layoutGraph`

`layoutNodeAttrs` layoutNodeAttrs used by the Rgraphviz function `layoutGraph`

`layoutEdgeAttrs` layoutEdgeAttrs used by the Rgraphviz function `layoutGraph`

`nodeRenderAttrs` nodeRenderAttrs used by the graph function `nodeRenderInfo`

`edgeRenderAttrs` edgeRenderAttrs used by the graph function `edgeRenderInfo`

These invisibly return information can be used as the input of function `plotGraph`.

Author(s)

Pan DU

See Also

See Also `plotOntologyGraph`

Examples

```
library(graph)
data(DO)
geneSummary <- geneFunSummarize('4267', gene2DO.map, DO.graph.closure.gene)[[1]]
pValue <- sapply(geneSummary$sigOntoInfo, function(x) x$pValue)
tmp <- plotOntologyGraph(pValue, geneSummary$allEvidence, DO.graph.gene, onto.graph.closu
# modify the graph
tmpGraph <- tmp$graph
nn <- nodes(tmpGraph)
# remove a node in the graph
tmpGraph <- subGraph(nn[nn != 'DOID:3347'], tmpGraph)
tmpGraph <- addEdge("DOID:1115", "DOID:3376", tmpGraph, 2)
edgeRenderAttrs <- tmp$edgeRenderAttrs
edgeRenderAttrs$lty <- c(edgeRenderAttrs$lty, "DOID:1115~DOID:3376"="dashed")
plotGraph(tmpGraph, layoutAttrs=tmp$layoutAttrs, nodeRenderAttrs=tmp$nodeRenderAttrs, edg
```

`plotOntologyGraph` *plot the ontology graph*

Description

plot the ontology graph in graphNEL class, the color of ontology is based on the enrichment p.value

Usage

```
plotOntologyGraph(onto.pValue, relatedOntoID, onto.graph, bestOntoID = NULL, ont
```

Arguments

`onto.pValue` a vector ontology p.values, whose names are corresponding ontology IDs

`relatedOntoID` a vector of related ontology IDs, which will be highlighted.

`onto.graph` an ontology graph in graphNEL class, which keeps the graph of the entire ontology DAG. Each node of it is an ontology ID

`bestOntoID` the ontology ID which has most significant p-value

`onto.graph.closure` a flatten ontology graph in graphNEL class, whose edges represent offspring relationships.

`rootID` the root ontology ID in the ontology graph

`ID2Name` a named vector of ontology IDs to ontology terms mapping

`p.value.th` enrichment p.value threshold to claim as significant

`fillColor` the default fill color of the ontology nodes

`colorLevel` the level of colors to show the significant enriched ontology terms.

`relative.color` determine whether the most significant enriched node has the deepest color.

`fontsize` the font size of ontology terms

`colorMap` the color map to show the significant enriched ontology terms

`selectionMethod` the method to select ontology nodes included in the plot graph `omitNode` whether to omit the intermediate insignificant ontology terms between the significant terms and terms with direct gene mapping

`saveImageName`
the name of image to be save. If it is NULL, then the image will not be saved.

Value

Invisibly return a list, which include following elements:

`graph` a graphNEL object includes the ontology subgraph to plot

`layoutAttrs` layoutAttrs used by the Rgraphviz function `layoutGraph`

`layoutNodeAttrs`
layoutNodeAttrs used by the Rgraphviz function `layoutGraph`

`layoutEdgeAttrs`
layoutEdgeAttrs used by the Rgraphviz function `layoutGraph`

`nodeRenderAttrs`
nodeRenderAttrs used by the graph function `nodeRenderInfo`

`edgeRenderAttrs`
edgeRenderAttrs used by the graph function `edgeRenderInfo`

These invisibly return information can be used as the input of function `plotGraph`.

Author(s)

Pan DU

See Also

See Also `plotGraph` and `plotGeneFunSummary`

Examples

```
data(DO)
geneSummary <- geneFunSummarize('4267', gene2DO.map, DO.graph.closure.gene)[[1]]
pValue <- sapply(geneSummary$sigOntoInfo, function(x) x$pValue)
plotOntologyGraph(pValue, geneSummary$allEvidence, DO.graph.gene, onto.graph.closure=DO.g
```

`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 date frame is a part of expression profile from a group of human Illumina array experiments.

References

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

```
data(sampleGroupsData)
head(sampleGroupsData)
```

`saveGeneFunSummary` *save the summarized gene function as a tab-separated text file*

Description

save the summarized gene function as a tab-separated text file

Usage

```
saveGeneFunSummary(geneFunSummarizeResult, simplifyInfo = NULL, addFDR = FALSE,
```

Arguments

`geneFunSummarizeResult`
the output of `geneFunSummarize` function

`simplifyInfo` the output of `simplifyGeneFunSummary` function

`addFDR` whether add FDR information or not

`species` the species of the related ontology under test, which is used to convert the Entrez Gene ID as gene symbols

`ID2Name` a named vector of ontology IDs to ontology terms mapping

`fileName` the file name to keep the summarized gene information

Value

Invisibly return a data.frame of summarized gene function, which includes "geneID", "geneSymbol", "bestOntology", "enrichedOntology" and "allEvidence". The corresponding annotation scores are kept in the parenthesis behind the ontology IDs.

Author(s)

Pan DU

See Also

See also `geneFunSummarize`

Examples

```
data(DO)
geneSummary <- geneFunSummarize(c("5037","9314"), gene2DO.map, DO.graph.closure.gene)
summaryInfo <- saveGeneFunSummary(geneSummary, fileName='geneSummarization.xls')
summaryInfo
```

---

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

simplifyGeneFunSummary *Simplify the significant ontology terms to a miniSet annotation*

Description

Simplify the significant ontology terms to a mini-set, which includes the non-overlapping most significant terms and some other ontology terms, which have direct gene mapping but not included in the significant ontology terms.

Usage

```
simplifyGeneFunSummary(geneFunSummarizeTestInfo, Onto.graph.closure, allOntoID.d
```

Arguments

`geneFunSummarizeTestInfo`
: the output of `geneFunSummarize` function

`Onto.graph.closure`
: a graphNEL object, whose edges represent the link between a ontology term and its offspring ontology terms

`allOntoID.direct`
: a vector of ontology IDs which has direct gene association. If not NULL, it will be used to filter the significant ontology terms

`fdr.adjust`
: the FDR estimation methods used to estimate the FDR of function enrichment

`p.value.th`
: the p-value threshold used to determine the significance of function enrichment

Value

The function return is a list with the same length of input "geneFunSummarizeTestInfo". Each element of the list is the simplified summarization of a gene. The simplified gene summarization is also a list with the following items

`keptSigOntoID`
: the significant ontology terms kept in the miniSet annotation

`keptEvidences`
: the non significant ontology terms which has direct gene mapping but are not the offspring of the ontology terms in keptSigOntoID

`scores`
: the annotation scores of all ontology terms in the miniSet annotation

The gene summarization object also includes the attributes of "pValueT".

Author(s)

Pan Du, Simon Lin, Gilbert Feng, Warren Kibbe

References

Pan Du, Simon Lin, Gilbert Feng, Warren Kibbe, "GeneRIFcompendiate: Ranked gene annotations using collective GeneRIF associations and ontology terms", under review

See Also

See Also `plotGeneFunSummary` and `geneFunSummarize`

Examples

```
data(DO)
## test all ontology terms related with gene 643387
geneSummary <- geneFunSummarize('643387', gene2DO.map, DO.graph.closure.gene, fdr.adjust
## the p.values of all related ontology terms
pValue <- sapply(geneSummary[[1]]$sigOntoInfo, function(x) x$pValue)
pValue
## simplify the annotation as a miniSet annotaiton
geneSummary.sim <- simplifyGeneFunSummary(geneSummary, DO.graph.closure.gene, p.value.th=
geneSummary.sim
```

`topcaBIO.PATH` *Present top CABIO.PATH enrichment test information*

Description

Function to present top CABIO.PATH enichmentInfo of given GeneAnswers instance.

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategory`

Examples

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

topCategoryGenes *Present top enrichment test information with genes*

Description

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

Usage

```
topCategoryGenes(inputX, orderby = c("geneNum", "pvalue", "foldChange", "oddsRat
```

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 speicify 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

Examples

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

`topCategory` *Present top enrichment test information*

Description

Function to present top enichmentInfo of given GeneAnswers instance.

Usage

```
topCategory(inputX, orderby = c("geneNum", "pvalue", "foldChange", "oddsRatio",
```

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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of 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')
```

topDOLiteGenes *Present top DOLITE enrichment test information with genes*

Description

Function to present top DOLITE enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., ’A collection of bioconductor methods to visualize gene-list annotations’, BMC Research Notes 2010, 3:10

See Also

`topCategoryGenes`

Examples

```
##x is a GeneAnswers instance with DOLite test
## Not run: topDOLiteGenes(x, geneSymbol=TRUE, orderby='pvalue', top=10, topGenes='ALL',
```

topDOLite *Present top DOLITE enrichment test information*

Description

Function to present top DOLITE enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategory`

Examples

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

topGOGenes — *Present top GO enrichment test information with genes*

Description

Function to present top GO enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategoryGenes`

Examples

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

`topGO` *Present top GO enrichment test information*

Description

Function to present top GO enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategory`

Examples

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

`topPATHGenes` *Present top KEGG enrichment test information with genes*

Description

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

Usage

```
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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

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

Examples

```
##x is a GeneAnswers instance with KEGG test
## Not run: topPATHGenes(x, geneSymbol=TRUE, orderby='genenum', top=6, topGenes=8, genesC
```

topPATH *Present top KEGG enrichment test information*

Description

Function to present top KEGG enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategory`

Examples

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

topREACTOME.PATHGenes *Present top REACTOME.PATH enrichment test information with genes*

Description

Function to present top REACTOME.PATH enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

`topCategoryGenes`

Examples

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

topREACTOME.PATH *Present top REACTOME.PATH enrichment test information*

Description

Function to present top REACTOME.PATH enichmentInfo 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

Feng, G., Du, P., Krett, N., Tessel, M., Rosen, S., Kibbe, W.A. and Lin, S.M., 'A collection of bioconductor methods to visualize gene-list annotations', BMC Research Notes 2010, 3:10

See Also

topCategory

Examples

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

Index

∗Topic **IO**
topcaBIO.PATH, 56
topcaBIO.PATHGenes, 4
topCategory, 58
topCategoryGenes, 57
topDOLite, 60
topDOLiteGenes, 59
topGO, 62
topGOGenes, 61
topPATH, 64
topPATHGenes, 63
topREACTOME.PATH, 66
topREACTOME.PATHGenes, 65
∗Topic **classes**
GeneAnswers-class, 15
∗Topic **datasets**
DO, 8
DOLite, 7
DOLiteTerm, 7
humanExpr, 46
humanGeneInput, 46
mouseExpr, 47
mouseGeneInput, 48
sampleGroupsData, 52
∗Topic **hplot**
plotGeneFunSummary, 48
plotGraph, 50
plotOntologyGraph, 51
∗Topic **methods**
buildNet, 1
caBIO2entrez, 3
categoryNet, 5
chartPlots, 6
drawTable, 9
entrez2caBIO, 10
geneAnnotationHeatmap, 11
geneAnswersBuilder, 12
geneAnswersChartPlots, 14
geneAnswersConceptNet, 17
geneAnswersConceptRelation, 18
geneAnswersConcepts, 19
geneAnswersHeatmap, 20
geneAnswersHomoMapping, 21
geneAnswersReadable, 23
geneAnswersSort, 24
geneConceptNet, 25
geneFunSummarize, 26
getcaBIOPATHList, 27
getcaBIOPATHTerms, 28
getCategoryList, 29
getCategoryTerms, 30
getConceptTable, 31
getConnectedGraph, 32
getDOLiteTerms, 33
getGOList, 34
getHomoGeneIDs, 35
getMultiLayerGraphIDs, 36
getNextGOIDs, 37
getPATHList, 38
getPATHTerms, 39
getREACTOMEPATHList, 39
getREACTOMEPATHTerms, 40
getSingleLayerGraphIDs, 41
getSymbols, 42
getTotalGeneNumber, 43
groupReport, 44
plotGeneFunSummary, 48
plotGraph, 50
plotOntologyGraph, 51
saveGeneFunSummary, 53
searchEntrez, 54
simplifyGeneFunSummary, 55
∗Topic **package**
GeneAnswers-package, 22

buildNet, 1, *33*

caBIO2entrez, 3
categoryNet, 5, *20*
chartPlots, 6, *14*
class:GeneAnswers
(GeneAnswers-class), 15

DO, 8
DO2gene.map *(DO)*, 8
DOLite, 7

DOLiteTerm, 7
drawTable, 9, *45*

edgeRenderInfo, *50*, *52*
entrez2caBIO, 10

gene2DO.map *(DO)*, 8
geneAnnotationHeatmap, 11, *21*
GeneAnswers
  (GeneAnswers-package), 22
GeneAnswers-class, *24*
GeneAnswers-class, 15
GeneAnswers-package, 22
geneAnswersBuilder, 12, *15, 16*, *31*, *44*
geneAnswersChartPlots, 14
geneAnswersConceptNet, 17
geneAnswersConceptRelation, 18
geneAnswersConcepts, 19
geneAnswersHeatmap, 20
geneAnswersHomoMapping, 21
geneAnswersReadable, 23
geneAnswersSort, 24
geneConceptNet, *17, 18*, 25
geneFunSummarize, 26, *48, 49*, *53*, *55, 56*
getAnnLib *(GeneAnswers-class)*, 15
getAnnLib,GeneAnswers-method
  (GeneAnswers-class), 15
getcaBIOPATHList, 27
getcaBIOPATHTerms, 28
getCategoryList, *28*, 29, *35*, *38*, *40*
getCategoryTerms, *3*, *23, 24*, 30, *34*, *39*
getCategoryType
  (GeneAnswers-class), 15
getCategoryType,GeneAnswers-method
  (GeneAnswers-class), 15
getConceptTable, *9, 10*, 31, *44, 45*
getConnectedGraph, *19*, 32
getDOLiteTerms, 33
getEnrichmentInfo
  (GeneAnswers-class), 15
getEnrichmentInfo,GeneAnswers-method
  (GeneAnswers-class), 15
getGeneExprProfile
  (GeneAnswers-class), 15
getGeneExprProfile,GeneAnswers-method
  (GeneAnswers-class), 15
getGeneInput *(GeneAnswers-class)*,
  15
getGeneInput,GeneAnswers-method
  (GeneAnswers-class), 15
getGenesInCategory
  (GeneAnswers-class), 15
getGenesInCategory,GeneAnswers-method
  (GeneAnswers-class), 15
getGOList, *12*, 34
getHomoGeneIDs, *22*, 35, *43*
getMultiLayerGraphIDs, *18*, 36
getNextGOIDs, 37
getPATHList, 38
getPATHTerms, 39
getPValueT *(GeneAnswers-class)*, 15
getPValueT,GeneAnswers-method
  (GeneAnswers-class), 15
getREACTOMEPATHList, 39
getREACTOMEPATHTerms, 40
getSingleLayerGraphIDs, *37*, 41
getSymbols, *24*, 42
getTestType *(GeneAnswers-class)*,
  15
getTestType,GeneAnswers-method
  (GeneAnswers-class), 15
getTotalGeneNumber, *13*, 43
groupReport, *10*, 44

help, *6*
humanExpr, 46
humanGeneInput, 46

layoutGraph, *50*, *52*

mouseExpr, 47
mouseGeneInput, 48

nodeRenderInfo, *50*, *52*

par, *9*
plotGeneFunSummary, *27*, 48, *52*, *56*
plotGraph, *49*, 50, *50*, *52*
plotOntologyGraph, *49, 50*, 51

sampleGroupsData, 52
saveGeneFunSummary, 53
searchEntrez, 54
setAnnLib *(GeneAnswers-class)*, 15
setAnnLib,GeneAnswers-method
  (GeneAnswers-class), 15
setCategoryType
  (GeneAnswers-class), 15
setCategoryType,GeneAnswers-method
  (GeneAnswers-class), 15
setEnrichmentInfo
  (GeneAnswers-class), 15
setEnrichmentInfo,GeneAnswers-method
  (GeneAnswers-class), 15
setGeneExprProfile
  (GeneAnswers-class), 15

setGeneExprProfile,GeneAnswers-method
(GeneAnswers-class), 15
setGeneInput *(GeneAnswers-class)*,
15
setGeneInput,GeneAnswers-method
(GeneAnswers-class), 15
setGenesInCategory
(GeneAnswers-class), 15
setGenesInCategory,GeneAnswers-method
(GeneAnswers-class), 15
setPValueT *(GeneAnswers-class)*, 15
setPValueT,GeneAnswers-method
(GeneAnswers-class), 15
setTestType *(GeneAnswers-class)*,
15
setTestType,GeneAnswers-method
(GeneAnswers-class), 15
show *(GeneAnswers-class)*, 15
show,GeneAnswers-method
(GeneAnswers-class), 15
simplifyGeneFunSummary, *27*, *53*, 55
summary *(GeneAnswers-class)*, 15
summary,GeneAnswers-method
(GeneAnswers-class), 15

topcaBIO.PATH, 56
topcaBIO.PATHGenes, 4
topCategory, *56*, 58, *60*, *62*, *64*, *66*
topCategoryGenes, *5*, 57, *59*, *61*, *63*, *65*
topDOLite, 60
topDOLiteGenes, 59
topGO, 62
topGOGenes, 61
topPATH, 64
topPATHGenes, 63
topREACTOME.PATH, 66
topREACTOME.PATHGenes, 65