Package 'LANDD'

October 1, 2016

Type Package

Title Liquid Association for Network Dynamics Detection

Version 1.1.0

Date 2016-9-30

Author Shangzhao Qiu, Yan Yan, Tianwei Yu

Maintainer Shangzhao Qiu <qsz1328@gmail.com>

Description Using Liquid Association for Network Dynamics Detection.

License GPL (>= 2)

Imports igraph, fdrtool, foreach, doParallel, GOstats, GOSemSim, GGally, intergraph, ggplot2, modeest, Matrix, Rcpp, mvtnorm, pROC

LinkingTo Rcpp,BH

Depends

RoxygenNote 5.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-10-01 01:14:24

R topics documented:

get.W *Record genes W*

Description

`get.W()` creates a table to record Gene Ontology Biological Process mapping results. Every gene x takes a row.

Usage

```
get.W(graph, laresult, z.matrix, cutoff, k = 2)
```

Arguments

`graph`	The graph of gene network.
`laresult`	The result of lascouting which finds the liquid association scouting genes.
`z.matrix`	A matrix representing gene Z (selected scouting genes). Row names are the gene id in gene network.
`cutoff`	The threshold to find LA scouting genes.
`k`	An Integer giving the order of the network.

Details

`get.W()` generates a result file of ego gene X, genes within k steps of X, the liquid association scouting genes of x and genes W.Every gene x takes a row in the table.

Value

A table records the intermediate result of Gene Ontology Biological Process which contains ego gene X, genes within k steps of X, the liquid association scouting genes of x and genes W.Each x occupies a row.

Examples

```
## Not run:
laresult <- lascouting(g,m,k=2,n.cores=4)
get.W(g,laresult,z,cutoff=0.8,k=2)
## End(Not run)
```

getgobp — *Create a table to record Gene Ontology Biological Process mapping results. Every gene W's community takes a row.*

Description

getgobp.community() generates a result file of ego gene X, significant GO terms of X, significant GO terms of genes within k steps of X, gene W, significant GO terms of W,

Usage

```
getgobp(graph, z.matrix, k = 2, n.cores = 4, cutoff = 1,
  community = TRUE, community.min = 5, term.limit = NA)
```

Arguments

graph	The graph of gene network.
z.matrix	A matrix representing gene Z (selected scouting genes). Row names are the gene id in gene network.
k	An Integer giving the order of the network.
n.cores	The number of cores used for parallel computing.
cutoff	The threshold to find LA scouting genes.
community	Boolean. Whether compute the community of genes W or not.
community.min	Integer. The minimum number of genes numbers in a community.
term.limit	The maximum number of GO terms to list in a row of the table.

Value

A table containing the IDs of scouting center genes W, over-represented GO terms by W, semantic similarity on the Gene Ontology system between the X ego network and all scouting center genes, average graph distance between gene X and W. W are grouped by network community. Each W community occupies a row.

Examples

```
## Not run:
g <- graph.data.frame(as.matrix(read.table("HumanBinaryHQ_HINT.txt")))
getgobp(g,z,k=2,n.cores=4,cutoff=1,community=TRUE,community.min=5,term.limit = NA)
## End(Not run)
```

graph.kd *Find weights based on kernel density on the graph. There are three common ways to invoke* graph.kd*:*

- graph.kd(relate_matrix, graph, smoothing.normalize=c('one'))
- graph.kd(relate_matrix, graph, smoothing.normalize=c('squareM'))
- graph.kd(relate_matrix, graph, smoothing.normalize=c('none'))

The first method is used when the total weight of all genes z is set to 'one'. In this way, those genes surrounded by more genes z will not take advantages over those surrounded by fewer genes. In contrast, the second method takes the number of genes around into consideration, the result of the first method will multiply the square of the number of genes around. The third method does not normalize the data. Thus genes with more neighbors are more likely to receive higher weights.

Description

Find weights based on kernel density on the graph.

There are three common ways to invoke graph.kd:

- graph.kd(relate_matrix, graph, smoothing.normalize=c('one'))
- graph.kd(relate_matrix, graph, smoothing.normalize=c('squareM'))
- graph.kd(relate_matrix, graph, smoothing.normalize=c('none'))

The first method is used when the total weight of all genes z is set to 'one'. In this way, those genes surrounded by more genes z will not take advantages over those surrounded by fewer genes. In contrast, the second method takes the number of genes around into consideration, the result of the first method will multiply the square of the number of genes around. The third method does not normalize the data. Thus genes with more neighbors are more likely to receive higher weights.

Usage

```
graph.kd(relate.matrix, network.graph, kernel.sd = 1,
  smoothing.normalize = c("one", "squareM", "none"))
```

Arguments

relate.matrix The matrix returned by lascouting.

network.graph The igraph object representing the gene network.

kernel.sd standard deviation for kernel, default would be 1.

smoothing.normalize
Different ways to normalize the result, default would be "one".

Value

A matrix representing the weights calculated using kernel density for each gene. Each row is an ego gene, columns are the weights of potential scouting genes for the gene.

Examples

```
## Not run:
relate.matrix <- lascouting(g,m,k=2,n.cores=4)
graph.kd(relate.matrix,g,smoothing.normalize = "one")
## End(Not run)
```

`lascouting` *Find the liquid association scouting genes.*

Description

`lascouting()` is used to find the liquid association scouting gene

Usage

```
lascouting(network.graph, express.matrix, k = 2, n.cores = 4)
```

Arguments

`network.graph`	An igraph object representing the gene network.
`express.matrix`	A matrix representing the expression matrix for the genes in gene network. Row names are the gene ids in gene network.
`k`	Integer giving the order of the ego-network.
`n.cores`	Number of cores used for parallel computing.

Value

A logical matrix representing the LA-scouting genes for each gene. Rows represent the ego gene id and columns represents the LA-scouting genes.

Examples

```
## Not run: laresult <- lascouting(g,m,k=2,n.cores=4)
```

`normalizeInputMatrix` *Normalize the input Matrix*

Description

Normalize the input Matrix

Usage

```
normalizeInputMatrix(x)
```

Arguments

`x`	A numeric matrix

`simulateLANDD` *Simulate LANDD*

Description

Simulate LANDD

Usage

```
simulateLANDD(rho, n.sample, z.percent, k, kernel.sd, normalize.method)
```

Arguments

`rho`	Strength of LA correlation
`n.sample`	sample number for LA
`z.percent`	percentage of z in sample
`k`	k step neighbour
`kernel.sd`	standard deviation for kernel
`normalize.method`	Different ways to normalize the result

Value

simulated data for LANDD

visualize *Visualize: Generate a graph which vividly displays the gene X, Y and W.*

Description

visualize() generates a graph. It is used to intuitively and vividly display the layout of gene X, Y and W.

Usage

```
visualize(graph, kernel.result, x, k = 2, cutoff = 1, path = NULL)
```

Arguments

graph	The igraph object of gene network.
kernel.result	The result of graph.kd which finds genes W of a gene X.
x	The gene the plot is generated for.
k	The degree of the neighborhood of X.
cutoff	A threshold to filter gene W.
path	The path where the result graph is saved to. The default path is the original path of input graph.

Value

a graph of gene X, Y and W

Examples

```
## Not run:
kernel <- graph.kd(relate.matrix,g,smoothing.normalize = "one")
visualize(g,kernel,x,k=2,cutoff=1,path= NULL)
## End(Not run)
```

visualize.community *visualize ego gene X, its k step neighbours, and the W gene communities: Generate a graph with different community in different colors. visualize.community()is used to create a graph to display the layout of genes X, X's k-step neighborhood, W and their corresponding community.*

Description

visualize ego gene X, its k step neighbours, and the W gene communities: Generate a graph with different community in different colors. visualize.community()is used to create a graph to display the layout of genes X, X's k-step neighborhood, W and their corresponding community.

Usage

```
visualize.community(graph, kernel.result, x, k = 2, cutoff = 1,
  community.min = 5, path = NULL)
```

Arguments

graph	The igraph object of gene network.
kernel.result	The result of graph.kd which finds genes W of a gene X.
x	The gene the plot is generated for.
k	The degree of the neighborhood of X.
cutoff	A threshold to filter gene W.
community.min	The minimum size of the community of W.
path	The path where the result graph is saved to. The default path is the original path of input graph.

Value

a graph displays genes X, X's k-step neighborhood, and W gene communities in different colors.

Examples

```
## Not run:
kernel <- graph.kd(relate.matrix,g,smoothing.normalize = "one")
visualize(g,kernel,x,k=2,cutoff=1,community.min=5,path=NULL)
## End(Not run)
```

xw.distance *Create a table to record the distance between gene x and gene w.*

Description

xw.distance() generates a table contains distance between all genes x and their correspongding genes w.

Usage

```
xw.distance(graph, z.matrix, cutoff = 0.8, n.cores = 4)
```

Arguments

graph	The graph of the gene network.
z.matrix	A matrix representing gene Z. Row names are the gene id in gene network.
cutoff	A number used to find LA scouting gene z.
n.cores	Core number used for parallel computing.

Value

a table contains distance between all genes x and their correspongding genes w.

Examples

```
## Not run:  xw.distance(g,m,cutoff=0.8,n.cores=4)
```

Index