# Package 'RTNduals'

July 10, 2025

**Type** Package

**Title** Analysis of co-regulation and inference of 'dual regulons'

**Version** 1.32.0

**Author** Vinicius S. Chagas, Clarice S. Groeneveld, Gordon Robertson, Kerstin B. Meyer, Mauro A. A. Castro

**Maintainer** Mauro Castro <mauro.a.castro@gmail.com>, Clarice Groeneveld <clari.groeneveld@gmail.com>

**Depends** R(>= 3.6.3), RTN(>= 2.14.1), methods

**Imports** graphics, grDevices, stats, utils

**Suggests** knitr, rmarkdown, BiocStyle, RUnit, BiocGenerics

**Description**
RTNduals is a tool that searches for possible co-regulatory loops between regulon pairs generated by the RTN package. It compares the shared targets in order to infer 'dual regulons', a new concept that tests whether regulators can co-operate or compete in influencing targets.

**License** Artistic-2.0

**biocViews** GeneRegulation, GeneExpression, NetworkEnrichment, NetworkInference, GraphAndNetwork

**LazyData** TRUE

**VignetteBuilder** knitr

**RoxygenNote** 7.1.1

**Encoding** UTF-8

**NeedsCompilation** no

**git_url** https://git.bioconductor.org/packages/RTNduals

**git_branch** RELEASE_3_21

**git_last_commit** dcd82a6

**git_last_commit_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-07-09

# Contents



---

`RTNduals-package` *RTNduals: An R/Bioconductor package for analysis of co-regulation and inference of 'dual regulons'.*

---

## Description

RTNduals is a tool that searches for possible co-regulatory loops between regulon pairs generated by the RTN package. It compares the shared targets in order to infer 'dual regulons', a new concept that tests whether regulators can co-operate or compete in influencing targets.

## Details

| | |
|---|---|
| Package: | RTNduals |
| Type: | Package |
| Depends: | R (>= 3.5.0), methods, RTN |
| Imports: | grDevices, stats, utils |
| Suggests: | knitr, rmarkdown, BiocStyle, RUnit, BiocGenerics |
| License: | Artistic-2.0 |
| biocViews: | NetworkInference, NetworkEnrichment, GeneRegulation, GeneExpression, GraphAndNetwork |

## Index

Further information is available in the vignettes by typing `vignette("RTNduals")`. Documented topics are also available in HTML by typing `help.start()` and selecting the RTNduals package from the menu.

## Author(s)

Vinicius S. Chagas, Clarice S. Groeneveld, Kerstin B Meyer, Gordon Robertson, Mauro A. A. Castro

## References

Fletcher M.N.C. et al., *Master regulators of FGFR2 signalling and breast cancer risk.* Nature Communications, 4:2464, 2013.

Castro M.A.A. et al., *Regulators of genetic risk of breast cancer identified by integrative network analysis.* Nature Genetics, 48:12-21, 2016.

---

`MBR-class` *MBR objects*

---

## Description

MBR: an S4 class for co-regulation analysis and inference of 'dual regulons'.

## Details

The MBR class is a container for results from the MBR methods. The class slots are used to store information of different transcriptional networks, regulator annotation, infered 'dual regulons' and parameters used in the analysis. All the information is stored in nine slots.

## Slots

`TNI` a 'TNI' object created by the RTN package.

`regulatoryElements` regulatory elements listed in the TNI.

`dualRegulons` all possible 'duals regulons' computed by `mbrAssociation`

`results` a list, results from the MBR methods.

`para` a list, parameters used in the MBR methods.

`summary` a list, summary for 'para' and 'results'.

`status` a character vector specifying the status of the MBR object based on the available methods.

## Constructor

There is one constructors to create an MBR object: `tni2mbrPreprocess`;

mbrAssociation,MBR-method

*Motifs analysis and inference of 'dual regulons'.*

## Description

This function takes an MBR object and compares the shared regulon targets in order to test whether regulon pairs agree on the predicted downstream effects.

## Usage

```
## S4 method for signature 'MBR'
mbrAssociation(
  object,
  regulatoryElements = NULL,
  minRegulonSize = 15,
  doSizeFilter = FALSE,
  pValueCutoff = 0.001,
  pAdjustMethod = "bonferroni",
  estimator = "spearman",
  nPermutations = 1000,
  miFilter = TRUE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| object | A processed object of class MBR |
| regulatoryElements | An optional character vector specifying which 'TNI' regulatory elements should be evaluated. If 'NULL' all regulatory elements will be evaluated. |
| minRegulonSize | A single integer or numeric value specifying the minimum number of elements in a regulon. Gene sets with fewer than this number are removed from the analysis. |
| doSizeFilter | a logical value. If TRUE, negative and positive targets are independently verified by the 'minRegulonSize' argument. |
| pValueCutoff | a single numeric value specifying the cutoff for p-values considered significant. |
| pAdjustMethod | A single character value specifying the p-value adjustment method to be used (see 'p.adjust' function for details). |
| estimator | A character value specifying the estimator used in the association analysis. One of "spearman" (default), "kendall", or "pearson". |
| nPermutations | A single integer value specifying the number of permutations for deriving p-values associating regulon pairs. |
| miFilter | A single logical value specifying to apply the 'miFilter' between two regulators. |
| verbose | A single logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE). |

## Value

An MBR object with two data.frames in the slot 'results' listing the inferred 'dual regulons' and correspoding statistics.

## Examples

```
##--- load a dataset for demonstration
data("tniData", package = "RTN")
gexp <- tniData$expData
annot <- tniData$rowAnnotation
tfs <- c("IRF8","IRF1","PRDM1","E2F3","STAT4","LMO4","ZNF552")

##--- construct a tni object
rtni <- tni.constructor(gexp, regulatoryElements = tfs, rowAnnotation=annot)

##--- compute regulons
## set nPermutations>=1000
rtni <- tni.permutation(rtni, nPermutations=30)
## set nBootstrap>=100
rtni <- tni.bootstrap(rtni, nBootstrap=30)
## 'eps=NA' estimates threshold from empirical null
rtni <- tni.dpi.filter(rtni, eps=NA)

##--- construct a mbr object
rmbr <- tni2mbrPreprocess(rtni)

##--- run mbrAssociation
## set nPermutations>=1000
rmbr <- mbrAssociation(rmbr, pValueCutoff = 0.05, nPermutations=30)
```

---

# mbrGet,MBR-method *Get information from individual slots in MBR object.*

---

## Description

Get information from individual slots in an MBR object and any available results from previous analysis.

## Usage

```
## S4 method for signature 'MBR'
mbrGet(object, what = "status")
```

## Arguments

| | |
|---|---|
| object | A preprocessed object of class MBR |

what — a single character value specifying which information should be retrieved from the slots. Options: "TNI", "regulatoryElements", "dualRegulons", "results", "para", "summary", "status", "dualsCorrelation", "dualsOverlap", and "dualsCorMatrix"

## Value

Content from slots in the MBR object

## Examples

```
##--- load a dataset for demonstration
data("tniData", package = "RTN")
gexp <- tniData$expData
annot <- tniData$rowAnnotation
tfs <- c("IRF8","IRF1","PRDM1","E2F3","STAT4","LMO4","ZNF552")

##--- construct a tni object
rtni <- tni.constructor(gexp, regulatoryElements = tfs, rowAnnotation=annot)

##--- compute regulons
## set nPermutations>=1000
rtni <- tni.permutation(rtni, nPermutations=30)
## set nBootstrap>=100
rtni <- tni.bootstrap(rtni, nBootstrap=30)
## 'eps=NA' estimates threshold from empirical null
rtni <- tni.dpi.filter(rtni, eps=NA)

##--- construct a mbr object
rmbr <- tni2mbrPreprocess(rtni)

##--- get the 'TNI' slot using 'mbrGet'
tni <- mbrGet(rmbr, what="TNI")
```

---

# mbrPlotDuals *Plot shared targets between regulons.*

---

## Description

This function plots the shared targets for a regulon pair.

## Usage

```
mbrPlotDuals(
  object,
  dualreg,
  filepath = NULL,
  cols = c("#006400FF", "#CD6600FF")
)
```

## Arguments

| | |
|---|---|
| object | A processed object of class MBR evaluated by the method mbrAssociation. |
| dualreg | A string indicating the name of a dual regulon. |
| filepath | A string indicating the file path where the plot should be saved. |
| cols | A vector of length 2 indicating a diverging color scheme for negative and positive correlations, respectively. |

## Value

A plot showing targets of dual regulons.

## Examples

```
##--- load a dataset for demonstration
data("tniData", package = "RTN")
gexp <- tniData$expData
annot <- tniData$rowAnnotation
tfs <- c("IRF8","IRF1","PRDM1","E2F3","STAT4","LMO4","ZNF552")

##--- construct a tni object
rtni <- tni.constructor(gexp, regulatoryElements = tfs, rowAnnotation=annot)

##--- compute regulons
## set nPermutations>=1000
rtni <- tni.permutation(rtni, nPermutations=30)
## set nBootstrap>=100
rtni <- tni.bootstrap(rtni, nBootstrap=30)
## 'eps=NA' estimates threshold from empirical null
rtni <- tni.dpi.filter(rtni, eps=NA)

##--- construct a mbr object
rmbr <- tni2mbrPreprocess(rtni)

##--- run mbrAssociation
## set nPermutations>=1000
rmbr <- mbrAssociation(rmbr, pValueCutoff = 0.05, nPermutations=30)

## Not run:

##--- get inferred duals and plot the shared cloud of targets
duals <- mbrGet(rmbr, what="dualRegulons")
mbrPlotDuals(rmbr, dualreg=duals[1])


## End(Not run)
```

`mbrPlotInteraction` *Plot interaction effects between two continuous variables.*

## Description

This function plots the interaction effects between two continuous variables for linear, logistic, or Cox models.

## Usage

```
mbrPlotInteraction(
  model,
  vars,
  xlim = NULL,
  ylim = NULL,
  zlim = NULL,
  xlab = NULL,
  ylab = NULL,
  zlab = NULL,
  zcenter = NULL,
  zlog = NULL,
  zcols = c("#008080ff", "#d45500ff"),
  ycols = c("#4A97C9", "#D92522"),
  showdata = FALSE,
  datacols = "grey50",
  fname = "interplot",
  fpath = ".",
  width = 4.5,
  height = 4,
  plotype = c("3D", "2D"),
  plotpdf = FALSE
)
```

## Arguments

| | |
|---|---|
| `model` | An object of class 'lm', 'glm', or 'coxph'. |
| `vars` | A character vector of length 2 with the names of two varibles in the 'model'. |
| `xlim` | A numeric vector of length 2, i.e. xlim = c(x1, x2), indicating the x limits of the plot. If xlim = NULL, it will be derevided from the observed data ranges. |
| `ylim` | A numeric vector of length 2, i.e. ylim = c(y1, y2), indicating the y limits of the plot. If ylim = NULL, it will be derevided from the observed data ranges. |
| `zlim` | A numeric vector of length 2, i.e. zlim = c(z1, z2), indicating the z limits of the plot. If zlim = NULL, it will be derevided from the observed data ranges. |
| `xlab` | A string with the label for the x-axis. |
| `ylab` | A string with the label for the y-axis. |

| | |
|---|---|
| zlab | A string with the label for the z-axis. |
| zcenter | A numeric value indicating a z value to center the color scale. |
| zlog | A logical value indicating whether the z axis is to be logarithmic. |
| zcols | A vector of length 2 indicating a diverging color scheme for the z-axis varible. |
| ycols | A vector of length 2 indicating a diverging color scheme for the y-axis varible (only used when type='2D'). |
| showdata | A logical value indicating whether to show the original data used to fit linear model. |
| datacols | When 'showdata = TRUE', this can be a named vector of colors for data points (names should match samples in the 'model' object). Alternativelly, it can be a single color value. |
| fname | A string. The name of the PDF file which will contain the plot. |
| fpath | A string. The directory where the file will be saved. |
| width | A numeric value. The width of the plot. |
| height | A numeric value. The height of the plot. |
| plotype | A string indicating '2D' of '3D' plot type. If plotype = '2D', the z-axis (and all related parameters) is transposed to the y-axis. |
| plotpdf | A logical value. |

## Value

A interaction plot.

## Examples

```
#-- Example of simulated data, with response variable modelled by:
#-- (1) Main effects of 'reg1' and 'reg2'
#-- (2) Interaction effects between 'reg1' and 'reg2'
#-- (3) Additional random uniform noise
reg1 <- rnorm(1000,0,2)
reg2 <- rnorm(1000,0,2)
response <- 3*reg1 + 2*reg1*reg2 + runif(1000,0,2)
dataset <- data.frame(reg1, reg2, response)
model <- lm(response ~ reg1*reg2, data=dataset)
mbrPlotInteraction(model, vars=c("reg1","reg2"))
```

---

# mbrPriorEvidenceTable,MBR-method

*Entry point for external evidences.*

---

## Description

If available, this function adds external evidences to an 'MBR' object.

## Usage

```
## S4 method for signature 'MBR'
mbrPriorEvidenceTable(
  object,
  priorEvidenceTable,
  evidenceColname,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| object | A processed object of class MBR evaluated by the method mbrAssociation. |
| priorEvidenceTable | An 'data.frame' with three columns representing (1) regulatory elements 1, (2) regulatory elements 2, and (3) external evidences between the regulatory elements. |
| evidenceColname | A single character value specifying a column in the 'priorEvidenceTable'. |
| verbose | A single logical value specifying to display detailed messages (when verbose=TRUE) or not (when verbose=FALSE). |

## Value

An MBR object with an updated 'data.frame' in the slot 'results' listing the input additional evidences.

## Examples

```
##--- load a dataset for demonstration
data("tniData", package = "RTN")
gexp <- tniData$expData
annot <- tniData$rowAnnotation
tfs <- c("IRF8","IRF1","PRDM1","E2F3","STAT4","LMO4","ZNF552")

##--- construct a tni object
rtni <- tni.constructor(gexp, regulatoryElements = tfs, rowAnnotation=annot)

##--- compute regulons
## set nPermutations>=1000
rtni <- tni.permutation(rtni, nPermutations=30)
## set nBootstrap>=100
rtni <- tni.bootstrap(rtni, nBootstrap=30)
## 'eps=NA' estimates threshold from empirical null
rtni <- tni.dpi.filter(rtni, eps=NA)

##--- construct a mbr object
rmbr <- tni2mbrPreprocess(rtni)

##--- run mbrAssociation
## set nPermutations>=1000
```

```
rmbr <- mbrAssociation(rmbr, pValueCutoff = 0.05, nPermutations=30)

##--- check results
results <- mbrGet(rmbr, what="dualsCorrelation")

##--- add supplementary evidence table
## here we build a 'toy' example using the 'rnorm' function
## for demonstration purposes only!
priorEvidenceTable <- results[,c("Regulon1","Regulon2")]
priorEvidenceTable$ToyEvidence <- rnorm(nrow(results))
priorEvidenceTable

##--- add supplementary evidences
# rmbr <- mbrPriorEvidenceTable(rmbr, priorEvidenceTable=priorEvidenceTable, evidenceColname = "ToyEvidence")

##--- check updated results
# mbrGet(rmbr, what="dualsCorrelation")
```

---

`tni2mbrPreprocess,TNI-method`

*A preprocessing function for objects of class MBR.*

---

## Description

This function converts a TNI class objects and into one MBR class object.

## Usage

```
## S4 method for signature 'TNI'
tni2mbrPreprocess(tni, regulatoryElements = NULL)
```

## Arguments

| | |
|---|---|
| `tni` | A 'TNI' class object. |
| `regulatoryElements` | An optional character vector specifying which 'TNI' regulatory elements should be evaluated. If 'NULL' all regulatory elements will be evaluated. |

## Value

An MBR object.

## Examples

```
##--- load a dataset for demonstration
data("tniData", package = "RTN")
tfs <- c("IRF8","IRF1","PRDM1","E2F3","STAT4","LMO4","ZNF552")
```

```
##--- construct a tni object
rtni <- tni.constructor(tniData$expData, regulatoryElements = tfs,
rowAnnotation=tniData$rowAnnotation)

##--- compute regulons
## set nPermutations>=1000
rtni <- tni.permutation(rtni, nPermutations=30)

## set nBootstrap>=100
rtni <- tni.bootstrap(rtni, nBootstrap=30)

## 'eps=NA' estimates threshold from empirical null
rtni <- tni.dpi.filter(rtni, eps=NA)

##--- construct a mbr object
rmbr <- tni2mbrPreprocess(rtni)
```

# Index