GeneAnswers

April 19, 2010

`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

~put references to the literature/web site here ~

See Also

~~objects to See Also as `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

~put references to the literature/web site here ~

See Also

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

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]
```

---

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
`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.
`...`	other parameters used by .heatmap.mds

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

See Also

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

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.
`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`	logical indicating if FDR correction of the enrichment test p-value is performed or not.
`verbose`	logical, display current building stage.
`sortBy`	sorted type
`...`	additional arguments passed on to the corresponding enrichment test.

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.Hs.eg.db’, ’org.Mm.eg.db’, ’org.Rn.eg.db’ and ’org.Dm.eg.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". "GO.BP" only test biological process Gene Ontology terms, "GO.CC" for cellular components, "GO.MF" for molecular functions, and "GO" for all of these three categories. For user provided annotation library, it should be NULL in most cases.

totalGeneNumber could be NULL if annotationLib is one of ’org.Hs.eg.db’(45384), ’org.Mm.eg.db’(61498), ’org.Rn.eg.db’(37536) and ’org.Dm.eg.db’(22606). If user has own annotationLib, totalGeneNumber should be an integer, or one of ’human’, ’mouse’, ’rat’ and ’fly’. geneAnswersBuilder will automatically assign the corresponding value to totalGeneNumber.

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

~put references to the literature/web site here ~

See Also

~~objects to See Also as `geneAnswersSort`, `geneAnswersConceptNet`, ~~~

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

~put references to the literature/web site here ~

See Also

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

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.

`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.

`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

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("geneNum", "p
```

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
`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

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.

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

~put references to the literature/web site here ~

See Also

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

Examples

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

`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

~put references to the literature/web site here ~

See Also

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

Examples

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

`geneAnswersHeatmap` *Generate Concept-Gene Tabulates*

Description

A function to generate specified Concept-Gene Tablulates

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
`...`	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

~put references to the literature/web site here ~

See Also

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

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

See Also

~~objects to See Also as `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 given genes biological or medical interpretation. It includes statistical test of given genes and specified categories.

Details

Package:	GeneAnswers
Type:	Package
Version:	1.0
Date:	2009-07-16
License:	LGPL version 2 or newer

Author(s)

Gang Feng, Pan Du 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., BioConductor Methods to Visualize Genelist Annotations (Submitted)

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.

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.

Value

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

Author(s)

Gang Feng, Pan Du and Simon Lin

References

~put references to the literature/web site here ~

See Also

~~objects to See Also as `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

~put references to the literature/web site here ~

See Also

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

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, inputValue = NULL, centroidSize = "geneNum", output =
```

Arguments

`inputList`	a character list to generate concept-gene network. Names of the list are concepts.
`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

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. Genes with positive values will be represented by red solid circles, while green nodes stand for negative values gene nodes. 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.

Value

a concept-gene network is generated.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

~put references to the literature/web site here ~

See Also

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

Examples

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

`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

~put references to the literature/web site here ~

See Also

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

Examples

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

`getCategoryTerms` *Convert Category IDs to Terms*

Description

Function to convert category IDs to category terms.

Usage

```
getCategoryTerms(catIDs, catType, strict = FALSE, missing=c('name', 'keep', 're
```

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.

Details

The current version only supports 'GO', 'DOLite' and 'KEGG'. 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

~put references to the literature/web site here ~

See Also

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

Examples

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

`getDOLiteTerms` *Get DOLite Terms of Given DOLite IDs*

Description

function to convert 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

~put references to the literature/web site here ~

See Also

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

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

~put references to the literature/web site here ~

See Also

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

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

~put references to the literature/web site here ~

See Also

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

Examples

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

`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

~put references to the literature/web site here ~

See Also

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

Examples

```
a <- getPATHList(c('56458', '16590'), 'org.Mm.eg.db')
length(a)
```

`getPATHTerms` *Get Pathway names of given KEGG IDs*

Description

Function to convert given KEGG IDs to Pathway names.

Usage

```
getPATHTerms(pathIDs)
```

Arguments

`pathIDs`	a KEGG IDs vector

Details

~~ If necessary, more details than the description above ~~

Value

return a KEGG pathway terms of given KEGG IDs.

Author(s)

Gang Feng, Pan Du and Simon Lin

References

~put references to the literature/web site here ~

See Also

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

Examples

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

`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

~put references to the literature/web site here ~

See Also

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

Examples

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

`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.

Source

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

References

~~ possibly secondary sources and usages ~~

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.

Source

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

References

~~ possibly secondary sources and usages ~~

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.

Source

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

References

~~ possibly secondary sources and usages ~~

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.

Source

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

References

~~ possibly secondary sources and usages ~~

Examples

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

`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.

Details

~~ If necessary, more details than the description above ~~

Value

an Entrez ID list containing all of relative genes from Entrez database.

Author(s)

Pan Du, Gang Feng and Simon Lin

References

~put references to the literature/web site here ~

See Also

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

Examples

```
tagList <- list(FSHR=c('"FSHR"', '"Follicle stimulating hormone receptor"'), apoptosis=c(
entrezList <- searchEntrez(tagList, species='mouse')
```

`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.

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

~put references to the literature/web site here ~

See Also

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

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

~put references to the literature/web site here ~

See Also

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

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, geneSymbol = TRUE, ...)
```

Arguments

`x`	a given GeneAnswers instance with DOLite test
`catTerm`	logic value to determine whether mapping DOLite IDs to DOLite terms
`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

~put references to the literature/web site here ~

See Also

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

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

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

```
topDOLite(x, catTerm = TRUE, ...)
```

Arguments

`x`	a given GeneAnswers instance containing DOLite information
`catTerm`	logic value to determine whether mapping to DOLite terms 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

~put references to the literature/web site here ~

See Also

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

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, geneSymbol = TRUE, ...)
```

Arguments

`x`	a given GeneAnswers instance with GO test
`catTerm`	logic value to determine whether mapping GO IDs to GO terms
`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

~put references to the literature/web site here ~

See Also

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

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, ...)
```

Arguments

`x`	a given GeneAnswers instance containing GO test information
`catTerm`	logic value to determine whether mapping to GO terms 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

~put references to the literature/web site here ~

See Also

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

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, geneSymbol = TRUE, ...)
```

Arguments

`x`	a given GeneAnswers instance with KEGG test
`catTerm`	logic value to determine whether mapping KEGG IDs to KEGG terms
`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

~put references to the literature/web site here ~

See Also

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

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

~~ A concise (1-5 lines) description of what the function does. ~~

Usage

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

Arguments

`x`	a given GeneAnswers instance containing KEGG information
`catTerm`	logic value to determine whether mapping to DOLite terms or not
`...`	other parameters to transfer to topCategory

Details

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

Value

~Describe the value returned If it is a LIST, use

`comp1`	Description of 'comp1'
`comp2`	Description of 'comp2'

...

Author(s)

Gang Feng, Pan Du and Simon Lin

References

~put references to the literature/web site here ~

See Also

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

Examples

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

Index