Package 'netboxr'

March 30, 2021

Type Package

Title netboxr

Version 1.2.0

Date 2020-09-03

Author Eric Minwei Liu [aut,cre], Augustin Luna [aut], Ethan Cerami [aut]

Maintainer Eirc Minwei Liu <emliu.research@gmail.com>

Description NetBox is a network-based approach that combines prior knowledge with a network clustering algorithm. The algorithm allows for the identification of functional modules and allows for combining multiple data types, such as mutations and copy number alterations. NetBox performs network analysis on human interaction networks, and comes pre-loaded with a Human Interaction Network (HIN) derived from four literature curated data sources, including the Human Protein Reference Database (HPRD), Reactome, NCI-Nature Pathway Interaction (PID) Database, and the MSKCC Cancer Cell Map.

License LGPL-3 + file LICENSE

Depends R (>= 4.0.0), igraph (>= 1.2.4.1), parallel

Imports RColorBrewer, DT, stats, clusterProfiler, data.table, gplots, jsonlite, plyr

Suggests paxtoolsr, BiocStyle, org.Hs.eg.db, knitr, rmarkdown, testthat, cgdsr

VignetteBuilder knitr

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

biocViews Software,Network,Pathways,GraphAndNetwork,Reactome, SystemsBiology, GeneSetEnrichment, NetworkEnrichment, KEGG

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

git_branch RELEASE_3_12

git_last_commit 52f9bc4

git_last_commit_date 2020-10-27

Date/Publication 2021-03-29

R topics documented:

`annotateGraph` *Annotate NetBox graph*

Description

This function annotates the graph based on user input.

Usage

```
annotateGraph(
  netboxResults,
  edgeColors = NULL,
  directed = FALSE,
  linker = TRUE
)
```

Arguments

`netboxResults`	Output from `geneConnector` function. a list of returned netboxr results • netboxGraph: igraph object of NetBox algorithm identified network nodes and connections • netboxCommunity: igraph object of network community assignment • netboxOutput: data frame of NetBox algorithm identified network nodes and connections • nodeType: data frame of node types ("candidate" or "linker") in the NetBox algorithm indentified network. • moduleMembership: data frame of module (community) membership. • neighborData: data frame of information of nodes directly connected to candidate gene nodes.
`edgeColors`	table containing hex color codes for interaction types. The first column is interaction type and the second column is hex color code.
`directed`	boolean value indicating whether the NetBox algorithm identified network is directed or undirected (default = FALSE)
`linker`	boolean value indicating whether "linker" nodes exist in the NetBox algorithm identified network or not (default = TRUE)

Details

If a table of color codes for interaction types is provided, then the edges will be colored accordingly by interaction types. If `directed` is TRUE, then the edges will be arrows with the same directionality as the original input network for NetBox. If `linker` is TRUE, then linker nodes will be shown as squares while non-linker nodes stay as circles.

Value

annotated version of netboxGraph

Author(s)

Guanlan Dong, <guanlan_dong@g.harvard.edu>

Examples

```
data(pathway_commons_v8_reactome)
interaction_type_color <- read.csv(system.file("interaction_type.color.txt", package = "netboxr"),
                                   header=TRUE, sep="\t", stringsAsFactors=FALSE)

sifNetwork<-pathway_commons_v8_reactome$network
graphReduced <- networkSimplify(sifNetwork,directed = FALSE)

geneList <- pathway_commons_v8_reactome$geneList

results <- geneConnector(geneList = geneList, networkGraph = graphReduced,
                         directed = FALSE, pValueAdj = "BH", pValueCutoff = 2e-5,
                         communityMethod = "ebc", keepIsolatedNodes = FALSE)

netboxGraphAnnotated <- annotateGraph(netboxResults = results,
                                      edgeColors = interaction_type_color,
                                      directed = TRUE,
                                      linker = TRUE)

# As an example, plot both the original and the annotated graphs
ll <- layout_with_fr(results$netboxGraph) # Save the layout for easier comparison
# Plot original graph
pdf("originalGraph.pdf", width = 50, height = 50)
plot(results$netboxCommunity, results$netboxGraph, layout = ll,
     vertex.size=3)
dev.off()
# Plot annotated graph
pdf("annotatedGraph.pdf", width = 50, height = 50)
plot(results$netboxCommunity, netboxGraphAnnotated, layout = ll,
     vertex.size = 3,
     vertex.shape = V(netboxGraphAnnotated)$shape,
     edge.color = E(netboxGraphAnnotated)$interactionColor,
     edge.width = 3)
# Add legend
ind <- which(interaction_type_color$INTERACTION_TYPE %in% E(netboxGraphAnnotated)$interaction)
legend_interaction_type <- interaction_type_color$INTERACTION_TYPE[ind]
legend_interaction_type_color <- interaction_type_color$COLOR[ind]
legend(x=-1.1, y=1.1,
       legend=c("Candidate", "Linker"),
       pch=c(19, 15), # solid circle, filled square
       pt.cex = 8,
```

```
       bty="n",
       title="Node Types",
       cex=4, ncol=1)
legend(x=-1.15, y=0.95,
       legend=legend_interaction_type,
       col = legend_interaction_type_color,
       lty = 1, lwd = 10,
       bty="n",
       title="Interaction Types (Edges)",
       cex=4, ncol=1)
dev.off()
```

`geneConnector` *Generate sub-network mapping from a list of candidate genes*

Description

This function generates sub-network mapping from a list of candidate genes

Usage

```
geneConnector(
  geneList,
  networkGraph,
  directed = FALSE,
  pValueAdj = "BH",
  pValueCutoff = 0.05,
  communityMethod = "ebc",
  keepIsolatedNodes = FALSE
)
```

Arguments

`geneList`	character vector containing a list of candidate genes
`networkGraph`	igraph network graph object. This igraph object contains curated network information
`directed`	boolean value indicating whether the input network is directed or undirected (default = FALSE)
`pValueAdj`	string for p-value correction method c("BH", "Bonferroni") as described in the details section (default = "BH")
`pValueCutoff`	numeric value of p-value cutoff for linker nodes (default = 0.05)
`communityMethod`	string for community detection method c("ebc","lec") as described in the details section (default = "ebc")
`keepIsolatedNodes`	A boolean value indicating whether to keep isolated nodes in the netboxr result (default = FALSE)

Details

P-value correction methods include the Bonferroni correction ("bonferroni") or Benjamini & Hochberg ("BH"). Community detection methods include using edge betweeness score ("ebc") or using leading eigenvector method ("lec)

Value

a list of returned netboxr results

- netboxGraph: igraph object of NetBox algorithm identified network nodes and connections
- netboxCommunity: igraph object of network community assignment
- netboxOutput: data frame of NetBox algorithm identified network nodes and connections
- nodeType: data frame of node types ("candidate" or "linker") in the NetBox algorithm indentified network.
- moduleMembership: data frame of module (community) membership.
- neighborData: data frame of information of nodes directly connected to candidate gene nodes.

Author(s)

Eric Minwei Liu, <emliu.research@gmail.com>

Examples

```
data(netbox2010)

sifNetwork<-netbox2010$network
graphReduced <- networkSimplify(sifNetwork,directed = FALSE)

geneList<-as.character(netbox2010$geneList)

results<-geneConnector(geneList=geneList,networkGraph=graphReduced,
                       pValueAdj='BH',pValueCutoff=0.05,
                       communityMethod='lec',keepIsolatedNodes=FALSE)


names(results)

plot(results$netboxGraph, layout = layout_with_fr)


write.table(results$netboxOutput,
  file = "network.sif", sep = " ",
  quote = FALSE, col.names = FALSE, row.names = FALSE
)

write.table(results$neighborData,
  file = "neighborList.txt", sep = " ",
  quote = FALSE, col.names = TRUE, row.names = FALSE
)

write.table(results$moduleMembership,
  file = "memb.ebc.txt", sep = " ",
  quote = FALSE, col.names = FALSE, row.names = FALSE
)
```

```
#
write.table(results$nodeType,
  file = "nodeType.txt", sep = " ", quote = FALSE,
  col.names = FALSE, row.names = FALSE
)
#
```

`globalNullModel` *Generate global null model p-value*

Description

Randomly select the same number of nodes in the largest connected component of netbox result as a new gene candidate list and repeat multiple times to produce a distribution of node size and edge numbers. This distribution will be used to produce global p-value of netbox result based on the node size or edge numbers of largest component in the final network result.

Usage

```
globalNullModel(
  netboxGraph,
  networkGraph,
  directed,
  iterations = 30,
  numOfGenes = NULL,
  pValueAdj = "BH",
  pValueCutoff = 0.05
)
```

Arguments

`netboxGraph`	igraph network graph object. This igraph object contains NetBox algorithm identified network from `geneConnector` function
`networkGraph`	igraph network graph object. This igraph object contains curated network information
`directed`	boolean value indicating whether the input network is directed or undirected (default = FALSE)
`iterations`	numeric value for number of iterations
`numOfGenes`	numeric value for number of genes mapped in the initial network
`pValueAdj`	string for p-value correction method c("BH", "Bonferroni") as described in the details section (default = "BH")
`pValueCutoff`	numeric value of p-value cutoff for linker nodes (default = 0.05)

Details

P-value correction methods include the Bonferroni correction ("bonferroni") or Benjamini & Hochberg ("BH").

Value

a list of returned results

- globalNull: data frame of global randomization results
- globalNodesResult: data frame of global null tested results based on nodes
- globalEdgesResult: data frame of global null tested results based on edges

Author(s)

Eric Minwei Liu, <emliu.research@gmail.com>

Examples

```
data(netbox2010)

sifNetwork<-netbox2010$network
graphReduced <- networkSimplify(sifNetwork,directed = FALSE)

geneList<-as.character(netbox2010$geneList)

results<-geneConnector(geneList=geneList,networkGraph=graphReduced,
                       pValueAdj='BH',pValueCutoff=0.05,
                       communityMethod='lec',keepIsolatedNodes=FALSE)

names(results)

# Suggested 100 iterations.
# Use 5 interations in the exampel to save running time.
# globalTest <- globalNullModel(netboxGraph=results$netboxGraph,
#                               networkGraph=graphReduced,
#                               iterations=5, numOfGenes = 274)
```

---

localNullModel *Generate local null model p-value*

Description

This function keeps the number of connections of each nodes in the graph but it rewires the partners of connections and produces modularity score. When it repeats multiple time, a modularity score distribution will be used to produce netbox loacl p-value.

Usage

```
localNullModel(netboxGraph, iterations = 30)
```

Arguments

netboxGraph	igraph network graph object. This igraph object contains NetBox algorithm identified network from geneConnector function
iterations	numeric value for number of iterations

Value

a list of returned results

- randomModularityScore: vector of modularity scores in the iterations of local re-wiring randomization process
- randomMean: numeric value of mean of modularity scores in the iterations of local re-wiring randomization process
- randomSD: numeric value of standard deviation of modularity scores in the iterations of local re-wiring randomization process
- modularityScoreObs: numeric value of observed modularity score in the NetBox algorithm identified network
- zScore: numeric value of z-score
- pValueObs: numeric value of observed p-value

Author(s)

Eric Minwei Liu, <emliu.research@gmail.com>

Examples

```
data(netbox2010)

sifNetwork<-netbox2010$network
graphReduced <- networkSimplify(sifNetwork,directed = FALSE)

geneList<-as.character(netbox2010$geneList)

results<-geneConnector(geneList=geneList,networkGraph=graphReduced,
                       pValueAdj='BH',pValueCutoff=0.05,
                       communityMethod='lec',keepIsolatedNodes=FALSE)

names(results)

# Suggested 1000 iterations.
# Use 10 interations in the exampel to save running time.
localTest <- localNullModel(netboxGraph=results$netboxGraph, iterations=10)
```

---

netbox2010 *network coming with Cerami et al. PLoS One 2010 paper.*

Description

Loading netbox2010 containing 9264 nodes and 68111 interactions. Treated as undirected network. After removing multiple interactions and loops. Returning igraph network of 9264 nodes and 68111 interactions.

Usage

```
netbox2010
```

Format

A data frame with 9264 nodes and 68111 interactions:

name vertex gene name

edges interaction types ...

Value

a data.frame

Source

https://www.ncbi.nlm.nih.gov/pubmed/20169195

---

networkSimplify *Simplify sif network into igraph network graph object*

Description

This function removes duplicated edges and loops to create an igraph graph object from tab delimited sif formatted network file.

Usage

```
networkSimplify(sifNetwork, directed = FALSE)
```

Arguments

sifNetwork	A file with sif network format (There are three columns in the file separated by tab, nodeA interactionType nodeB)
directed	Logical, treat network as directed or undirected graph

Details

For undirected graph, networkSimplify removes duplicated edges and loops to create an igraph graph object from tab delimited sif formatted network file.

For directed graph, networkSimplify selects the first edge and removes the rest duplicated edges and loops to create an igraph graph object from tab delimited sif formatted network file.

Value

a igraph graph object

Author(s)

Eric Minwei Liu, <emliu.research@gmail.com>

Examples

```
data(netbox2010)

sifNetwork <- netbox2010$network
graphReduced <- networkSimplify(sifNetwork, directed = FALSE)
```

`pathway_commons_v8_reactome` *Pathway Commons V8 Reactome*

Description

Contains an example gene list and Pathway Commons V8 Reactome dataset for annotateGraph().

Usage

```
pathway_commons_v8_reactome
```

Format

A list of 354 genes and a data frame of 246590 interactions

geneList an example list of genes

network Pathway Commons V8 Reactome ...

Value

A list of two elements.

Source

https://www.pathwaycommons.org

Index