Package 'PloGO2'

June 27, 2022

Type Package

Title Plot Gene Ontology and KEGG pathway Annotation and Abundance

Version 1.9.0

Description Functions for enrichment analysis and plotting gene ontology or KEGG pathway information for multiple data subsets at the same time. It also enables encorporating multiple conditions and abundance data.

Author Dana Pascovici, Jemma Wu

Maintainer Jemma Wu <jemma.wu@mq.edu.au>, Dana Pascovici <dana.pascovici@mq.edu.au>

Depends R (>= 4.0), GO.db, GOstats

Imports lattice, httr, openxlsx, xtable

biocViews Annotation, Clustering, GO, GeneSetEnrichment, KEGG, MultipleComparison, Pathways, Software, Visualization

License GPL-2

git_url https://git.bioconductor.org/packages/PloGO2

git_branch master

git_last_commit 3794a66

git_last_commit_date 2022-04-26

Date/Publication 2022-06-27

R topics documented:

`abundancePlot` *Function to summarize and plot abundance information from an annotations results list*

Description

Generates one GO/pathway abundance plot for each file, and (provided the number of GO/pathway cateGO/pathwayries is reasonably small) also an abundance plot for each GO/pathway category across all files provided. The abundance values are *added* for all values in one category. Hence if the initial values represented percentages such as NSAF, the final values represent percentages of the respective category.

Usage

```
abundancePlot(res.list, log = FALSE, printLimit = 16, Group=NULL, Plot=FALSE,
CountCutOff=3, ...)
```

Arguments

`res.list`	The result returned by processAnnotation
`log`	TRUE/FALSE: use raw or log abundance data
`printLimit`	The most number of category to be plotted
`Group`	The groups
`Plot`	To plot or not
`CountCutOff`	The minimum number of proteins in a category for it to be plotted
`...`	Parameters to pass

Value

A list object, with the following values:

`abundance`	The abundance matrix
`ag.mat`	The aggregated abundance matrix
`list.levelplots`	A list of abundance levelplots trellis object
`list.barplots`	A list of abundance barcharts trellis object

Author(s)

D.Pascovici and J.Wu

See Also

See Also as processAnnotation

Examples

```
# get list of ID's
GOIDlist <- GOTermList("MF", 2)

# find existing files
path <- system.file("files", package="PloGO2")
file.names <- paste(path, c("00100.txt", "01111.txt", "10000.txt",
"11111.txt","Control.txt"), sep="/")
datafile <- file.path(path, "NSAF.csv")

# summarize annotation
res.list <- processAnnotation(file.names, GOIDlist, data.file.name = datafile)

abundance.res <- abundancePlot(res.list)

# Plot levelplots
list.levelplots <- abundance.res$list.levelplots

for(i in seq_along(list.levelplots)) {
png(paste(names(list.levelplots)[i], ".png"), 2000, 4000, res=200)
print(list.levelplots[[i]])
dev.off()
}


# KEGG pathway
path <- system.file("files", package="PloGO2")
file.names <- file.path(path,"PWFiles", c("AllData.txt","black.txt","blue.txt","brown.txt","green.txt",
"red.txt","turquoise.txt") )
datafile <- file.path(path,"Abundance_data.csv")
Group <- names(read.csv(datafile))[-1]

AnnotIDlist <- c("osa01100","osa01110","osa01230","osa00300","osa00860")

res.list <- processAnnotation(file.names, AnnotIDlist,  data.file.name = datafile)

abundance.res <- abundancePlot(res.list, Group=Group, Plot=TRUE)

# Plot levelplots
list.levelplots <- abundance.res$list.levelplots

for(i in seq_along(list.levelplots)) {
png(paste(names(list.levelplots)[i], ".png"), 2000, 4000, res=200)
print(list.levelplots[[i]])
```

```
dev.off()
}
# Plot barchats by categories
list.barplots <- abundance.res$list.barplots

for(i in seq_along(list.barplots)) {
png(paste(names(list.barplots)[i], ".png"), 2000, 2000, res=200)
print(list.barplots[[i]])
dev.off()

}
```

---

annotationPlot *Function to summarize and plot extracted GO or pathway annotation*

Description

Summarize the information from the GO or pathway annotation list into a table of counts and percentages, and possibly print a few images.

Usage

```
annotationPlot(res.list, percentages = FALSE, plot = TRUE, trimzero = FALSE, type=c("GO", "pathway"))
```

Arguments

res.list	The list of summarized annotation as generated by processAnnotation.
percentages	TRUE or FALSE.
plot	TRUE or FALSE: should plots be printed.
trimzero	TRUE or FALSE: should GO categories with no counts be removed. This is not really relevant when a small number of GO categories has been selected.
type	The type of annotation.

Value

counts	A matrix of counts, GO categories (rows) by samples (cols).
percentages	A matrix of percentages, GO categories (rows) by samples (cols).

Author(s)

D. Pascovici

See Also

See Also processAnnotation

Examples

```
## Not run:
# get list of ID's
GOIDlist <- GOTermList("BP", 2)

# find existing files
dir <- system.file("files", package="PloGO2")
file.names <- paste(dir,c("00100.txt", "01111.txt", "10000.txt",
"11111.txt","Control.txt"), sep="/")

# summarize annotation
res.list <- processAnnotation(file.names, GOIDlist)

annotationPlot(res.list, plot=FALSE)

## End(Not run)
# KEGG pathway
dir <- system.file("files", package="PloGO2")
fname <- file.path(dir,"PWFiles", c("red.txt", "blue.txt", "yellow.txt", "green.txt", "turquoise.txt") )
datafile <- file.path(dir,"Abundance_data.csv")

AnnotIDlist <- c("osa01100","osa01110","osa01230","osa00300","osa00860")

res.list <- processAnnotation(fname, AnnotIDlist,  data.file.name = datafile)

annotationPlot(res.list, plot=TRUE, type="pathway")
```

`compareAnnot` *Function to compare annotation percentages*

Description

Compare annotation percentages by means of Fisher's exact test. A reference must be selected, and that name must be amongst the annotation result list names.

Usage

```
compareAnnot(res.list, referenceName, removeZeros = FALSE, correction = TRUE)
```

Arguments

`res.list`	List returned by the processAnnotation function
`referenceName`	Name of the condition to compare with
`removeZeros`	Remove the categories with no annotation in them from the result
`correction`	TRUE or FALSE: apply the BH fdr correction to the p-values in each column

Value

A matrix of p-values or NA. A comparison is made for each sample other than the reference with the reference, and each GO annotation category. The result is not recorded if the annotation numbers were small for that category (<5). This is strictly not needed for Fisher's exact test (though was needed for the chi-square approximation used initially).

Author(s)

D.Pascovici

See Also

See Also processAnnotation

Examples

```
## Not run:
# get list of ID's
GOIDlist <- GOTermList("BP", 2)

# find existing files
dir <- system.file("files", package="PloGO2")
file.names <- paste(dir,c("00100.txt", "01111.txt", "10000.txt",
"11111.txt","Control.txt"), sep="/")

# summarize annotation
res.list <- processAnnotation(file.names, GOIDlist)

# compare with Control
compareAnnot(res.list, "Control")

compareAnnot(res.list, "Control", correction=FALSE)

## End(Not run)
```

ExcelToPloGO *Function to add GO annotation to an Excel spread sheet*

Description

Function to add GO annotation to an Excel spread sheet

Usage

```
ExcelToPloGO(fname, colName = "Uniprot", termFile= NA, compareWithReference="none", data.file.name = "
```

Arguments

fname	Name of the Excel spread sheet to be annotated
colName	The name of the column containing the protein identifiers
termFile	The name of the file containing the GO categories
compareWithReference	The name of the tab that serves as basis for enrichement comparison.
data.file.name	Abundance data for PloGO if any
outFolder	The output files folder

Value

A list object, with the following values:

Counts	The GO counts matrix summarized for all files
Percentages	The GO percentages matrix summarized for all files
Abundance	The GO percentages matrix summarized for all files, ONLY generated if an abundance file was provided
FisherPval	The Fisher p-values matrix summarized for all files, ONLY generated if a reference file was provided
res.list	The full list result of processAnnotation function
list.levelplots	The abundance levelplots if the data.file.name is not "none"
list.barplots	The abundance barcharts if the data.file.name is not "none"

Author(s)

D.Pascovici

Examples

```
## Not run:
# where sample files are stored
path <- system.file("files", package = "PloGO2")
termFile = paste(path, "GODefault.txt", sep="/")
xlfile <- paste(path, "ResultsWGCNA_Input4PloGO2.xlsx", sep="/")

res <- ExcelToPloGO(xlfile, termFile=termFile, compareWithReference="AllData")



## End(Not run)
```

`ExcelToPloPathway` *Function to add KEGG pathway annotation to an Excel spread sheet*

Description

Function to add KEGG pathway annotation to an Excel spread sheet

Usage

```
ExcelToPloPathway(fname, colName = "Uniprot", compareWithReference = "none", DB.name = "pathwayDB.csv"
```

Arguments

`fname`	Name of the Excel spread sheet to be annotated
`colName`	The name of the column containing the protein identifiers
`compareWithReference`	The name of the tab that serves as basis for enrichment comparison.
`DB.name`	The DB file name for the pathway
`data.file.name`	Abundance data for PloGO if any
`outFolder`	The output files folder

Value

A list object, with the following values:

`Counts`	The pathway counts matrix summarized for all files
`Percentages`	The pathway percentages matrix summarized for all files
`Abundance`	The pathway percentages matrix summarized for all files, ONLY generated if an abundance file was provided
`aggAbundance`	Aggregated abundance matrix by combination of tab and abundance file columns for all pathways
`FisherPval`	The Fisher p-values matrix summarized for all files, ONLY generated if a reference file was provided
`res.list`	The full list result of processAnnotation function
`list.levelplots`	The abundance levelplots if the data.file.name is not "none"
`list.barplots`	The abundance barcharts if the data.file.name is not "none"

Examples

```
path <- system.file("files", package = "PloGO2")

res <- ExcelToPloPathway(file.path(path, "ResultsWGCNA_Input4PloGO2.xlsx"),
colName="Uniprot", compareWithReference="AllData", DB.name=file.path(path, "pathwayDB.csv"),
data.file.name = file.path(path, "Abundance_data.csv") )
```

genAnnotationFiles *Function to generate a format in Wego native style from a list of Uniprot identifiers*

Description

Given a Excel spreadsheet with multiple tabs, generate a Wego file for each tab using a pre-download DB file.

Usage

```
genAnnotationFiles(fExcelName,  colName="Uniprot",
DB.name = "pathwayDB.csv", folder="PWFiles",outFolder=tempdir())
```

Arguments

fExcelName	An excel file containing one or multiple tabs of protein IDs. For example, the proteins in each tab come from the same cluster.
colName	The column name of the protein ID.
DB.name	The database file name, in .csv format.
folder	The folder name for saving the generated files.
outFolder	The output files folder

Value

The folder path for generated annotation files.

Author(s)

J. Wu

Examples

```
path <- system.file("files", package = "PloGO2")

genAnnotationFiles(fExcelName = file.path(path, "ResultsWGCNA_Input4PloGO2.xlsx"),
colName="Uniprot",
DB.name = file.path(path, "pathwayDB.csv"))
```

genWegoFile *Function to generate a format in Wego native style from a list of Uniprot identifiers*

Description

Connects to the Uniprot Biomart using functionality from the biomaRt package, downloads GO information and organizes it as needed.

Usage

```
genWegoFile(IDList, fname = "Wego.txt", database = "uniprot", outFolder=tempdir())
```

Arguments

IDList	A list of ID's, either Ensembl or Uniprot.
fname	The name of the text file to be outputed.
database	"ensembl" or "uniprot".
outFolder	The output files folder

Value

File path of generated Wego files

Author(s)

D. Pascovici

Examples

```
v <- c("Q9HWC9","Q9HWD0","Q9I4N8","Q9HW18","Q9HWC9","Q9HWD0")

## Not run: genWegoFile(v, fname = "F1.txt")
```

getGoID *Function to map a vector of GO terms to the corresponding GO ID's*

Description

This function is rather slow and inefficient, as it first extracts all GO terms then matches the current one. However, it in only intended for a one off matching of terms of interest to the respective nodes.

Usage

```
getGoID(v)
```

Arguments

v	the precise spelling of the go term

Value

The list of matched terms, with names the respective GO ID's/

Author(s)

D. Pascovici

Examples

```
getGoID(c("biological_process", "transport"))

# however the next one is not found as the proper term name has an
# underscore "cellular_component"
getGoID("cellular component")
```

GOTermList *Function to extract a list of GO terms at level 2,3 or 4 of the GO hierarchy*

Description

At the moment a quick and dirty way to extract all GO nodes at levels 2, 3 or 4 of the GO hierarchy.

Usage

```
GOTermList(ontology = "BP", level = 2, node = NULL)
```

Arguments

ontology	Either "BP", "CC" or "MF"
level	Either 2, 3 or 4
node	Null or a GO node

Details

This is a quick and dirty way to extract a list of GO terms of interest. That can be one of the levels 2, 3 or 4 or all subnodes (children) of a particular node if a valid GO is provided for the node parameter. Should be rewritten.

Value

A vector of GO nodes

Author(s)

D. Pascovici

Examples

```
GOTermList("BP", 2)
GOTermList("CC", 2)
```

`PloGO` *Function to do steps of GO annotation and annotation/abundance plots*

Description

Summarize GO categories for all the files in the zip, if provided merge data from data file, generate annotation and abundance plots and comparison with reference.

Usage

```
PloGO(zipFile = "none", termFile="none", ontology = "BP", ontologyLevel = 2, reference = "none", data.fi
filesPath=".", node=NULL, aggregateFun="sum", logAb=FALSE, ...)
```

Arguments

`zipFile`	Zip containing all the GO files
`termFile`	A file with the GO terms of interest, if a limited set provided
`ontology`	Wither BP, MF or CC
`ontologyLevel`	A small level such as 2 or 3
`reference`	The file name of the reference file, if any, for instance "Control" for Control.txt
`data.file.name`	The file containing all the experimental data, for instance "NASF.csv"
`datafile.ignore.cols`	The number of columns in the experimental file given in data.file.name that should not be used as numerical, for instance 2 if there is an ID field and a Description field. By default 1.
`filesPath`	If the zip file is not provided, the path to the GO files
`node`	NULL, ignored at the moment
`aggregateFun`	Either "sum" or "product"; the aggregation operation for abundance data
`logAb`	TRUE or FALSE; the abundance data to be logged or not
`...`	Parameters to pass

Details

Process all the GO files provided, in the directory or the zip, and assign ID's to the respective categories. The categories can come from a target list, or from a choice of level and ontology. If a reference is provided, then the numbers of identifiers in each category is compared to the reference by means of Fisher's exact test. If a data file is provided, then the quantitative values are aggregated into the sets.

Value

A list object, with the following values:

`Counts`	The GO counts matrix summarized for all files
`Percentages`	The GO percentages matrix summarized for all files
`Abundance`	The GO percentages matrix summarized for all files, ONLY generated if an abundance file was provided
`aggregatedAbundance`	The aggregated abundance matrix if the data file is provided
`FisherPval`	The Fisher p-values matrix summarized for all files, ONLY generated if a reference file was provided
`res.list`	The full list result of processAnnotation function
`list.levelplots`	The abundance levelplots if the data.file.name is not "none"
`list.barplots`	The abundance barcharts if the data.file.name is not "none"

Author(s)

D. Pascovici

See Also

`processAnnotation`

Examples

```
# where sample files are stored
path <- system.file("files", package = "PloGO2")
# run PloGO with list of id's, data file and reference
res <- PloGO( zipFile=paste(path, "GOfiles.zip", sep="/"),
reference="Control", termFile = paste(path, "GOListDrought.txt", sep="/"),
data.file.name = paste(path, "NSAFDesc.csv", sep="/"),
datafile.ignore.cols = 2)
```

PloPathway *Function to do steps of pathway annotation and annotation/abundance plots*

Description

Summarize pathway categories for all the files in the zip, if provided merge data from data file, generate annotation and abundance plots and comparison with reference.

Usage

```
PloPathway(zipFile = "none", reference = "none", data.file.name = "none", datafile.ignore.cols = 1, fil
```

Arguments

zipFile	Zip containing all the GO files
reference	The file name of the reference file, if any, for instance "Control" for Control.txt
data.file.name	The file containing all the experimental data, for instance "NASF.csv"
datafile.ignore.cols	The number of columns in the experimental file given in data.file.name that should not be used as numerical, for instance 2 if there is an ID field and a Description field. By default 1.
filesPath	If the zip file is not provided, the path to the GO files
aggregateFun	Either "sum" or "product"; the aggregation operation for abundance data
logAb	TRUE or FALSE; the abundance data to be logged or not
...	Parameters to pass

Details

Process all the pathway files provided, in the directory or the zip, and assign ID's to the respective categories. If a reference is provided, then the numbers of identifiers in each category is compared to the reference by means of Fisher's exact test. If a data file is provided, then the quantitative values are aggregated into the sets.

Value

A list object, with the following values:

Counts	The GO counts matrix summarized for all files
Percentages	The GO percentages matrix summarized for all files
Abundance	The GO percentages matrix summarized for all files, ONLY generated if an abundance file was provided
aggregatedAbundance	The aggregated abundance matrix if the data file is provided

FisherPval	The Fisher p-values matrix summarized for all files, ONLY generated if a reference file was provided
res.list	The full list result of processAnnotation function
list.levelplots	The abundance levelplots if the data.file.name is not "none"
list.barplots	The abundance barcharts if the data.file.name is not "none"

Author(s)

J. Wu

See Also

processAnnotation

Examples

```
# where sample files are stored
path <- system.file("files", package = "PloGO2")
# run PloPathway with list of id's, data file and reference
res <- PloPathway( zipFile=paste(path, "PWFiles.zip", sep="/"),
reference="Alldata",
data.file.name = paste(path, "Abundance_data.csv", sep="/"),
datafile.ignore.cols = 1)
```

---

plotAbundanceBar *Function to plot abundance barplot*

Description

Plot the aggregated abundance barplot for all files/clusters

Usage

```
plotAbundanceBar(mat.abundance, mat.counts, min.count=5)
```

Arguments

mat.abundance	A matrix of abundance
mat.counts	A matrix of the counts
min.count	The cutoff for the minimum counts to be included

Value

None returned, generate a barplot.

Author(s)

J.Wu

Examples

```
path <- system.file("files", package = "PloGO2")
# run PloPathway with list of id's, data file and reference
res <- PloPathway( zipFile=paste(path, "PWFiles.zip", sep="/"),
reference="Alldata",
data.file.name = paste(path, "Abundance_data.csv", sep="/"),
datafile.ignore.cols = 1)

png("AbundanceBarplot.png", 2500, 2000, res=300)
par(mar=c(4,10,4,14))

plot.res <- plotAbundanceBar(res$aggregatedAbundance, res$Counts)

dev.off()
```

---

printSummary *Function to print the summary file of PloGO2 results.*

Description

Each tab in the summary spreadsheet of the

Usage

```
printSummary(results, file="PloGO2Results.xlsx")
```

Arguments

results	A list of results from PloGO2 analysis
file	The output file name

Value

TRUE if at least some annotations were found, FALSE otherwise.

Author(s)

J. Wu

Examples

```
## Not run:
path <- system.file("files", package = "PloGO2")
# run PloPathway with list of id's, data file and reference
res <- PloPathway( zipFile=paste(path, "PWFiles.zip", sep="/"),
reference="Alldata",
data.file.name = paste(path, "Abundance_data.csv", sep="/"),
datafile.ignore.cols = 1)

printSummary(res)

## End(Not run)
```

`processAnnotation` *Function to process a set of annotation files given a list of GO/pathway identifiers of interest*

Description

For each file in the list extract all identifiers that belong to each GO or pathway category in the list of identifiers. An identifier "belongs" to a GO or pathway category if it is annotated at the category itself or any of its children in the GO graph.

Usage

```
processAnnotation(file.list, AnnotIDlist, data.file.name = NULL, printFiles = FALSE,
format = c("compact","long"), datafile.ignore.cols = 1, aggregateFun="sum")
```

Arguments

`file.list`	A list of files
`AnnotIDlist`	A list of GO/pathway identifiers
`data.file.name`	A list containing additional data such as abundance information
`printFiles`	TRUE/FALSE If true an annotation summary file is generated for each input file
`format`	One of "compact" by default or or "long", the format of the annotation files. See details.
`datafile.ignore.cols`	How many columns in the abundance file to ignore. By default assume the first only, containing identifiers.
`aggregateFun`	Either "sum" or "product"; the aggregation operation for abundance data

Details

The format is for the GO/pathway files is "compact" by default, meaning a text file containing ID's in the first column, and GO/pathway identifiers in the second, separated by spaces or semicolons. This is the same as the "Wego native format". A "long" format is also accepted, meaning a text file with two or more columns separated by spaces, containing an identifier, followed by a GO/pathway id, followed optionally by other columns which are ignored. The GO/pathway id's will first be aggregated for each identifier.

Value

A list of the same length as the list of files submitted. Each list element is the result of processGoFile or processPathFile.

Author(s)

D. Pascovici, J.Wu

See Also

processGoFile, processPathFile

Examples

```
# For GO analysis
# choose two simple GO categories
termList <- c("response to stimulus", "transport")
GOIDmap <- getGoID(termList)
GOIDlist <- names(GOIDmap)


path <- system.file("files", package = "PloGO2")
file.names <- file.path(path, c("00100.txt", "01111.txt", "10000.txt",
"11111.txt", "Control.txt"))

# summarize annotation
res.list <- processAnnotation(file.names, GOIDlist)

# files in long format
longFormat <- file.path(path,  "goRetOutput.txt")
processAnnotation(longFormat, GOIDlist, format = "long")

# For KEGG pathway analysis
fname <- file.path(path,"PWFiles", "red.txt")
datafile <- file.path(path,"Abundance_data.csv")

AnnotIDlist <- c("osa01100","osa01110","osa01230","osa00300","osa00860")

res.list <- processAnnotation(fname, AnnotIDlist,  data.file.name = datafile)
```

processGoFile *Function to process a single file in wego-like native format*

Description

For each GO category of interest extract all ID's from the file that are annotated at either the category or its GO children. If abundance data is present extract and merge.

Usage

```
processGoFile(fname, GOIDlist, datafile = NULL, datafile.ignore.cols = 1, format = c("compact","long"),
```

Arguments

fname	The GO file name, in either Wego native format or long format
GOIDlist	The list of GO id's of interest
datafile	The file containing abundance or NULL if none.
datafile.ignore.cols	How many columns in the abundance file to ignore. By default assume the first only, containing identifiers.
format	Either "compact" or "long"; see details
aggregateFun	Either "sum" or "product"; the aggregation operation for abundance data

Details

The format is "compact" by default, meaning a text file containing ID's in the first column, and GO identifiers in the second, separated by spaces or semicolons. This is the same as the "Wego native format". A "long" format is also accepted, meaning a text file with two or more columns separated by spaces, containing an identifier, followed by a GO id, followed optionally by other columns which are ignored. The GO id's will first be aggregated for each identifier. The output of GOretriever can be used as "long" format.

Value

A list with the following components

counts	A vector of the same length as the list of GO id's of interest giving the number of ID's in each category
ID.list	The list of ID's for each GO category
datafile	The abundance datafile provided passed through
abundance	A matrix with as many rows as the GO list provided, and as many columns as the abundance data file
N	The number of protein (gene etc) identifiers in each file
fname	The filename without the file path

Author(s)

D. Pascovici

Examples

```
termList <- c("response to stimulus", "transport")
GOIDmap <- getGoID(termList)
GOIDlist <- names(GOIDmap)

# use one of the stored files
dir <- system.file("files", package="PloGO2")
fname <- paste(dir,"00100.txt", sep="/")
datafile <- paste(dir, "NSAF.csv", sep="/")


# or if abundance in present aggregate that by category
processGoFile(fname, GOIDlist, datafile=datafile)
```

---

processPathFile *Function to process a single pathway file in wego-like native format*

Description

For each pathway extract all ID's from the file. If abundance data is present extract and merge.

Usage

```
processPathFile(fname, AnnotIDlist, datafile=NULL, datafile.ignore.cols=1,
format=c("compact","long"), aggregateFun="sum")
```

Arguments

fname	The pathway file name, in either Wego native format or long format
AnnotIDlist	The list of pathway annotation ID
datafile	The file containing abundance or NULL if none.
datafile.ignore.cols	How many columns in the abundance file to ignore. By default assume the first only, containing identifiers.
format	Either "compact" or "long"; see details
aggregateFun	The aggregation function for abundance data

Details

The format is "compact" by default, meaning a text file containing ID's in the first column, and pathway identifiers in the second, separated by spaces or semicolons. This is the same as the "Wego native format". A "long" format is also accepted, meaning a text file with two or more columns separated by spaces, containing an identifier, followed by a pathway id, followed optionally by other columns which are ignored. The pathway id's will first be aggregated for each identifier.

Value

A list with the following components

counts	A vector of the same length as the list of pathway id's of interest giving the number of ID's in each category
ID.list	The list of ID's for each pathway category
datafile	The abundance datafile provided passed through
abundance	A matrix with as many rows as the pathway list provided, and as many columns as the abundance data file
N	The number of protein (gene etc) identifiers in each file
fname	The filename without the file path

Author(s)

J. Wu

Examples

```
# use one of the stored files
dir <- system.file("files", package="PloGO2")
fname <- paste(dir,"PWFiles/AllData.txt", sep="/")
datafile <- paste(dir, "Abundance_data.csv", sep="/")
AnnotIDlist <- unique(unlist(sapply(read.delim(fname, stringsAsFactors=FALSE)[,2], function(x) strsplit(x, split


# or if abundance in present aggregate that by category
processPathFile(fname, AnnotIDlist, datafile=datafile)
```

---

read.annot.file *Function to read an annotation file.*

Description

Accepts GO id's separated by space or semicolon

Usage

```
read.annot.file(fname, format = c("compact","long"))
```

Arguments

fname	The file name containing GO annotation
format	Either "compact" or "long"; see details

Details

The format is "compact" by default, meaning a text file containing ID's in the first column, and GO identifiers in the second, separated by spaces or semicolons. This is the same as the "Wego native format". A "long" format is also accepted, meaning a text file with two or more columns separated by tabs, containing an identifier, followed by a GO id, followed optionally by other columns which are ignored. The GO id's will first be aggregated for each identifier. GO files in long format can be obtained using for instance biomart, or GoRetriever.

Value

A data frame with two columns, ID's and GO separated by spaces

Author(s)

T. Keighley, D.Pascovici

Examples

```
# use one of the stored files
dir <- system.file("files", package="PloGO2")
fname <- paste(dir,"00100.txt", sep="/")

# Example with GoRetriever download
longFormat <- paste(dir,"goRetOutput.txt", sep="/")
read.annot.file(fname)
read.annot.file(longFormat, format="long")

# Example with biomart download
biomartDownload <- paste(dir,"mart_export.txt", sep="/")
read.annot.file(biomartDownload, format="long")
```

---

writeAnnotation *Function to print GO/pathway annotation to files*

Description

Prints available GO or pathway annotation and abundance (if existing) in a long format or an adjacency matrix type format.

Usage

```
writeAnnotation(res.list, datafile = NULL, datafile.ignore.cols = 1, format = c("list","matrix"),outFo
```

Arguments

`res.list`	The result of processAnnotation
`datafile`	A CSV file with additional experimental information, if any
`datafile.ignore.cols`	The number of columns to ignore in the data file
`format`	Either `"matrix"` or `"list"`.
`outFolder`	The output files folder

Details

The GO/pathway information and abundance will be printed to files. If the format is `"list"`, then the files will be text files, and each category will be printed in turn, with all the identifiers and data underneath. If the format is `"matrix"`, then the data will be printed in matrix format, identifiers (rows) by GO categories (columns), with the abundance data appended.

Value

The path of the annotation folder.

Author(s)

D. Pascovici

See Also

`processAnnotation`

Examples

```
# choose two simple GO categories
termList <- c("response to stimulus", "transport", "signaling")
GOIDmap <- getGoID(termList)
GOIDlist <- names(GOIDmap)


dir <- system.file("files", package="PloGO2")
file.names <- paste(dir,c("00100.txt", "01111.txt", "10000.txt",
"11111.txt","Control.txt"), sep="/")

# summarize annotation
res.list <- processAnnotation(file.names, GOIDlist)

# write to "matrix" or alternatively "list" format
writeAnnotation(res.list, format="matrix")
```

Index