

# Package ‘rTRM’

May 26, 2021

**Type** Package

**Title** Identification of transcriptional regulatory modules from PPI networks

**Version** 1.31.0

**Date** 2015-12-25

**Author** Diego Diez

**Depends** R (>= 2.10), igraph (>= 1.0)

**Imports** AnnotationDbi, DBI, RSQLite

**Suggests** RUnit, BiocGenerics, MotifDb, graph, PWMEnrich, biomaRt, knitr, Biostrings, BSgenome.Mmusculus.UCSC.mm8.masked, org.Hs.eg.db, org.Mm.eg.db, ggplot2

**Maintainer** Diego Diez <diego10ruiz@gmail.com>

**Description** rTRM identifies transcriptional regulatory modules (TRMs) from protein-protein interaction networks.

**License** GPL-3

**LazyLoad** yes

**ByteCompile** yes

**VignetteBuilder** knitr

**biocViews** Transcription, Network, GeneRegulation, GraphAndNetwork

**URL** <https://github.com/ddiez/rTRM>

**BugReports** <https://github.com/ddiez/rTRM/issues>

**git\_url** <https://git.bioconductor.org/packages/rTRM>

**git\_branch** master

**git\_last\_commit** 1fa16e6

**git\_last\_commit\_date** 2021-05-19

**Date/Publication** 2021-05-26

**R topics documented:**

rTRM-package	2
annotateFreq	3
annotateModule	4
annotateTRM	4
biogrid_hs	5
biogrid_mm	5
findTRM	6
getAnnotations	7
getBiogridData	7
getConcentricList	8
getLargestComp	9
getMaps	9
getMatrices	10
getMotifsFromEntrezgene	10
getMotifsFromSymbol	11
getOrthologFromMatrix	11
getOrthologs	12
getOrthologsFromBiomart	12
getSequencesFromGenome	13
getSimilarityMatrix	14
getTFclass	14
getTFclassFromEntrezgene	15
getTFterms	15
initBiomart	16
layout.arc	16
layout.concentric	17
plotDegree	17
plotGraph	18
plotTRM	19
plotTRMlegend	20
processBiogrid	20
removeVertices	21
writeTRMreport	21
<b>Index</b>	<b>23</b>

rTRM-package

*Identification transcription regulatory modules (TRMs)***Description**

This package identifies transcriptional regulatory modules (TRMs) from PPI networks.

**Details**

Package: rTRM  
Type: Package  
Version: 1.0  
Date: 2013-02-01  
License: GPL-3

**Author(s)**

Diego Diez

Maintainer: Diego Diez <diego10ruiz@gmail.com>

**Examples**

```
getAnnotations()
```

---

annotateFreq

*Annotate a graph with frequency of nodes/edges in other graphs.*

---

**Description**

Returns an annotated graph with node size and edge width proportional at the number of occurrences of nodes/edges in a supplied list of graphs.

**Usage**

```
annotateFreq(g, graph_list)
```

**Arguments**

`g` target graph to annotate.  
`graph_list` list of graph to extract information from.

**Details**

Commonly `graph_list` refers to a list of predicted TRMs (with `findTRM`) and `g` is the combined TRM. This function annotates the nodes/edges in `g` to know their frequency in the original list of graphs.

**Author(s)**

Diego Diez

---

annotateModule	<i>Annotate a network module with information</i>
----------------	---

---

**Description**

Uses information about expression, enrichment and parent PPI network to annotate a subgraph.

**Usage**

```
annotateModule(g, enrich, trm, targets, ppi, exprs, tfs)
```

**Arguments**

g	graph to annotate in igraph format.
enrich	list of enriched transcription factors (or motifs).
trm	TRM to compare with (to identify bridges).
targets	list of target transcription factors (typically those with ChIP-seq data).
ppi	parent PPI network (to check membership of nodes).
exprs	list of entrezgene ids representing expressed genes.
tfs	

**Author(s)**

Diego Diez

---

annotateTRM	<i>Annotate a network object with information about clusters.</i>
-------------	---

---

**Description**

This function takes a network object and includes cluster information as piecolor attribute, suitable to be plotted with plotTRM()

**Usage**

```
annotateTRM(g, target)
```

**Arguments**

g	a network object.
target	target node (from findTRM())

**Author(s)**

Diego Diez

---

biogrid_hs	<i>Network dataset of class 'igraph'</i>
------------	--

---

**Description**

Human protein-protein interaction (PPI) dataset from the BioGRID database release .

**Usage**

```
data(biogrid_hs)
```

**Format**

An igraph object.

**Author(s)**

Diego Diez

---

biogrid_mm	<i>Network dataset of class 'igraph'</i>
------------	--

---

**Description**

Mouse protein-protein interaction (PPI) dataset from the BioGRID database .

**Usage**

```
data(biogrid_mm)
```

**Format**

An igraph object.

**Author(s)**

Diego Diez

---

findTRM	<i>Identifies a TRM associated with a target node and one or more query nodes.</i>
---------	--

---

### Description

This is the main function used to identify TRMs. It takes a graph object and uses it to search in the neighborhood of a target node for query nodes that are separated by a maximum distance (controlled by the `max.bridge` parameter).

### Usage

```
findTRM(g, target, query, method = "nsa", max.bridge = 1, extended = FALSE, strict = FALSE, type = "igraph")
```

### Arguments

<code>g</code>	the network used to identify TRMs (typically a PPI network)
<code>target</code>	character variable with the name of a target node.
<code>query</code>	character vector with the list of query nodes.
<code>method</code>	method to use.
<code>max.bridge</code>	maximum number of nodes allowed between the target and query nodes.
<code>extended</code>	whether to allow distance restrictions to include both target and query nodes.
<code>strict</code>	whether to return a single component (using <code>decompose.graph()</code> )
<code>type</code>	type of graph object to return, either an "igraph" (the default) or a "graphNEL"

### Details

Currently only "first" and "nsa" methods are available. First is used for tests and returns the first neighborhood of the target node. Method "nsa" implements the TRM finding algorithm.

### Value

A network in igraph format or NULL.

### Author(s)

Diego Diez

### Examples

```
# load example network.
load(system.file(package = "rTRM", "extra/example.rda"))

# define target and query nodes.
target = "N6"
query = c("N7", "N12", "N28")
```

```
# find TRM:  
s = findTRM(g, target = target, query = query, method = "nsa", max.bridge = 1)
```

---

getAnnotations	<i>Obtain the 'pwm' table from the database, containing PWM's annotations.</i>
----------------	--

---

### Description

Obtain the 'pwm' table from the database, containing PWM's annotations.

### Usage

```
getAnnotations(filter, dbname = NULL)
```

### Arguments

filter	one or more PWM ids.
dbname	the location of the database (to load custom databases).

### Author(s)

Diego Diez

### Examples

```
ann = getAnnotations()
```

---

getBiogridData	<i>Downloads network data from BioGRID in TAB2 format.</i>
----------------	--

---

### Description

This function is used to generate igraph network objects from BioGRID data. It downloads the database into a data.frame object that can be used later with processBiogrid()

### Usage

```
getBiogridData(release)
```

### Arguments

release	release of BioGRID to download.
---------	---------------------------------

**Details**

The release to download must be specified as currently there is no way to download automatically the latests release.

**Value**

An data.frame object.

**Author(s)**

Diego Diez

---

getConcentricList	<i>Returns a list with nodes membership to be used in a graph with a concentric layout</i>
-------------------	--

---

**Description**

Specify target and enriched motifs and returns a list with circle membership. This information is used by layout.concentric to position the nodes in plots.

**Usage**

```
getConcentricList(g, t, e, max.size = 60, order.by = "label")
```

**Arguments**

g	graph to layout (extract the nodes).
t	list of target nodes (will go in the center).
e	list of enriched nodes (will go in the periphery).
max.size	maximum number of nodes per layer.
order.by	ordering attribute for list before split.

**Author(s)**

Diego Diez



---

`getLargestComp`      *Gets the largest connected component*

---

**Description**

Returns the largest connected component from a graph.

**Usage**

```
getLargestComp(g)
```

**Arguments**

`g`                    an igraph object.

**Author(s)**

Diego Diez

---

`getMaps`                    *Obtain the mapping between PWM and Entrez Gene identifiers.*

---

**Description**

Obtain the mapping between PWM and Entrez Gene identifiers.

**Usage**

```
getMaps(filter, dbname = NULL)
```

**Arguments**

`filter`                    vector of PWMs to filter results.  
`dbname`

**Author(s)**

Diego Diez

**Examples**

```
getMaps()
```

---

getMatrices	<i>Obtain a list of PWMs.</i>
-------------	-------------------------------

---

**Description**

Returns a list of PWMs, by default all the PWMs in the database. Alternatively, filtered by the ids provided by filter.

**Usage**

```
getMatrices(filter, dbname = NULL)
```

**Arguments**

filter	list of PWMs to filter results.
dbname	

**Author(s)**

Diego Diez

**Examples**

```
pwms = getMatrices()
```

---

getMotifsFromEntrezgene	<i>Retrieve PWMs associated with genes provided as entrezgene identifiers.</i>
-------------------------	--

---

**Description**

Retrieve PWMs associated with genes provided as entrezgene identifiers.

**Usage**

```
getMotifsFromEntrezgene(e, organism)
```

**Arguments**

e	vector of entrezgene identifiers to retrieve exiting PWMs.
organism	target organism.

**Author(s)**

Diego Diez

---

`getMotifsFromSymbol`    *Retrieve PWMs associated with genes provided as symbol.*

---

**Description**

Retrieve PWMs associated with genes provided as symbol.

**Usage**

```
getMotifsFromSymbol(s, organism)
```

**Arguments**

`s`                    vector of gene symbols.  
`organism`            target organism.

**Author(s)**

Diego Diez

---

`getOrthologFromMatrix`    *Obtain gene identifiers for a target organism associated with a list of PWMs.*

---

**Description**

Obtain gene identifiers for a target organism associated with a list of PWMs.

**Usage**

```
getOrthologFromMatrix(filter, organism = "human", dbname = NULL)
```

**Arguments**

`filter`                vector of matrices to filter results.  
`organism`            target organism.  
`dbname`                database- usually not need to specify.

**Author(s)**

Diego Diez

getOrthologs                      *Obtain the mapping to Entrez Gene identifiers in the given organism.*

---

**Description**

Obtain the mapping to Entrez Gene identifiers in the given organism.

**Usage**

```
getOrthologs(filter, organism, dbname = NULL)
```

**Arguments**

filter	entrezgene identifiers for the original mapping (PWM to gene). These can belong to diverse species and correspond to the "entrezgene" column obtained with getMaps() function.
organism	target organisms, currently supported "human" and "mouse"
dbname	

**Details**

If organism is not specified the entire table of orthologs (with all supported species) is returned.

**Value**

A data.frame object with ortholog information.

**Author(s)**

Diego Diez

**Examples**

```
getOrthologs(organism = "human")
```

---

getOrthologsFromBiomart  
*Returns ortholog genes for a target organism*

---

**Description**

Returns ortholog genes for a target organism

**Usage**

```
getOrthologsFromBiomart(eg, target_org, mart)
```

**Arguments**

eg	list of entrezgene ids to obtain orthologs.
target_org	target organism.
mart	mart object.

**Author(s)**

Diego Diez

---

getSequencesFromGenome

*Retrieves a set of sequences from a BSgenome object and optionally appends a label to each sequence id.*

---

**Description**

This is just a wrapper to getSeq() in package Biostrings that facilitates adding a label to each sequence.

**Usage**

```
getSequencesFromGenome(BED, genome, append.id)
```

**Arguments**

BED	file with peak locations in BED format.
genome	a BSgenome object (e.g. Mmusculus)
append.id	optional label to append to each sequence id.

**Author(s)**

Diego Diez

---

getSimilarityMatrix     *Compute similarity matrix of list of graphs.*

---

**Description**

This function computes pair-wise similarity based on common nodes (default) or edges between the graphs passed as a list.

**Usage**

```
getSimilarityMatrix(g_list, type = "edges")
```

**Arguments**

g\_list            list of graph objects.  
type             type of similarity, either node or edge (default).

**Author(s)**

Diego Diez

---

getTFclass            *Return the ontology in the TFclass database associated with an entrezgene identifier*

---

**Description**

Return the ontology in the TFclass database associated with an entrezgene identifier.

**Usage**

```
getTFclass(dbname = NULL)
```

**Arguments**

dbname            SQLite file to use as database.

**Author(s)**

Diego Diez

---

`getTFclassFromEntrezgene`*Applies getTFclass sequentially to a vector of entrezgene identifiers.*

---

**Description**

Applies getTFclass sequentially to a vector of entrezgene identifiers.

**Usage**

```
getTFclassFromEntrezgene(x, subset = "Class", tfclass, dbname = NULL)
```

**Arguments**

x	vector of entrezgene identifiers.
subset	level in the ontology (subset in TFclass terminology. By default "Class")
tfclass	data.frame with tfclass data to pass to the recursive function.
dbname	SQLite file to use as database.

**Author(s)**

Diego Diez

---

`getTFterms`*Get terms associated with a specified TFclass subset.*

---

**Description**

Returns a vector of names (not ids) with the members of a particular subset in the TFclass database. By default it returns the Class subset.

**Usage**

```
getTFterms(subset = "Class", dbname = NULL)
```

**Arguments**

subset	a subset in TFclass (default Class).
dbname	SQLite file to use as database.

**Author(s)**

Diego Diez

initBiomart                      *Initializes mart objects to identify ortholog genes*

---

**Description**

Initializes mart objects to identify ortholog genes

**Usage**

```
initBiomart(filter, biomart = "ensembl", host)
```

**Arguments**

filter                      list of supported organisms  
biomart  
host

**Author(s)**

Diego Diez

---

layout.arc                      *Layouts a graph using arcs.*

---

**Description**

Generates a layout for graphs that places in the center the target transcription factors, in the sides the enriched transcription factors and in between of them the bridge proteins.

**Usage**

```
layout.arc(g, target, query)
```

**Arguments**

g                              the graph object to layout.  
target                        list of target nodes (typically target transcription factors.)  
query                         list of query nodes (typically enriched transcription factors.)

**Value**

A matrix with the x and y locations of each node in the target graph.

**Author(s)**

Diego Diez



---

layout.concentric      *Generates a concentric layout for graphs*

---

**Description**

Generates a matrix with x,y coordinates for each node in a target graph, which layouts the nodes using concentric circles.

**Usage**

```
layout.concentric(g, concentric = NULL, radius = NULL, order.by)
```

**Arguments**

g	graph (igraph) to layout.
concentric	list with the components of each layer.
radius	radius of each layer.
order.by	graph attributes to order nodes by.

**Author(s)**

Diego Diez

---

plotDegree      *Plot degree distribution for network nodes*

---

**Description**

Plots the degree distribution and fits a power law, returning in the legend the values of the fitted parameters.

**Usage**

```
plotDegree(g)
```

**Arguments**

g	igraph object
---	---------------

**Author(s)**

Diego Diez

---

plotGraph                      *Plot an graph in igraph format.*

---

### Description

This function plots graphs of the class igraph.

### Usage

```
plotGraph(g, layout = layout.fruchterman.reingold, mar = .5, vertex.pch = 21, vertex.cex, vertex.col, v
```

### Arguments

<code>g</code>	a network object.
<code>layout</code>	graph layout, either a function or the output of a layout function.
<code>mar</code>	plot margin.
<code>vertex.pch</code>	node size.
<code>vertex.cex</code>	node size.
<code>vertex.col</code>	node line color.
<code>vertex.bg</code>	node background color.
<code>vertex.lwd</code>	node line width.
<code>edge.col</code>	edge color.
<code>edge.lwd</code>	edge line width.
<code>edge.lty</code>	edge line type.
<code>label</code>	logical; whether to plot labels.
<code>label.col</code>	label color.
<code>label.cex</code>	label expansion.
<code>label.pos</code>	label position.
<code>label.offset</code>	label offset.
<code>adjust.label.col</code>	whether to automatically adjust label color depending on the luminance of the node's color/s.
<code>normalize.layout</code>	whether to apply layout.norm (with limits xmin=-1, xmax=1, ymin=-1, ymax=1) to the layout.

### Author(s)

Diego Diez

---

plotTRM	<i>Plot an annotated TRM.</i>
---------	-------------------------------

---

### Description

This function plots the output `findTRM()` after it has been annotated with cluster information with `annotateTRM()` function. Cluster membership is plotted using a pie plot.

### Usage

```
plotTRM(g, layout = layout.fruchterman.reingold, mar = .5, vertex.col, vertex.cex, vertex.lwd, edge.col,
```

### Arguments

<code>g</code>	a network object with cluster information (attribute <code>piecolor</code> ).
<code>layout</code>	graph layout, either a function or the output of a layout function.
<code>mar</code>	plot margin.
<code>vertex.col</code>	node color.
<code>vertex.cex</code>	node size.
<code>vertex.lwd</code>	node border line width.
<code>edge.col</code>	edge color.
<code>edge.lwd</code>	edge line width.
<code>edge.lty</code>	edge line type.
<code>label</code>	logical; whether to plot labels.
<code>label.cex</code>	label expansion.
<code>label.col</code>	label color.
<code>label.pos</code>	label position.
<code>label.offset</code>	label offset.
<code>adjust.label.col</code>	whether to automatically adjust label color depending on the luminance of the node's color.
<code>normalize.layout</code>	whether to apply <code>layout.norm</code> (with limits <code>xmin=-1, xmax=1, ymin=-1, ymax=1</code> ) to the layout.

### Author(s)

Diego Diez

---

plotTRMlegend	<i>Plot the legend of a TRM with information about the cluster families.</i>
---------------	--

---

**Description**

This function just plots a legend with the cluster membership of the provided list of genes. The legend includes the most prominent families of each cluster and there is some name polishing as well.

**Usage**

```
plotTRMlegend(x, title = NULL, cex = 1)
```

**Arguments**

x	list of family names or igraph object.
title	title for the legend.
cex	numeric value controlling the size of the legend's text.

**Author(s)**

Diego Diez

---

processBiogrid	<i>Process a data.frame with BioGRID data into a network for a target organism</i>
----------------	--

---

**Description**

Process a data.frame with BioGRID data into a network for a target organism.

**Usage**

```
processBiogrid(dblast, org = "human", simplify = TRUE, type = "physical", mimic.old = FALSE)
```

**Arguments**

dblist	data.frame containing the BioGRID data.
org	target organism (default: "human")
simplify	whether to eliminate redundant edges (default TRUE)
type	type of interaction (physical or genetic) to include (default: "physical")
mimic.old	mimic old behavior of processBiogrid() when interactions for multiple species could be retrieved. Used only for testing.

**Value**

An igraph object.

**Author(s)**

Diego Diez

---

removeVertices	<i>Remove nodes from a graph and returns the largest component</i>
----------------	--

---

**Description**

Remove nodes from a graph and returns the largest component

**Usage**

```
removeVertices(g, filter, keep.hanging = FALSE)
```

**Arguments**

`g` graph to remove nodes.  
`filter`  
`keep.hanging` (logical) whether to return the largest component or not.

**Author(s)**

Diego Diez

---

writeTRMreport	<i>Export a table with TRM nodes and associated information.</i>
----------------	--

---

**Description**

This function generates a data.frame with the nodes in the provided graph and associated annotations.

**Usage**

```
writeTRMreport(graph, file, organism, target, query, sort.by = "symbol")
```

**Arguments**

graph	a graph object.
file	file name.
organism	organisms for the annotations.
target	target transcription factor.
query	query transcription factors.
sort.by	order the columns of the data.frame by (default: "symbol").

**Author(s)**

Diego Diez

# Index

- \* **datasets**
  - biogrid\_hs, [5](#)
  - biogrid\_mm, [5](#)
- \* **package**
  - rTRM-package, [2](#)
  
- annotateFreq, [3](#)
- annotateModule, [4](#)
- annotateTRM, [4](#)
  
- biogrid\_hs, [5](#)
- biogrid\_mm, [5](#)
  
- findTRM, [6](#)
  
- getAnnotations, [7](#)
- getBiogridData, [7](#)
- getConcentricList, [8](#)
- getLargestComp, [9](#)
- getMaps, [9](#)
- getMatrices, [10](#)
- getMotifsFromEntrezgene, [10](#)
- getMotifsFromSymbol, [11](#)
- getOrthologFromMatrix, [11](#)
- getOrthologs, [12](#)
- getOrthologsFromBiomart, [12](#)
- getSequencesFromGenome, [13](#)
- getSimilarityMatrix, [14](#)
- getTFclass, [14](#)
- getTFclassFromEntrezgene, [15](#)
- getTFterms, [15](#)
  
- initBiomart, [16](#)
  
- layout.arc, [16](#)
- layout.concentric, [17](#)
  
- plotDegree, [17](#)
- plotGraph, [18](#)
- plotTRM, [19](#)
- plotTRMlegend, [20](#)
  
- processBiogrid, [20](#)
  
- removeVertices, [21](#)
- rTRM (rTRM-package), [2](#)
- rTRM-package, [2](#)
  
- writeTRMreport, [21](#)