Package 'MEGENA'

July 21, 2025

Type Package

Title Multiscale Clustering of Geometrical Network

Version 1.3.7

Date 2018-08-30

Author Won-Min Song, Bin Zhang

Maintainer Won-Min Song <wonmin1984@gmail.com>

Description Co-Expression Network Analysis by adopting network embedding technique. Song W.-M., Zhang B. (2015) Multiscale Embedded Gene Co-expression Network Analysis. PLoS Comput Biol 11(11): e1004574. <doi:10.1371/journal.pcbi.1004574>.

URL https://github.com/songw01/MEGENA

License GPL (>= 3)

Imports Rcpp (>= 0.11.3),Matrix (>= 1.1-5),ggplot2 (>= 1.0.0),reshape (>= 0.8.5),fpc (>= 2.1-11),cluster (>= 2.0.7-1),ggrepel (>= 0.5), ggraph (>= 1.0.1)

Depends R (>= 3.4.0),doParallel (>= 1.0.11),foreach (>= 1.4.4),igraph (>= 1.2.1)

LinkingTo Rcpp,BH

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-09-10 19:00:03 UTC

Contents

`MEGENA-package` *co-expression network analysis*

Description

construction of gene-gene interaction network and dissection into multi-scale functional modules, and network key drivers.

Details

Package:	MEGENA
Type:	Package
Version:	1.3
Date:	2015-12-18
License:	GPL (>= 2)

Multiscale Embedded Gene Co-expression Network Analysis (MEGENA)

Author(s)

Won-Min Song

Maintainer: Won-Min Song <won-min.song@mssm.edu>

References

Song W-M, Zhang B (2015) Multiscale Embedded Gene Co-expression Network Analysis. PLoS Comput Biol 11(11): e1004574. doi: 10.1371/journal.pcbi.1004574

`calculate.correlation` *correlation calculation*

Description

correlation analysis with FDR calculation

Usage

```
calculate.correlation(datExpr,doPerm = 100,doPar = FALSE,num.cores = 8,method = "pearson",
FDR.cutoff = 0.05,n.increment = 100,is.signed = FALSE,
output.permFDR = TRUE,output.corTable = TRUE,saveto = NULL)
```

Arguments

`datExpr`	gene expression data matrix
`doPerm`	Number of permutations to perform. If `doPerm = NULL`, calculates BH FDR p-values instead of permutation based FDR.
`doPar`	TRUE/FALSE logical variable to choose parallelization. Parallelization is utilized when BH FDR p-values are calculated for all pairs.
`num.cores`	number of cores to use in parallelization.
`method`	correlation method to be passed to `cor` for `method` argument.
`FDR.cutoff`	FDR threshold to output final results of significant correlations.
`n.increment`	When permutation is utilized, 0 <= |rho| <= 1 is broken down into n.increment to map each |rho| cutoff to respective FDR.
`is.signed`	TRUE/FALSE to indicate using signed/unsigned correlation.
`output.permFDR, output.corTable`	TRUE/FALSE to choose to output permutation indices and FDR table.
`saveto`	folder to output results.

Details

If `doPar = TRUE`, then num.cores are registered for PCP.

Value

output is three column edgelist data.frame, third column being the weight.

Author(s)

Won-Min Song

Examples

```
# test simplest case of planar network (a 3-clique).
data(Sample_Expression)
calculate.correlation(datExpr[1:100,],doPerm = 5)
```

`calculate.PFN` *PFN calculation*

Description

main function to calculate PFN a ranked list of edge pairs

Usage

```
calculate.PFN(edgelist,max.skipEdges = NULL,maxENum = NULL,doPar = FALSE,
num.cores = NULL,keep.track = TRUE)
```

Arguments

`edgelist`	three column edgelist: first two columns are topological edges, and the third column is the weight. Must be a data.frame object.
`max.skipEdges`	Maximum number of edges to be searched by planarity test without any inclusion to PFN. If set NULL, it will be automatically set to number of cores x 1000. It acts as a threhold to quicken PFN construction termination during PCP.
`maxENum`	maximum number of edges to include in final PFN. Default value is NULL, which invokes maximal number of edges allowed in planar network.
`doPar`	TRUE/FALSE logical variable to choose parallelization.
`num.cores`	number of cores to use in parallelization.
`keep.track`	If TRUE, pfg_el.RData will be created in working folder. This file can be used later for restart in case PFN construction did not finish successfully. Default is TRUE.

Details

If `doPar = TRUE`, then num.cores are registered for PCP.

Value

output is three column edgelist data.frame, third column being the weight.

Author(s)

Won-Min Song

Examples

```
# test simplest case of planar network (a 3-clique).
a <- c(1,1,2);b <- c(2,3,3);w <- runif(3,0,1);
el <- cbind(a,b,w);el <- as.data.frame(el[order(el[,3],decreasing = TRUE),])
calculate.PFN(edgelist = el,max.skipEdges = Inf,doPar = FALSE,num.cores = NULL)
```

compute.PFN.par *Parallelized PFN computation*

Description

PFN construction by parallelized edge screening.

Usage

```
compute.PFN.par(sortedEdge,Ng,maxENum,Njob,Ncore,max.skipEdges = NULL,
keep.track = TRUE,initial.links = NULL)
```

Arguments

sortedEdge	3-column matrix for the input edgelist (e.g. - correlation pair list). Must be sorted by third column, which is usually weight vector.
Ng	integer. number of genes included in sortedEdge.
maxENum	Maximum number of edges to include in final PFN. The theoretical maximal number enforced by Euler's formula is 3(Ng-2).
max.skipEdges	Maximum number of edges to be counted before any valid edge to be included in PFN. This works as a termination condition to avoid exhaustive planarity testing over all edges provided in sortedEdge.
Njob	Number of edges to be passed to each core for parallelized edge screening.
Ncore	Number of cores to utilize.
keep.track	TRUE/FALSE logical. Indicate if the record of PFN construction is saved in temporary file "pfg_el.RData". Default is TRUE.
initial.links	If provided, PFN construction will restart by regarding these initial.links as already-built PFN.

Details

This is parallelized implementation of PFN construction, where it is possible to re-capture PFN construction by providing already computed edgelist into initial.links. Although provivded, this function itself may require careful caution and users are encouraged to use more user-friendly "calculate.PFN()" instead.

Value

A 3-column matrices, where first two columns are integer indices for vertices, and third is the weight vector.

Author(s)

Won-Min Song

datExpr *Toy example data*

Description

A portion of TCGA breast cancer data to test run MEGENA.

Usage

```
Sample_Expression.RData
```

Format

Contains a matrix object, "datExpr". Use data(Sample_Expression) to load.

Details

a gene expression matrix.

References

1. Song, W.M. and B. Zhang, Multiscale Embedded Gene Co-expression Network Analysis. PLoS Comput Biol, 2015. 11(11): p. e1004574.

do.MEGENA *MEGENA clustering + MHA*

Description

multiscale clustering analysis (MCA) and multiscale hub analysis (MHA) pipeline

Usage

```
do.MEGENA(g,
do.hubAnalysis = TRUE,
mod.pval = 0.05,hub.pval = 0.05,remove.unsig = TRUE,
min.size = 10,max.size = 2500,
doPar = FALSE,num.cores = 4,n.perm = 100,singleton.size = 3,
save.output = FALSE)
```

Arguments

g	igraph object of PFN.
do.hubAnalysis	TRUE/FALSE indicating to perform multiscale hub analysis (MHA) in downstream. Default is TRUE.
mod.pval	cluster significance p-value threshold w.r.t random planar networks
hub.pval	hub significance p-value threshold w.r.t random planar networks
remove.unsig	TRUE/FALSE indicating to remove insignificant clusters in MHA.
min.size	minimum cluster size
max.size	maximum cluster size
doPar	TRUE/FALSE indicating parallelization usage
num.cores	number of cores to use in parallelization.
n.perm	number of permutations to calculate hub significance p-values/cluster significance p-values.
singleton.size	Minimum module size to regard as non-singleton module. Default is 3.
save.output	TRUE/FALSE to save outputs from each step of analysis

Details

Performs MCA and MHA by taking PFN as input. Returns a list object containing clustering outputs, hub analysis outputs, and node summary table.

Value

A series of output files are written in wkdir. Major outputs are,

module.output	outputs from MCA
hub.output	outputs from MHA
node.summary	node table summarizing clustering results.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)

## End(Not run)
```

draw_sunburst_wt_fill *Draw sunburst plot showing MEGENA module hierarchy.*

Description

Sunburst plot and colored heatmaps

Usage

```
draw_sunburst_wt_fill(module.df,
                      parent.col = "module.parent",id.col = "id",
                      min.angle = 5,
                      feat.col,
                      fill.type = "continuous",log.transform = TRUE,
                      fill.scale = NULL,
                      theme.adjust = NULL
                      )
```

Arguments

module.df	A data.frame table summarizing module information. Must contain module parent and child relation for hierarchy visualization.
parent.col	Character object, name for the parent module column in module.df.
id.col	Character object for the module id column in module df.
min.angle	Minimum angle that rectangles in the sunburst are labeled with respective module id.
feat.col	Chracter object, for the feature column in module.df to color the heatmaps.
fill.type	continuous/discrete, is the variable numeric (continuous) or factor (discrete)?
log.transform	TRUE/FALSE. do log10 transform for p-values?
fill.scale	A ggplot object to specify heatmap coloring scheme. Permissible functions are: scale_fill_gradient,scale_fill_gradient2,scale_fill_gradientn,scale_fill_manual.
theme.adjust	A ggplot object to specify theme for plotting.

Details

makes use of ggraph scheme to manipulate and draw sunburst plot in ggplot2 framework. fill.scale and theme.adjust provide flexibility to designate heatmap coloring schemes and figure aesthetics.

Value

ggplot object for the figure

Author(s)

Won-Min Song

Examples

```
## Not run:
  rm(list = ls())
  data(Sample_Expression)
  ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
  el <- calculate.PFN(ijw[,1:3])
  g <- graph.data.frame(el,directed = FALSE)
  MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
  output.summary <- MEGENA.ModuleSummary(MEGENA.output,
  mod.pvalue = 0.05,hub.pvalue = 0.05,
  min.size = 10,max.size = 5000,
  annot.table = NULL,id.col = NULL,symbol.col = NULL,
  output.sig = TRUE)

  # no coloring
  sbobj = draw_sunburst_wt_fill(module.df = output.summary$module.table,
  feat.col = NULL,id.col = "module.id",parent.col = "module.parent")
  sbobj

  # get some coloring (with log transform option)
  mdf= output.summary$module.table
  mdf$heat.pvalue = runif(nrow(mdf),0,0.1)

 sbobj = draw_sunburst_wt_fill(module.df = mdf,feat.col = "heat.pvalue",log.transform = TRUE,
  fill.type = "continuous",
  fill.scale = scale_fill_gradient2(low = "white",mid = "white",high = "red",
  midpoint = -log10(0.05),na.value = "white"),
  id.col = "module.id",parent.col = "module.parent")
  sbobj

  # get discrete coloring done
  mdf$category = factor(sample(x = c("A","B"),size = nrow(mdf),replace = TRUE))
  sbobj = draw_sunburst_wt_fill(module.df = mdf,feat.col = "category",
  fill.type = "discrete",
  fill.scale = scale_fill_manual(values = c("A" = "red","B" = "blue")),
  id.col = "module.id",parent.col = "module.parent")
  sbobj

## End(Not run)
```

`get.DegreeHubStatistic` *calculate module degree statistics based on random triangulation model via T1 and T2 moves.*

Description

calculation of module p-values.

Usage

```
get.DegreeHubStatistic(subnetwork,n.perm = 100,doPar = FALSE,n.core = 4)
```

Arguments

subnetwork	a planar network as an igraph object.
n.perm	number of random networks generated, constraint with number of links and nodes same to "subnetwork".
doPar	TRUE/FALSE to parallelize.
n.core	number of cores/threads to use.

Details

Hub significance calculation functionality. Make sure that, if doPar = TRUE, register cores using registerDoParallel() from doParallel package.

Value

a data.frame table showing node-wise statistics.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)

out <- get.DegreeHubStatistic(subnetwork = g,n.perm = 100,doPar = FALSE,n.core = 4)

## End(Not run)
```

---

get.hub.summary *summarize hub information.*

Description

hubs in different scales are summarized.

Usage

```
get.hub.summary(MEGENA.output)
```

Arguments

MEGENA.output	A list object. The output from "do.MEGENA()".

Details

returns a data.frame object

Value

A data.frame object with columns:

node	hub gene node names
S1, ...	binary vector indicating hubs in each scale
frequency	number of scales that respective gene emerges as hub.
scale.summary	list of scales that respective gene as hub.

Author(s)

Won-Min Song

---

get.union.cut *Scale-thresholding of multiscale modules.*

Description

obtain a discrete, disjoint clustering results from multiscale MEGENA modules for a given alpha value.

Usage

```
get.union.cut(module.output,alpha.cut,output.plot = T,
plotfname = "validModules_alpha",module.pval = 0.05,remove.unsig = T)
```

Arguments

module.output	A direct output from "do.MEGENA". (i.e. MEGENA.output$module.output).
alpha.cut	Resolution parameter cut-off (i.e. alpha) value. alpha.cut = 1 corresponds to classical definition of "small-world" compactness.
output.plot	TRUE/FALSE to indicate outputting a .png file showing hierarchical structure with final outputted modules highlighted in red.
plotfname	.png file outputname.
module.pval	module significance p-value.
remove.unsig	TRUE/FALSE indicating to remove insignificant clusters.

Details

Returns a list object where each entry is a module.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
get.union.cut(module.output = MEGENA.output$module.output,alpha.cut = 1,
output.plot = FALSE,plotfname = NULL,module.pval = 0.05,remove.unsig = TRUE)

## End(Not run)
```

MEGENA.ModuleSummary *MEGENA module summary*

Description

Summarizes modules into a table.

Usage

```
MEGENA.ModuleSummary(MEGENA.output,
mod.pvalue = 0.05,hub.pvalue = 0.05,
min.size = 10,max.size = 2500,
annot.table = NULL,symbol.col = NULL,id.col = NULL,
output.sig = TRUE)
```

Arguments

MEGENA.output	A list object. The output from "do.MEGENA()".
mod.pvalue	module compactness significance p-value, to identify modules with significant compactness.
hub.pvalue	node degree significance p-value to identify nodes with significantly high degree.
min.size	minimum module size allowed to finalize in the summary output.
max.size	maximum module size allowed to finalize in the summary output.

annot.table	Default value is NULL, indicating no mapping is provided between node names to gene symbols. If provided, the mapping between node names (id.col) and gene symbol (symbol.col) are used.
id.col	column index of annot.table for node names.
symbol.col	column index of annot.table for gene symbols.
output.sig	Default value is TRUE, indicating significant modules are outputted.

Details

output$module.table contains many important information including module hierarchy, as indicated by

Value

A list object with the components:

modules	Final set of modules obtained upon apply mod.pvalue for significance, min.size and max.size for module size thresholding.
mapped.modules	gene symbol mapped modules when "annot.table" is provided.
module.table	data.frame object for module summary table. Columns include: id, module.size, module.parent, module.hub, module.scale and module.pvalue.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
output.summary <- MEGENA.ModuleSummary(MEGENA.output,
mod.pvalue = 0.05,hub.pvalue = 0.05,
min.size = 10,max.size = 5000,
annot.table = NULL,id.col = NULL,symbol.col = NULL,
output.sig = TRUE)

## End(Not run)
```

`module_convert_to_table` *conversion of module list object to a data.frame table format*

Description

Summarizes module hub/hierarchy/membership into a data.frame table format.

Usage

```
module_convert_to_table(MEGENA.output,mod.pval = 0.05,
hub.pval = 0.05,min.size = 10,max.size)
```

Arguments

`MEGENA.output`	A list object. The output from "do.MEGENA()".
`mod.pval`	module compactness significance p-value, to identify modules with significant compactness.
`hub.pval`	node degree significance p-value to identify nodes with significantly high degree.
`min.size`	minimum module size allowed to finalize in the summary output.
`max.size`	maximum module size allowed to finalize in the summary output.

Details

the resulting data.frame contains the following essential columns: id, module.parent and module. If the co-expression network bears significant hubs, it will additionally have node.degree (connectivity), node.strength (sum of edge weights) and is.hub column to supplement hub information.

Value

A data.frame with the columns:

`id`	gene name
`module.parent`	parent module id
`module`	module name.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
output.summary <- MEGENA.ModuleSummary(MEGENA.output,
mod.pvalue = 0.05,hub.pvalue = 0.05,
min.size = 10,max.size = 5000,
annot.table = NULL,id.col = NULL,symbol.col = NULL,
output.sig = TRUE)
module.df = module_convert_to_table(MEGENA.output,mod.pval = 0.05,
hub.pval = 0.05,min.size = 10,max.size)
head(module.df)

## End(Not run)
```

---

output.geneSet.file *output gene signatures into .gmt file format*

Description

An interface function to output .gmt format gene signature file.

Usage

```
output.geneSet.file(geneSet,outputfname)
```

Arguments

geneSet	a list object
outputfname	output file name

Details

Outputs each signature into a single line of lists in outputfname.

Author(s)

Won-Min Song

`planaritytest` *Boyer-Myvold Planarity test of a network*

Description

wrapper function of _MEGENA_planaritytest. imports from Boost graph library, and test planarity of a network

Usage

```
planaritytest(N, rows, cols)
```

Arguments

`N`	must be an integer. number of nodes in the network.
`rows`	first column of edgelist. a vector of integers.
`cols`	second column of edgelist. a vector of integers.

Details

cbind(rows,cols) is equivalent to the two column edge list of the network. We assume that the network is undirected.

Value

TRUE/FALSE is returned to indicate planarity. (TRUE -> network is planar).

Author(s)

Won-Min Song

Examples

```
# test simplest case of planar network (a 3-clique).
planaritytest(as.integer(3),c(1,1,2),c(2,3,3))
```

plot_module *Module plotting function.*

Description

Extract subnetworks for modules and plot.

Usage

```
plot_module(output.summary,PFN,subset.module = NULL,col.names,
gene.set = NULL,color.code = "logFC",show.legend = TRUE,
label.hubs.only = TRUE,hubLabel.col = "red",hubLabel.sizeProp = 0.5,show.topn.hubs = 10,
node.sizeProp = 13,label.sizeProp = 13,label.scaleFactor = 10,label.alpha = 0.5,
layout = "kamada.kawai",output.plot = TRUE,out.dir = "modulePlot")
```

Arguments

output.summary	output from summary function, "MEGENA.ModuleSummary".
PFN	igraph object retaining PFN topology.
subset.module	A character vector for list of module names to plot. Default = NULL plots all modules in output.summary.
col.names	a character vector for list of colors to be used for coloring children modules.
gene.set	A list object containing signatures for customized coloring of nodes in resulting network plot.
color.code	A character vector with matched length to "gene.set", to specify colors for each signature.
label.hubs.only	TRUE/FALSE to show labels for significant hub genes only, or all genes. Defauly is TRUE.
hubLabel.col	Label color for hubs. Default is "red"
show.legend	TRUE/FALSE for showing node legend on the bottom of the figure.
hubLabel.sizeProp	A multiplicative factor to adjust hub label sizes with respect to node size values. Default is 0.5
show.topn.hubs	Maximal number of hubs to label on module subnetwork. Default is 10.
node.sizeProp	A multiplicative factor to adjust node sizes with respect to 90th percentile degree node size. Default is 13
label.sizeProp	A multiplicative factor to adjust node label sizes with respect to 90th percentile degree node size. Default is 13
label.scaleFactor	Overall scale factor to control the final size of node labels appearing in figure. Default is 10.
label.alpha	Transparency value ranging from 0 (transparent) to 1 (solid). Default is 0.5.

layout	Network layout algorithm to apply. Options are: "kamada.kawai", "fruchterman.reingold".
output.plot	logical value. output.plot = TRUE generates figure files under folder, "modulePlot".
out.dir	if output.plot = TRUE, then out.dir is created and resulting figures are exported to .png files to the folder.

Details

Subnetwork plot functionality with application of "ggrepel" package for node labeling. The most effective way to control overall node label size is through label.scaleFactor.

Value

A list object holding ggplot objects for plotted modules.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
library(MEGENA)

data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
output.summary <- MEGENA.ModuleSummary(MEGENA.output,
mod.pvalue = 0.05,hub.pvalue = 0.05,
min.size = 10,max.size = 5000,
annot.table = NULL,id.col = NULL,symbol.col = NULL,
output.sig = TRUE)

pnet.obj <- plot_module(output = output.summary,PFN = g,subset.module = "comp1_2",
layout = "kamada.kawai",label.hubs.only = FALSE,
gene.set = list("hub.set" = c("CD3E","CD2")),color.code =  c("red"),
output.plot = FALSE,out.dir = "modulePlot",col.names = c("grey","grey","grey"),
hubLabel.col = "black",hubLabel.sizeProp = 1,show.topn.hubs = Inf,show.legend = TRUE)

pnet.obj

## End(Not run)
```

plot_module_hierarchy *Plot module hierarchy*

Description

visualized module hierarchical structure.

Usage

```
plot_module_hierarchy(module.table,plot.coord = NULL,
edge.color = "grey",node.color = "black",node.label.color = "black",
label.scaleFactor = 0.5,node.scaleFactor = 0.2,arrow.size = 0.015,
data.col = NULL,low.color = "blue",mid.color = "white",
high.color = "red",mid.value = 0.05)
```

Arguments

module.table	output from MEGENA.ModuleSummary. Specifically $module.table component of the output.
plot.coord	Two column coordinate matrix. rownames must be labelled according to module.table$id.
edge.color	Edge color to be shown.
node.color	If data.col = NULL, node.color is used to color nodes in figure.
node.label.color	Node label color.
label.scaleFactor	scale number to adjust node label sizes.
node.scaleFactor	scale number to adjust node sizes.
arrow.size	scale number to arrow size.
data.col	A character to specify data vector to color nodes in module.table.
low.color	If data.col != NULL, color to be used in lower value spectrum.
mid.color	If data.col != NULL, color to be used in middle value spectrum.
high.color	If data.col != NULL, color to be used in high value spectrum.
mid.value	If data.col != NULL, value to define middle value spectrum.

Details

Module hierarchy plotting functionality using ggplot2.

Value

A list containing output$hierarchy.obj = ggplot2 object, output$node.data = node attributes, output$edge.data = edge attributes.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
data(Sample_Expression)
ijw <- calculate.correlation(datExpr,doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
output.summary <- MEGENA.ModuleSummary(MEGENA.output,
mod.pvalue = 0.05,hub.pvalue = 0.05,
min.size = 10,max.size = 5000,
annot.table = NULL,id.col = NULL,symbol.col = NULL,
output.sig = TRUE)

module.table = output.summary$module.table
colnames(module.table)[1] <- "id"
output.obj <- plot_module_hierarchy(module.table = module.table,
label.scaleFactor = 0.15,arrow.size = 0.005,node.label.color = "blue")

print(output.obj[[1]])

## End(Not run)
```

---

plot_subgraph *subnetwork plotting functionality.*

Description

A modification of plot_module() function for more general subnetwork plotting purpose.

Usage

```
plot_subgraph(module,hub = NULL,PFN,node.default.color = "black",
gene.set = NULL,color.code = "grey",show.legend = TRUE,
label.hubs.only = TRUE,hubLabel.col = "red",hubLabel.sizeProp = 0.5,show.topn.hubs = 10,
node.sizeProp = 13,label.sizeProp = 13,label.scaleFactor = 10,layout = "kamada.kawai")
```

Arguments

module	A character vector containing gene names to be subsetted.
hub	If provided, genes in hub will be highlighted as triangles in resulting figure.
PFN	igraph object retaining PFN topology.
node.default.color	Default node colors for those that do not intersect with signatures in gene.set.

gene.set	A list object containing signatures for customized coloring of nodes in resulting network plot.
color.code	A character vector with matched length to "gene.set", to specify colors for each signature.
show.legend	TRUE/FALSE for showing node legend on the bottom of the figure.
label.hubs.only	TRUE/FALSE to show labels for significant hub genes only, or all genes. Defauly is TRUE.
hubLabel.col	Label color for hubs. Default is "red"
hubLabel.sizeProp	A multiplicative factor to adjust hub label sizes with respect to node size values. Default is 0.5
show.topn.hubs	Maximal number of hubs to label on module subnetwork. Default is 10.
node.sizeProp	A multiplicative factor to adjust node sizes with respect to 90th percentile degree node size. Default is 13
label.sizeProp	A multiplicative factor to adjust node label sizes with respect to 90th percentile degree node size. Default is 13
label.scaleFactor	Overall scale factor to control the final size of node labels appearing in figure. Default is 10.
layout	Network layout algorithm to apply. Options are: "kamada.kawai", "fruchterman.reingold".

Details

Subnetwork plot functionality with application of "ggrepel" package for node labeling. The most effective way to control overall node label size is through label.scaleFactor.

Value

A list object holding ggplot object and node annotation table.

Author(s)

Won-Min Song

Examples

```
## Not run:
rm(list = ls())
library(MEGENA)

data(Sample_Expression)
ijw <- calculate.correlation(datExpr[1:100,],doPerm = 2)
el <- calculate.PFN(ijw[,1:3])
g <- graph.data.frame(el,directed = FALSE)
MEGENA.output <- do.MEGENA(g = g,remove.unsig = FALSE,doPar = FALSE,n.perm = 10)
output.summary <- MEGENA.ModuleSummary(MEGENA.output,
```

```
mod.pvalue = 0.05,hub.pvalue = 0.05,
min.size = 10,max.size = 5000,
annot.table = NULL,id.col = NULL,symbol.col = NULL,
output.sig = TRUE)

pnet.obj <- plot_subgraph(module = output.summary$modules[[1]],
hub = c("CD3E","CD2"),PFN = g,node.default.color = "black",
gene.set = NULL,color.code = c("grey"),show.legend = TRUE,
label.hubs.only = TRUE,hubLabel.col = "red",hubLabel.sizeProp = 0.5,
show.topn.hubs = 10,node.sizeProp = 13,label.sizeProp = 13,
label.scaleFactor = 10,layout = "kamada.kawai")

# the plot
pnet.obj[[1]]

# the annotation
pnet.obj[[2]]

## End(Not run)
```

`read.geneSet` *.gmt file reader function*

Description

An interface function to read-in .gmt format gene signature file.

Usage

```
read.geneSet(geneSet.file)
```

Arguments

`geneSet.file`	text file containing gene signatures in .gmt format

Details

Each line of lists in `geneset.file` is a single set of signature.

Value

loads signatures into a list object.

Author(s)

Won-Min Song

`Sample_Expression` *Toy example data*

Description

A portion of TCGA breast cancer data to test run MEGENA.

Usage

```
data(Sample_Expression)
```

Format

Contains a matrix object, "datExpr". Use data(Sample_Expression) to load.

Details

a gene expression matrix.

References

1. Song, W.M. and B. Zhang, Multiscale Embedded Gene Co-expression Network Analysis. PLoS Comput Biol, 2015. 11(11): p. e1004574.

Index