Package ‘graphite’

April 25, 2017

Version  1.22.0
Date     2017-04-24
Title    GRAPH Interaction from pathway Topological Environment
Author   Gabriele Sales <gabriele.sales@unipd.it>, Enrica Calura
          <enrica.calura@gmail.com>, Chiara Romualdi
          <chiara.romualdi@unipd.it>
Maintainer Gabriele Sales <gabriele.sales@unipd.it>
Description Graph objects from pathway topology derived from Biocarta,
               HumanCyc, KEGG, NCI, Panther, Reactome and SPIKE databases.
License   AGPL-3
Depends  R (>= 2.10), methods
Imports   AnnotationDbi, graph, stats, utils, rappdirs
Suggests BiocStyle, hgu133plus2.db, org.Hs.eg.db, SPIA (>= 2.2),
          topologyGSA (>= 1.4.0), clipper, ALL, testthat
Enhances DEGraph, RCytoscape, RCy3
LazyData  yes
biocViews Pathways, ThirdPartyClient, GraphAndNetwork, Network,
           Reactome, KEGG, BioCarta
NeedsCompilation no

R topics documented:

as.list.DeprecatedPathwayList  ........................................ 2
as.list.PathwayList  ................................................ 3
biocarta  ........................................................... 3
buildPathway  ......................................................... 4
convertIdentifiers  ................................................... 5
cytoscapePlot  ........................................................ 6
DeprecatedPathwayList-class  ........................................... 6
humanyc  ............................................................... 7
kegg  ................................................................. 8
nci  ................................................................. 9
### as.list.DeprecatedPathwayList

**Conversion of DeprecatedPathwayLists into lists.**

#### Description

Converts a `DeprecatedPathwayList` into a list of `Pathways`.

#### Usage

```r
## S3 method for class 'DeprecatedPathwayList'
as.list(x, ...)
```

#### Arguments

- `x`  
  a `DeprecatedPathwayList` object

- `...`  
  extra arguments to `as.list`

#### Value

A list of pathways.

#### Author(s)

Gabriele Sales

#### See Also

- `DeprecatedPathwayList`

#### Examples

```r
as.list(reactome)
```
as.list.PathwayList

Conversion of PathwayLists into lists.

Description

Converts a PathwayList into a list of Pathways.

Usage

```r
## S3 method for class 'PathwayList'
as.list(x, ...)
```

Arguments

- `x`: a PathwayList object
- `...`: extra arguments to as.list

Value

A list of pathways.

Author(s)

Gabriele Sales

See Also

PathwayList

Examples

```r
as.list(kegg)
```

biocarta

BioCarta pathways

Description

BioCarta pathways.

Direct access to this object is deprecated. Call pathways(species, "biocarta") instead.
Pathway topology conversion

BioCarta pathways were retrieved in BioPax format from the PDI database web page. We define a pathway for each BioPax tag “pathway”. Pathway nodes often correspond to multiple gene products. These can be divided into protein complexes (proteins linked by protein-protein interactions) and groups made of alternative members (genes with similar biochemical functions). Thus, when considering signal propagation these groups are considered differently. The first kind (hereafter group AND) should be expanded into a clique (all proteins connected to the others), while the second (hereafter group OR) should be expanded without connection among them. In the BioPax format only one type of group is allowed: protein complexes (group AND) with the tag ‘complex’. However, it often happens that the ‘protein’ tag contains multiple ‘xref’ pointing to alternative elements of the process (group OR).

Compound mediated interactions are interactions for which a compound acts as a bridge between two elements. Since chemical compounds are not usually measured with high-throughput technology, they should be removed from the network to analyse gene signals. However, the trivial elimination of the compounds, without signal propagation, will strongly bias the topology interrupting the signals that pass through them. If element ‘A’ is linked to compound ‘c’ and compound ‘c’ is linked to element ‘B’, element ‘A’ should be linked to element ‘B’. Not all compounds are considered for the propagation because some of them (for example: H2O, ATP, ADP) are highly frequent in map descriptions and the signal propagation through them would lead to chains too long. Compounds not considered for propagation are not characteristic of a specific reaction, but act as secondary substrates/products widely shared among different processes.

graphite allows the user to see the single/multiple relation types that characterized an edge. The type of edges have been kept as much as possible similar to those annotated in the original data format. Some new types have been introduced due to topological conversion needs.

See Also

PathwayList

buildPathway

Build a Pathway object.

This function creates a new object of type Pathway given a data frame describing its edges.

Usage

buildPathway(id, title, edges, species, database, identifier, timestamp=NULL)

Arguments

id the pathway identifier.
title the title of the pathway.
edges a data.frame of pathway edges. Must have the following columns: src, dest, direction and type. Direction must be one of the two strings: "directed" or "undirected".
species the species the pathway belongs to.
**convertIdentifiers**

- **database**: the name of the database the pathway derives from.
- **identifier**: the type of identifier used to label the pathway nodes.
- **timestamp**: when the pathway was annotated, by default the time `buildPathway` is called.

**See Also**

`Pathway-class`

**Examples**

```r
edges <- data.frame(src="672", dest="7157", direction="undirected", type="binding")
pathway <- buildPathway("#1", "example", edges, "hsapiens", "database", "ENTREZID")
```

---

**convertIdentifiers**  
*Convert the node identifiers of a pathway.*

**Description**

Converts the node identifiers of pathways.

**Usage**

`convertIdentifiers(x, to)`

**Arguments**

- **x**: can be a list of pathways or a single pathway
- **to**: a string describing the type of the identifier. Can assume the values "entrez", "symbol" or the name of one of the columns provided by an Annotation package (for example, "UNIPROT").

**Value**

A `Pathway` object.

**See Also**

`Pathway`

**Examples**

```r
r <- pathways("hsapiens", "reactome")
convertIdentifiers(r$"mTOR signalling", "symbol")
```
### cytoscapePlot

*Plot a pathway graph in Cytoscape*

#### Description
Renders the topology of a pathway as a Cytoscape graph.

#### Usage
```
cytoscapePlot(pathway, ..., cy.ver=3)
```

#### Arguments
- **pathway**
  - a `Pathway` object.
- **...**
  - optional arguments forwarded to `pathwayGraph`.
- **cy.ver**
  - select a Cytoscape version. It can be 3 (the default) or 2.

#### Details
Requires the `RCytoscape` package.

#### See Also
- `Pathway`
- `pathwayGraph`

#### Examples
```
## Not run:
r <- pathways()
r <- pathways()
cytoscapePlot(convertIdentifiers(reactome$`Unwinding of DNA`, "symbol"))
## End(Not run)
```

### DeprecatedPathwayList-class

*Class " DeprecatedPathwayList"

#### Description
Represents deprecated objects for accessing pathway databases.

#### Extends
Class "Pathways", directly.
Methods

1[i]: Returns a selection of the pathways contained in the pathway list.
1[[i]]: Access one of the pathways contained in the pathway list.
1$'title' \ Access one of the pathways by its title.
convertIdentifiers(l, to) Returns a new list of pathways using a different type of node identifiers.
length(l) Returns the number of pathways contained in the list.
names(l) Returns the titles of the pathways contained in the list.
prepareSPIA(l, pathwaySetName, print.names=FALSE) Prepares the pathways for a SPIA analysis.
runClipper(l, expr, classes, method, maxNodes=150, ...) Runs a clipper analysis over all the pathways in the list.
runDEGraph(l, expr, classes, maxNodes=150, ...) Runs a DEGraph analysis over all the pathways in the list.
runTopologyGSA(l, test, exp1, exp2, alpha, maxNodes=150, ...) Runs a topologyGSA analysis over all the pathways in the list.

Author(s)

Gabriele Sales

See Also

PathwayList

Description

HumanCyc pathways.

Direct access to this object is deprecated. Call pathways(species, "humancyc") instead.


Pathway topology conversion

HumanCyc pathways were retrieved in the BioPax format downloaded by the Pathway Commons web page http://www.pathwaycommons.org.

We define a pathway for each BioPax tag “pathway”. Pathway nodes often correspond to multiple gene products. These can be divided into protein complexes (proteins linked by protein-protein interactions) and groups made of alternative members (genes with similar biochemical functions). Thus, when considering signal propagation these groups are considered differently. The first kind (hereafter group AND) should be expanded into a clique (all proteins connected to the others), while the second (hereafter group OR) should be expanded without connection among them. In
the BioPax format only one type of group is allowed: protein complexes (group AND) with the tag 'complex'. However, it often happens that the 'protein' tag contains multiple 'xref' pointing to alternative elements of the process (group OR).

Compound mediated interactions are interactions for which a compound acts as a bridge between two elements. Since chemical compounds are not usually measured with high-throughput technology, they should be removed from the network to analyse gene signals. However, the trivial elimination of the compounds, without signal propagation, will strongly bias the topology interrupting the signals that pass through them. If element 'A' is linked to compound 'c' and compound 'c' is linked to element 'B', element 'A' should be linked to element 'B'. Not all compounds are considered for the propagation because some of them (for example: H2O, ATP, ADP) are highly frequent in map descriptions and the signal propagation through them would lead to chains too long. Compounds not considered for propagation are not characteristic of a specific reaction, but act as secondary substrates/products widely shared among different processes.

graphite allows the user to see the single/multiple relation types that characterized an edge. The type of edges have been kept as much as possible similar to those annotated in the original data format. Some new types have been introduced due to topological conversion needs.

See Also

PathwayList

---

**Description**

**KEGG** pathways.

Direct access to this object is deprecated. Call `pathways(species, "kegg")` instead.


**Pathway topology conversion**

KEGG pathway were retrieved in KGML format from the KEGG ftp site.

KEGG database provides separate xml files, one for each pathway. A pathway is therefore define by all the reactions described within each file.

Pathway nodes often correspond to multiple gene products. These can be divided into protein complexes (proteins linked by protein-protein interactions) and groups made of alternative members (genes with similar biochemical functions). Thus, when considering signal propagation these groups are considered differently. The first kind (hereafter group AND) should be expanded into a clique (all proteins connected to the others), while the second (hereafter group OR) should be expanded without connection among them. In the KGML format there are two ways of defining nodes with multiple elements: protein complexes (group AND defined by entry type="group") and groups with alternative members (group OR defined by entry type="gene").

Compound mediated interactions are interactions for which a compound acts as a bridge between two elements. Since chemical compounds are not usually measured with high-throughput technology, they should be removed from the network to analyse gene signals. However, the trivial elimination of the compounds, without signal propagation, will strongly bias the topology interrupting the signals that pass through them. If element 'A' is linked to compound 'c' and compound
'c' is linked to element 'B', element 'A' should be linked to element 'B'. Within the KGML format there are two different ways of describing a compound mediated interaction: i) direct interaction type="PPrel" ('A' interacts with 'B' through compound 'c') and ii) indirect one type="PCrel" ('A' interacts to compound 'c' and 'c' interacts with 'B').

Not all compounds are considered for the propagation because some of them (for example: H2O, ATP, ADP) are highly frequent in map descriptions and the signal propagation through them would lead to chains too long. Compounds not considered for propagation are not characteristic of a specific reaction, but act as secondary substrates/products widely shared among different processes.

graphite allows the user to see the single/multiple relation types that characterized an edge. The type of edges have been kept as much as possible similar to those annotated in the original data format. Some new types have been introduced due to topological conversion needs.

See Also
PathwayList

nci  NCI pathways

nci

Description
NCI pathways.

Pathway topology conversion

NCI pathways were retrieved in BioPax format from the PDI database web page.

We define a pathway for each BioPax tag "pathway". Pathway nodes often correspond to multiple gene products. These can be divided into protein complexes (proteins linked by protein-protein interactions) and groups made of alternative members (genes with similar biochemical functions). Thus, when considering signal propagation these groups are considered differently. The first kind (hereafter group AND) should be expanded into a clique (all proteins connected to the others), while the second (hereafter group OR) should be expanded without connection among them. In the BioPax format only one type of group is allowed: protein complexes (group AND) with the tag 'complex'. However, it often happens that the 'protein' tag contains multiple 'xref' pointing to alternative elements of the process (group OR).

Compound mediated interactions are interactions for which a compound acts as a bridge between two elements. Since chemical compounds are not usually measured with high-throughput technology, they should be removed from the network to analyse gene signals. However, the trivial elimination of the compounds, without signal propagation, will strongly bias the topology interrupting the signals that pass through them. If element 'A' is linked to compound 'c' and compound 'c' is linked to element 'B', element 'A' should be linked to element 'B'. Not all compounds are considered for the propagation because some of them (for example: H2O, ATP, ADP) are highly frequent in map descriptions and the signal propagation through them would lead to chains too long. Compounds not considered for propagation are not characteristic of a specific reaction, but act as secondary substrates/products widely shared among different processes.

graphite allows the user to see the single/multiple relation types that characterized an edge. The type of edges have been kept as much as possible similar to those annotated in the original data format. Some new types have been introduced due to topological conversion needs.
See Also

PathwayList

---

**panther**

**PathwayList**

---

**Description**

PANTHER pathways.

Direct access to this object is deprecated. Call `pathways(species, "panther")` instead.


**Pathway topology conversion**

Panther pathways were retrieved in the BioPax format downloaded by the Pathway Commons web page [http://www.pathwaycommons.org](http://www.pathwaycommons.org).

We define a pathway for each BioPax tag “pathway”. Pathway nodes often correspond to multiple gene products. These can be divided into protein complexes (proteins linked by protein-protein interactions) and groups made of alternative members (genes with similar biochemical functions). Thus, when considering signal propagation these groups are considered differently. The first kind (hereafter group AND) should be expanded into a clique (all proteins connected to the others), while the second (hereafter group OR) should be expanded without connection among them. In the BioPax format only one type of group is allowed: protein complexes (group AND) with the tag ‘complex’. However, it often happens that the ‘protein’ tag contains multiple ‘xref’ pointing to alternative elements of the process (group OR).

Compound mediated interactions are interactions for which a compound acts as a bridge between two elements. Since chemical compounds are not usually measured with high-throughput technology, they should be removed from the network to analyse gene signals. However, the trivial elimination of the compounds, without signal propagation, will strongly bias the topology interrupting the signals that pass through them. If element ‘A’ is linked to compound ‘c’ and compound ‘c’ is linked to element ‘B’, element ‘A’ should be linked to element ‘B’. Not all compounds are considered for the propagation because some of them (for example: H2O, ATP, ADP) are highly frequent in map descriptions and the signal propagation through them would lead to chains too long. Compounds not considered for propagation are not characteristic of a specific reaction, but act as secondary substrates/products widely shared among different processes.

graphite allows the user to see the single/multiple relation types that characterized an edge. The type of edges have been kept as much as possible similar to those annotated in the original data format. Some new types have been introduced due to topological conversion needs.

See Also

PathwayList
Description

A biological pathway.

Slots

id: the native ID of the pathway.
title: the title of the pathway.
edges: a data.frame describing the edges of this pathway.
database: the name of the database the pathway was derived from.
species: the name of the species in which the pathway was annotated.
identifier: the type of node identifier used by this pathway.
timestamp: the date in which the pathway data was retrieved.

Methods

convertIdentifiers(p, to) Returns a new pathway using a different type of node identifiers.
edges(p) Returns a data.frame describing the edges of this pathway.
nodes(p) Return the names of the nodes belonging to this pathway.
plot(p) Shows the pathway topology in Cytoscape.
runClipper(p, expr, classes, method, ...) Runs a clipper analysis over the pathway.
runDEGraph(p, expr, classes, ...) Runs a DEGraph analysis over the pathway.
runTopologyGSA(p, test, exp1, exp2, alpha, ...) Runs a topologyGSA analysis over the pathway.

Author(s)

Gabriele Sales

See Also

pathways
### pathwayDatabases

**Description**
Obtains the list of pathway databases available through graphite.

**Usage**
```r
pathwayDatabases()
```

**Value**
Returns a `data.frame` with two columns: species and database.

**Author(s)**
Gabriele Sales

**See Also**
- `pathways`

**Examples**
```r
pathwayDatabases()
```

---

### pathwayGraph

**Description**
Graph representing the topology of a pathway.

**Usage**
```r
pathwayGraph(pathway, edge.types=NULL)
```

**Arguments**
- `pathway`: a `Pathway` object.
- `edge.types`: keep only the edges matching the type names in this vector.

**Value**
A `graphNEL` object.

**See Also**
- `Pathway`
- `graphNEL`
**PathwayList-class**

**Examples**

```r
r <- pathways("hsapiens", "reactome")
pathwayGraph(r$"mTOR signalling", edge.types="Binding")
```

**Description**

A collection of pathways from a single database.

**Extends**

Class "Pathways", directly.

**Methods**

- `l[i]`: Returns a selection of the pathways contained in the pathway list.
- `l[[i]]`: Access one of the pathways contained in the pathway list.
- `l$title` Access one of the pathways by its title.
- `convertIdentifiers(l, to)`: Returns a new list of pathways using a different type of node identifiers.
- `length(l)`: Returns the number of pathways contained in the list.
- `names(l)`: Returns the titles of the pathways contained in the list.
- `prepareSPIA(l, pathwaySetName, print.names=FALSE)`: Prepares the pathways for a SPIA analysis.
- `runClipper(l, expr, classes, method, maxNodes=150, ...)`: Runs a clipper analysis over all the pathways in the list.
- `runDEGraph(l, expr, classes, maxNodes=150, ...)`: Runs a DEGraph analysis over all the pathways in the list.
- `runTopologyGSA(l, test, exp1, exp2, alpha, maxNodes=150, ...)`: Runs a topologyGSA analysis over all the pathways in the list.

**Author(s)**

Gabriele Sales

**See Also**

`pathways`
pathways  Retrieve a list of pathways.

Description
This function retrieves a list of pathways from a database for a given species.

Usage
pathways(species, database)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>species</td>
<td>one of the supported species</td>
</tr>
<tr>
<td>database</td>
<td>the name of the pathway database</td>
</tr>
</tbody>
</table>

Value
A PathwayList object.

See Also
PathwayList

Examples
pathways("hsapiens", "reactome")

Pathways-class  Class "Pathways"

Description
A virtual class acting as a common parent to all other classes representing pathway databases.

Objects from the Class
A virtual Class: No objects may be created from it.

Methods
No methods defined with class "Pathways" in the signature.

Author(s)
Gabriele Sales

See Also
PathwayList
**prepareSPIA**

*Prepare pathway dataset needed by runSPIA.*

### Description

Prepare pathway dataset needed by runSPIA. See runSPIA and spia for more details.

### Usage

```r
prepareSPIA(db, pathwaySetName, print.names=FALSE)
```

### Arguments

- `db`: a `PathwayList` object or a list of `Pathways`.
- `pathwaySetName`: name of the output pathway set.
- `print.names`: print pathway names as the conversion advances.

### References


### See Also

- runSPIA
- spia
- PathwayList

---

**reactome**

*Reactome pathways*

### Description

Reactome pathways.

Direct access to this object is deprecated. Call pathways(species, "reactome") instead.

Pathway topology conversion

Reactome pathways were retrieved in the BioPax format from the Reactome database web page. We define a pathway for each BioPax tag “pathway”. Pathway nodes often correspond to multiple gene products. These can be divided into protein complexes (proteins linked by protein-protein interactions) and groups made of alternative members (genes with similar biochemical functions). Thus, when considering signal propagation these groups are considered differently. The first kind (hereafter group AND) should be expanded into a clique (all proteins connected to the others), while the second (hereafter group OR) should be expanded without connection among them. In the BioPax format only one type of group is allowed: protein complexes (group AND) with the tag 'complex'. However, it often happens that the 'protein' tag contains multiple 'xref' pointing to alternative elements of the process (group OR).

Compound mediated interactions are interactions for which a compound acts as a bridge between two elements. Since chemical compounds are not usually measured with high-throughput technology, they should be removed from the network to analyse gene signals. However, the trivial elimination of the compounds, without signal propagation, will strongly bias the topology interrupting the signals that pass through them. If element 'A' is linked to compound 'c' and compound 'c' is linked to element 'B', element 'A' should be linked to element 'B'. Not all compounds are considered for the propagation because some of them (for example: H2O, ATP, ADP) are highly frequent in map descriptions and the signal propagation through them would lead to chains too long. Compounds not considered for propagation are not characteristic of a specific reaction, but act as secondary substrates/products widely shared among different processes.

graphite allows the user to see the single/multiple relation types that characterized an edge. The type of edges have been kept as much as possible similar to those annotated in the original data format. Some new types have been introduced due to topological conversion needs.

See Also

PathwayList

---

runClipper

Run a topological analysis on an expression dataset using clipper.

Description

clipper is a package for topological gene set analysis. It implements a two-step empirical approach based on the exploitation of graph decomposition into a junction tree to reconstruct the most relevant signal path. In the first step clipper selects significant pathways according to statistical tests on the means and the concentration matrices of the graphs derived from pathway topologies. Then, it "clips" the whole pathway identifying the signal paths having the greatest association with a specific phenotype.

Usage

runClipper(x, expr, classes, method, ...)

Arguments

x a PathwayList, a list of Pathways or a single Pathway object.
expr a matrix (size: number p of genes x number n of samples) of gene expression.
runClipperMulti

Run a topological analysis on an expression dataset using clipper.

Description
This function is deprecated and will be removed in a future release. You can use runClipper instead.

Usage
runClipperMulti(pathways, expr, classes, method, maxNodes=150, ...)

Arguments
- pathways: a PathwayList object.
- expr: a matrix (size: number p of genes x number n of samples) of gene expression.
- classes: a vector (length: n) of class assignments.
- method: the kind of test to perform on the cliques. It could be either "mean" or "variance".
- maxNodes: ignore a pathway when it has more than this number of nodes.
- ...: Additional options; see for details easyClip.

Details
The expression data and the pathway have to be annotated in the same set of identifiers.

Examples
```r
if (require(clipper) & require(ALL)) {
  k <- pathways("hsapiens", "kegg")
  path <- convertIdentifiers(k$"Chronic myeloid leukemia", "entrez")
  genes <- nodes(path)
  data(ALL)
  all <- as.matrix(exprs(ALL[1:length(genes),1:20]))
  classes <- c(rep(1,10), rep(2,10))
  rownames(all) <- genes
  runClipper(path, all, classes, "mean", pathThr=0.1)
}
```
Details

The expression data and the pathway have to be annotated in the same set of identifiers.

Value

A list with two elements:

- results: a list with one entry for each successfully analyzed pathway;
- errors: a vector containing the error messages of failed analyses.

References


See Also

clipper

Examples

```r
if (require(clipper) & require(ALL)){
  k <- pathways("hsapiens", "kegg")
  paths <- convertIdentifiers(k[1:5], "entrez")
  genes <- unlist(lapply(paths, nodes))
  data(ALL)
  all <- as.matrix(exprs(ALL[1:length(genes),1:20]))
  classes <- c(rep(1,10), rep(2,10))
  rownames(all) <- genes
  runClipperMulti(paths, all, classes, "mean", pathThr=0.1)
}
```

---

**runDEGraph**

Run a topological analysis on an expression dataset using DEGraph package.

**Description**

DEGraph implements recent hypothesis testing methods which directly assess whether a particular gene network is differentially expressed between two conditions.

**Usage**

`runDEGraph(x, expr, classes, ...)`

**Arguments**

- `x`: a `PathwayList`, a list of `Pathways` or a single `Pathway` object.
- `expr`: a matrix (size: number p of genes x number n of samples) of gene expression.
- `classes`: a vector (length: n) of class assignments.
- `...`: when invoked on a `PathwayList`, can use the named option "maxNodes" to limit the analysis to those pathways having up to this given number of nodes.
runDEGraphMulti

Run a topological analysis on an expression dataset using DEGraph package.

Description

This function is deprecated and will be removed in a future release. You can use runDEGraph instead.

Usage

runDEGraphMulti(pathways, expr, classes, maxNodes=150)

Arguments

- pathways: a PathwayList object.
- expr: A matrix (size: number p of genes x number n of samples) of gene expression.
- classes: A vector (length: n) of class assignments.
- maxNodes: Ignore pathways with more than "maxNodes" nodes. Set to NULL to disable the filter.

Details

The expression data and the pathway have to be annotated in the same set of identifiers.

Value

A list with two elements:

- results: a list with one entry for each successfully analyzed pathway;
- errors: a vector containing the error messages of failed analyses.

Details

The expression data and the pathway have to be annotated in the same set of identifiers.

References


See Also

testOneGraph

Examples

```r
if (require(DEGraph)) {
  data("Loi2008_DEGraphVignette"

  b <- pathways("hsapiens", "biocarta"
  p <- convertIdentifiers(b[["actions of nitric oxide in the heart"]], "entrez"
  runDEGraph(p, exprLoi2008, classLoi2008)
}
```
runSPIA

Run SPIA analysis

Description

Run a topological analysis on an expression dataset using SPIA.

Usage

runSPIA(de, all, pathwaySetName, ...)

Arguments

de A named vector containing log2 fold-changes of the differentially expressed genes. The names of this numeric vector are Entrez gene IDs.

all A vector with the Entrez IDs in the reference set. If the data was obtained from a microarray experiment, this set will contain all genes present on the specific array used for the experiment. This vector should contain all names of the 'de' argument.

pathwaySetName A list of pathways like kegg, nci or reactome.

... Additional options to pass to spia.

Details

The spia option "organism" is internally used. It is an error use it in the additional options.

References


See Also
testOneGraph

Examples

if (require(DEGraph)) {
  data("Loi2008_DEGraphVignette")

  b <- pathways("hsapiens", "biocarta")
  ps <- convertIdentifiers(b[1:3], "entrez")
  runDEGraphMulti(ps, exprLoi2008, classLoi2008)
}
Value

The same of spia, without KEGG links. A data frame containing the ranked pathways and various statistics: pSize is the number of genes on the pathway; NDE is the number of DE genes per pathway; tA is the observed total perturbation accumulation in the pathway; pNDE is the probability to observe at least NDE genes on the pathway using a hypergeometric model; pPERT is the probability to observe a total accumulation more extreme than tA only by chance; pG is the p-value obtained by combining pNDE and pPERT; pGFdr and pGFWER are the False Discovery Rate and respectively Bonferroni adjusted global p-values; and the Status gives the direction in which the pathway is perturbed (activated or inhibited).

References


See Also

spia

Examples

```r
if (require(SPIA) && require(hgu133plus2.db)) {
  data(colorectalcancer)
  x <- hgu133plus2ENTREZ
  top$ENTREZ <- unlist(as.list(x[top$ID]))
  top <- top[!is.na(top$ENTREZ), ]
  top <- top[!duplicated(top$ENTREZ), ]
  tg1 <- top[top$adj.P.Val < 0.05, ]
  DE_Colorectal = tg1$logFC
  names(DE_Colorectal) <- as.vector(tg1$ENTREZ)
  ALL_Colorectal <- top$ENTREZ
  b <- pathways("hsapiens", "biocarta")
  prepareSPIA(b[1:20], "biocartaEx")
  runSPIA(de=DE_Colorectal, all=ALL_Colorectal, "biocartaEx")
}
```

---

**runTopologyGSA**

*Run a topological analysis on an expression dataset using topologyGSA.*

**Description**

Use graphical models to test the pathway components highlighting those involved in its deregulation.
Usage

runTopologyGSA(x, test, exp1, exp2, alpha, ...)

Arguments

x a PathwayList, a list of Pathways or a single Pathway object.
test Either "var" and "mean". Determine the type of test used by topologyGSA.
exp1 Experiment matrix of the first class, genes in columns.
exp2 Experiment matrix of the second class, genes in columns.
alpha Significance level of the test.
... Additional parameters forwarded to topologyGSA.

When invoked on a PathwayList, can use the named option "maxNodes" to limit the analysis to those pathways having up to this given number of nodes.

Details

This function produces a warning and returns NULL when the number of genes in common between the expression matrices and the pathway is less than 3.

References

Massa MS, Chiogna M, Romualdi C. Gene set analysis exploiting the topology of a pathway. BMC System Biol. 2010 Sep 1;4:121.

See Also

pathway.var.test pathway.mean.test

Examples

if (require(topologyGSA)) {
  data(examples)
  k <- pathways("hsapiens", "kegg")
  p <- convertIdentifiers(k["Fc epsilon RI signaling pathway"], "symbol")
  runTopologyGSA(p, "var", y1, y2, 0.05)
}

runTopologyGSAMulti Run a topological analysis on an expression dataset using topologyGSA.

Description

This function is deprecated and will be removed in a future release. You can use runTopologyGSA instead.

Usage

runTopologyGSAMulti(pathways, test, exp1, exp2, alpha, maxNodes=150, ...)
Arguments

- **pathways**: a `PathwayList` object.
- **test**: Either "var" and "mean". Determine the type of test used by topologyGSA.
- **exp1**: Experiment matrix of the first class, genes in columns.
- **exp2**: Experiment matrix of the second class, genes in columns.
- **alpha**: Significance level of the test.
- **maxNodes**: Ignore pathways with more than "maxNodes" nodes. Set to "NULL" to disable the filter.
- **...**: Additional parameters for topologyGSA.

Details

This function produces a warning and whenever the number of genes in common between the expression matrices and a pathway is less than 3.

Value

A list with two elements:

- results: a list with one entry for each successfully analyzed pathway;
- errors: a vector containing the error messages of failed analyses.

References

Massa MS, Chiogna M, Romualdi C. Gene set analysis exploiting the topology of a pathway. BMC System Biol. 2010 Sep 1;4:121.

See Also

`pathway.var.test` `pathway.mean.test`

Examples

```r
if (require(topologyGSA)) {
  data(examples)
  k <- pathways("hsapiens", "kegg")
  ps <- convertIdentifiers(
    k[c("Acute myeloid leukemia", "Fc epsilon RI signaling pathway")],
    "symbol")
  runTopologyGSAMulti(ps, "var", y1, y2, 0.05)
}
```
Index

*Topic **DEGraph**
  - runDEGraph, 18
  - runDEGraphMulti, 19

*Topic **analysis**
  - runClipper, 16
  - runClipperMulti, 17
  - runDEGraph, 18
  - runDEGraphMulti, 19
  - runSPIA, 20
  - runTopologyGSA, 21
  - runTopologyGSAMulti, 22

*Topic **classes**
  - DeprecatedPathwayList-class, 6
  - Pathway-class, 11
  - PathwayList-class, 13
  - Pathways-class, 14

*Topic **clipper**
  - runClipper, 16
  - runClipperMulti, 17

*Topic **spia**
  - runSPIA, 20

*Topic **topologyGSEA**
  - runTopologyGSA, 21
  - runTopologyGSAMulti, 22

*Topic **topology**
  - runClipper, 16
  - runClipperMulti, 17
  - runDEGraph, 18
  - runDEGraphMulti, 19
  - runSPIA, 20
  - runTopologyGSA, 21
  - runTopologyGSAMulti, 22

[,DeprecatedPathwayList-method
  (DeprecatedPathwayList-class), 6

$,PathwayList-method
  (PathwayList-class), 13

as.list.DeprecatedPathwayList, 2
as.list.PathwayList, 3
biocarta, 3
buildPathway, 4
clipper, 17, 18
convertIdentifiers, 5
convertIdentifiers,DeprecatedPathwayList-method
  (DeprecatedPathwayList-class), 6
convertIdentifiers,Pathway-method
  (Pathway-class), 11
convertIdentifiers,PathwayList-method
  (PathwayList-class), 13
cytoscapePlot, 6
DeprecatedPathwayList, 2
DeprecatedPathwayList-class, 6
easyClip, 17
edges,Pathway,ANY-method
  (Pathway-class), 11
graphNEL, 12
humancyc, 7
kegg, 8, 20
length,DeprecatedPathwayList-method
  (DeprecatedPathwayList-class), 6
length,PathwayList-method
  (PathwayList-class), 13
names,DeprecatedPathwayList-method
  (DeprecatedPathwayList-class), 6
names,PathwayList-method
  (PathwayList-class), 13

24
nci, 9, 20
nodes, Pathway-method (Pathway-class), 11

panther, 10
Pathway, 2, 3, 5, 6, 12, 15, 16, 18, 22
Pathway-class, 11
pathway.mean.test, 22, 23
pathway.var.test, 22, 23
pathwayDatabases, 12
pathwayGraph, 6, 12
PathwayList, 3, 4, 7–10, 14–19, 22, 23
PathwayList-class, 13
Pathways, 6, 13
pathways, 11–13, 14
Pathways-class, 14
plot, Pathway, ANY-method
(Pathway-class), 11
prepareSPIA, 15
prepareSPIA, DeprecatedPathwayList-method
(DeprecatedPathwayList-class), 6
prepareSPIA, list-method (prepareSPIA), 15
prepareSPIA, PathwayList-method
(PathwayList-class), 13

reactome, 15, 20
runClipper, 16, 17
runClipper, DeprecatedPathwayList-method
(DeprecatedPathwayList-class), 6
runClipper, list-method (runClipper), 16
runClipper, Pathway-method
(Pathway-class), 11
runClipper, PathwayList-method
(PathwayList-class), 13
runClipperMulti, 17
runDEGraph, 18, 19
runDEGraph, DeprecatedPathwayList-method
(DeprecatedPathwayList-class), 6
runDEGraph, list-method (runDEGraph), 18
runDEGraph, Pathway-method
(Pathway-class), 11
runDEGraph, PathwayList-method
(PathwayList-class), 13
runDEGraphMulti, 19
runSPIA, 15, 20
runTopologyGSA, 21, 22
runTopologyGSA, DeprecatedPathwayList-method
(DeprecatedPathwayList-class), 6
runToplogyGSA, list-method
(runTopologyGSA), 21
runToplogyGSA, Pathway-method
(Pathway-class), 11
runToplogyGSA, PathwayList-method
(PathwayList-class), 13
runToplogyGSAMulti, 22
spia, 15, 20, 21
testOneGraph, 19, 20