

# Package ‘epiNEM’

February 3, 2025

**Type** Package

**Title** epiNEM

**Version** 1.30.0

**Author** Madeline Diekmann & Martin Pirkl

**Maintainer** Martin Pirkl <martinpirkl@yahoo.de>

**Description** epiNEM is an extension of the original Nested Effects Models (NEM). EpiNEM is able to take into account double knockouts and infer more complex network signalling pathways. It is tailored towards large scale double knock-out screens.

**Depends** R (>= 4.1)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**biocViews** Pathways, SystemsBiology, NetworkInference, Network

**RoxygenNote** 7.3.1

**Imports** BoutrosLab.plotting.general, BoolNet, e1071, gtools, stats, igraph, utils, lattice, latticeExtra, RColorBrewer, pcalg, minet, grDevices, graph, mnem, latex2exp

**VignetteBuilder** knitr

**Suggests** knitr, RUnit, BiocGenerics, STRINGdb, devtools, rmarkdown, GOSemSim, AnnotationHub, org.Sc.sgd.db, BiocStyle

**BugReports** <https://github.com/cbg-ethz/epiNEM/issues>

**URL** <https://github.com/cbg-ethz/epiNEM/>

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

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** befa45a

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2025-02-03

## Contents

AddLogicGates . . . . .	2
CreateExtendedAdjacency . . . . .	3
CreateRandomGraph . . . . .	3
CreateTopology . . . . .	4
epiAnno . . . . .	4
epiNEM . . . . .	5
epiScreen . . . . .	6
ExtendTopology . . . . .	7
GenerateData . . . . .	8
HeatmapOP . . . . .	9
Mll . . . . .	11
perm.rank.test . . . . .	12
plot.epiNEM . . . . .	12
plot.epiScreen . . . . .	13
plot.epiSim . . . . .	14
rank.enrichment . . . . .	14
sameith_GO . . . . .	16
sameith_string . . . . .	16
samscreen . . . . .	16
sim . . . . .	17
SimEpiNEM . . . . .	17
wageningen_GO . . . . .	18
wageningen_string . . . . .	18
wagscreen . . . . .	19
<b>Index</b>	<b>20</b>

---

AddLogicGates	<i>Add logic.</i>
---------------	-------------------

---

### Description

extend model with node representing logic gate

### Usage

```
AddLogicGates(child, logic, model)
```

### Arguments

child	define the child
logic	define the logical gate
model	normal model

### Value

model list with additional logic gate

**Examples**

```
model <- CreateRandomGraph(c("Ikk1", "Ikk2", "RelA"))
model2 <- AddLogicGates("RelA", "OR", model)
```

---

CreateExtendedAdjacency

*Create an extended adjacency matrix*

---

**Description**

extend adjacency matrices taking cycles and logics into account. For every given start state, the final state is computed using BoolNet.

**Usage**

```
CreateExtendedAdjacency(network, mutants, experiments)
```

**Arguments**

network	network created by BoolNet from file
mutants	vector of single knockouts
experiments	vector of all knockouts

**Value**

extended adjacency matrix

**Examples**

```
library(BoolNet)
data(cellcycle)
extModel <- CreateExtendedAdjacency(cellcycle,
c(cellcycle$genes, "CycD.Rb"), cellcycle$genes)
```

---

CreateRandomGraph

*Create a random graph*

---

**Description**

Returns a model graph with randomly sampled edges. Every possible edge has a probability to exist in the graph.

**Usage**

```
CreateRandomGraph(pathwayGenes, edgeProb = 0.5)
```

**Arguments**

pathwayGenes	vector of genes in the pathway
edgeProb	probability of random edge

**Value**

adjacency matrix

**Examples**

```
graph <- CreateRandomGraph(c("Ikk1", "Ikk2", "RelA"))
```

---

CreateTopology            *Create Topology.*

---

**Description**

Create topology for a randomly generated pathway topology

**Usage**

```
CreateTopology(single, double, force = TRUE)
```

**Arguments**

single	number of single knockouts
double	number of double knockouts
force	if true the random model will have a sophisticated logical gate

**Value**

adjacency matrix

**Examples**

```
model <- CreateTopology(3, 1)
```

---

epiAnno                    *Gate visualisation.*

---

**Description**

Plots logical gate data annotation. The 8 heatmaps visualize what perfect data would look like in respective to each logical gate. Perfect data is equivalent to Boolean truth tables.

**Usage**

```
epiAnno()
```

**Value**

plot of heatmaps showing the silencing scheme (=expected data, truth tables)

**Author(s)**

Martin Pirkl

**References**[https://en.wikipedia.org/wiki/Boolean\\_algebra](https://en.wikipedia.org/wiki/Boolean_algebra)**Examples**

epiAnno()

---

epiNEM*Epistatic NEMs - main function.*

---

**Description**

This function contains the inference algorithm to learn logical networks from knock-down data including double knock-downs.

**Usage**

```
epiNEM(
  filename = "random",
  method = "greedy",
  nIterations = 10,
  nModels = 0,
  random = list(single = 4, double = 1, reporters = 100, FPrate = 0.1, FNrate = 0.1,
    replicates = 1),
  ltype = "marginal",
  para = c(0.13, 0.05),
  init = NULL
)
```

**Arguments**

filename	A binary, tab-delimited matrix. Columns: single and double knockdowns. Rows: genes showing effect or not? Default: random; artificial data is generated to 'random' specifications
method	greedy or exhaustive search. Default: greedy
nIterations	number of iterations. Default: 10
nModels	number of Models. Default: 0
random	list specifying how the data should be generated: no. of single mutants, no. of double mutants, no. of reporterGenes, FP-rate, FN-rate, no. of replicates
ltype	likelihood either "marginal" or "maximum"
para	false positive and false negative rates
init	adjacency matrix to initialise the greedy search

**Value**

List object with an adjacency matrix denoting the network, the model of the silencing scheme (rows are knock-downs, columns are signalling genes), a string with the inferred logical gates, a column indices denoting position of logical gates, the log transformed likelihood and the effect reporter distribution (rows are the signalling genes including the null node).

**Author(s)**

Madeline Diekmann

**See Also**

nem

**Examples**

```
data <- matrix(sample(c(0,1), 100*4, replace = TRUE), 100, 4)
colnames(data) <- c("A", "A.B", "B", "C")
rownames(data) <- paste("E", 1:100, sep = "_")
res <- epiNEM(data, method = "exhaustive")
plot(res)
```

---

epiScreen

*Analyse large double knock-out screen.*

---

**Description**

This function is used to analyse knock-out screens with multiple double and single knock-outs combined in one data set.

**Usage**

```
epiScreen(data, ...)
```

**Arguments**

data	data matrix containing multiple single and double knock-downs in columns and effect reporters in the rows
...	additional parameters, e.g. for the main epiNEM function

**Value**

list object with vectors of double knock-downs, single knock-downs and two matrices with doubles in the columns and singles in the rows. The first matrix denotes the respective logical gate for the triple and the second matrix the log-likelihood

**Author(s)**

Martin Pirkl

**Examples**

```
data <- matrix(sample(c(0,1), 100*9, replace = TRUE), 100, 9)
colnames(data) <- c("A.B", "A.C", "B.C", "A", "B", "C", "D", "E", "G")
rownames(data) <- paste("E", 1:100, sep = "_")
res <- epiScreen(data)
```

---

ExtendTopology

*Extending topology of normal "nem"*

---

**Description**

Extending topology of normal "nem"

**Usage**

```
ExtendTopology(topology, nReporters)
```

**Arguments**

topology            model of a topology from CreateTopology  
nReporters         number of effects reporters

**Value**

extended topology in which reporters are linked to pathway genes

**Author(s)**

Madeline Diekmann

**See Also**

CreateTopology

**Examples**

```
topology <- CreateTopology(3, 1, force = TRUE)
topology <- unlist(unique(topology), recursive = FALSE)
extTopology <- ExtendTopology(topology$model, 100)
```

---

GenerateData	<i>Generate data from extended model.</i>
--------------	---

---

### Description

Given a model created from `CreateTopology` and `ExtendTopology`, this function creates a corresponding artificial data matrix, which is used as a ground truth for simulation studies.

### Usage

```
GenerateData(model, extTopology, FPrate, FNrate, replicates)
```

### Arguments

<code>model</code>	model of a topology from <code>CreateTopology</code>
<code>extTopology</code>	extended topology
<code>FPrate</code>	false positive rate
<code>FNrate</code>	false negative rate
<code>replicates</code>	number of replicates

### Value

data matrix with effect reporters as rows and knock-downs (including double knock-downs) as columns.

### Author(s)

Madeline Diekmann

### See Also

`CreateTopology`

### Examples

```
topology <-  
CreateTopology(3, 1, force = TRUE)  
topology <-  
unlist(unique(topology), recursive = FALSE)  
extTopology <-  
ExtendTopology(topology$model, 100)  
sortedData <-  
GenerateData(topology$model, extTopology, 0.05, 0.13, 3)
```



---

HeatmapOP

*Heatmap.*

---

## Description

Heatmap function based on the lattice package more information: `?xyplot`

## Usage

```
HeatmapOP(  
  x,  
  col = "RdYlGn",  
  colNA = "grey",  
  coln = 11,  
  bordercol = "grey",  
  borderwidth = 0.1,  
  breaks = "sym",  
  main = "",  
  sub = "",  
  dendrogram = "none",  
  colorkey = "right",  
  Colv = TRUE,  
  Rowv = TRUE,  
  xrot = 90,  
  yrot = 0,  
  shrink = c(1, 1),  
  cexCol = 1,  
  cexRow = 1,  
  cexMain = 1,  
  cexSub = 1,  
  colSideColors = NULL,  
  aspect = "fill",  
  contour = FALSE,  
  useRaster = FALSE,  
  xlab = NULL,  
  ylab = NULL,  
  colSideColorsPos = "top",  
  clust = NULL,  
  clusterx = NULL,  
  axis.padding = 0.5,  
  legend = NULL,  
  ...  
)
```

## Arguments

x	Matrix.
col	Color. See <code>brewer.pal.info</code> for all available color schemes. Alternatively, any number of colors, which are then used to create a color gradient. E.g., <code>c('blue','red')</code> produces a color scheme with a gradient from blue to red.

<code>colNA</code>	color for NAs; default is grey
<code>coln</code>	Number of colors.
<code>bordercol</code>	Border color.
<code>borderwidth</code>	Border width.
<code>breaks</code>	Defines the breaks in the color range. "sym" makes the breaks symmetric around 0.
<code>main</code>	Main title.
<code>sub</code>	Subtitle.
<code>dendrogram</code>	Draw dendrogram with "both", "col" or "row", or do not draw with "none".
<code>colorkey</code>	Draw colorkey "left", "right" (default), "top", "bottom" or NULL for no colorkey. See <code>?lattice::levelplot</code> for more complex colorkey options.
<code>Colv</code>	Cluster columns (TRUE) or not (FALSE).
<code>Rowv</code>	Cluster rows (TRUE) or not (FALSE).
<code>xrot</code>	Rotate the column names by degree.
<code>yrot</code>	Rotate the row names by degree.
<code>shrink</code>	<code>c(x,y)</code> defines a range of size for the data boxes from low to high.
<code>cexCol</code>	Font size of column names.
<code>cexRow</code>	Font size of row names.
<code>cexMain</code>	Font size of main title.
<code>cexSub</code>	Font size of subtitle.
<code>colSideColors</code>	Defines a numeric vector to annotate columns with different colors.
<code>aspect</code>	"iso" for quadratic boxes or "fill" for stretched boxes.
<code>contour</code>	TRUE adds a contour plot.
<code>useRaster</code>	TRUE to add raster visuals
<code>xlab</code>	Label for the x-axis.
<code>ylab</code>	Label for the y-axis.
<code>colSideColorsPos</code>	Place <code>colSideColors</code> at the "top" or "bottom".
<code>clust</code>	p, s, or k for correlation clustering
<code>clusterx</code>	Optional data matrix y with the same dimensions as x. x's columns or rows are sorted by the cluster information of y. Col- and rownames of y must be in the same order as in x.
<code>axis.padding</code>	padding around the heatmap (0.5 is no padding, default)
<code>legend</code>	list object. For parameters see base function <code>?legend</code> for details. x and y parameters are relative to the inside of the heatmap and are between 0 and 1. E.g., to place the legend outside of the heatmap x and y need to be either less than 0 or greater than 1.
<code>...</code>	Optional arguments.

**Value**

lattice object/matrix

**Author(s)**

Martin Pirkl & Oscar Perpinan at <http://oscarperpinan.github.io/rastervis/>

**Examples**

```
x <- matrix(rnorm(50), 10, 5)
HeatmapOP(x, dendrogram = "both", aspect = "iso", xrot = 45)
```

---

M11

*Evaluation of graphs*

---

**Description**

Computes marginal log-likelihood for model Phi given observed data matrix D1

**Usage**

```
M11(Phi, D1, D0, ltype = "marginal", para = c(0.13, 0.05))
```

**Arguments**

Phi	model to be evaluated
D1	observed data matrix
D0	complementary D1
ltype	likelihood type either "marginal" or "maximum"
para	false positive and false negative rates

**Value**

list with likelihood poster probability, egene positions

**Examples**

```
Phi <- matrix(sample(c(0,1), 9, replace = TRUE), 3, 3)
data <- matrix(sample(c(0,1), 3*10, replace = TRUE), 10, 3)
rownames(Phi) <- colnames(Phi) <- colnames(data) <- c("Ikk1", "Ikk2", "Re1A")
score <- M11(Phi, D1 <- data, D0 <- 1 - data)
```

---

perm.rank.test	<i>AUC permutation test</i>
----------------	-----------------------------

---

**Description**

computes the area under the rank enrichment score curve and does a permutation test to compute the p-value

**Usage**

```
perm.rank.test(
  x,
  y = NULL,
  alternative = c("two.sided", "less", "greater"),
  iter = 1000
)
```

**Arguments**

x	numeric vector of ranks
y	numeric vector of the superset of x
alternative	character for test type: 'less', 'greater', 'two.sided'
iter	integer number of iterations

**Value**

p-value

**Author(s)**

Martin Pirkl

**Examples**

```
x <- 1:10
y <- 1:100
perm.rank.test(x,y,alternative='less')
perm.rank.test(x,y,alternative='greater')
```

---

plot.epiNEM	<i>Plot pathway.</i>
-------------	----------------------

---

**Description**

Plots the winning pathway structure

**Usage**

```
## S3 method for class 'epiNEM'
plot(x, ...)
```

**Arguments**

x                    object of class epiNEM  
 ...                 other arguments

**Value**

plot of the logical network

**Examples**

```
data <- matrix(sample(c(0,1), 100*4, replace = TRUE), 100, 4)
colnames(data) <- c("A", "A.B", "B", "C")
rownames(data) <- paste("E", 1:100, sep = "_")
res <- epiNEM(data, method = "exhaustive")
plot(res)
```

---

plot.epiScreen            *Plot screen.*

---

**Description**

Plots the sresults of a systematic knock-out screen

**Usage**

```
## S3 method for class 'epiScreen'
plot(
  x,
  global = TRUE,
  ind = NULL,
  colorkey = TRUE,
  cexGene = 1,
  off = 0.05,
  cexLegend = 1,
  ...
)
```

**Arguments**

x                    object of class epiScreen  
 global             plot global distribution or for each pair (FALSE)  
 ind                 index of pairs to plot  
 colorkey           if TRUE prints colorkey  
 cexGene            size of modulator annotation  
 off                 relative distance from the gene names to the respective likelihoods  
 cexLegend         font size of the legend  
 ...                 other arguments

**Value**

plot(s) of an epiNEM screen analysis

**Examples**

```
data <- matrix(sample(c(0,1), 100*9, replace = TRUE), 100, 9)
colnames(data) <- c("A.B", "A.C", "B.C", "A", "B", "C", "D", "E", "G")
rownames(data) <- paste("E", 1:100, sep = "_")
res <- epiScreen(data)
plot(res)
plot(res, global = FALSE, ind = 1:3)
```

---

plot.epiSim	<i>Plot simulations.</i>
-------------	--------------------------

---

**Description**

Plots the simulation results

**Usage**

```
## S3 method for class 'epiSim'
plot(x, ...)
```

**Arguments**

x	object of class epiSim
...	other arguments

**Value**

plot(s) of an epiNEM simulation analysis

**Examples**

```
res <- SimEpiNEM(runs = 1)
plot(res)
```

---

rank.enrichment	<i>Rank enrichment</i>
-----------------	------------------------

---

**Description**

Infers a signalling pathway from peerturbation experiments.

**Usage**

```
rank.enrichment(
  data,
  list,
  list2 = NULL,
  n = 1000,
  main = NULL,
  col1 = "RdBu",
  col2 = rgb(1, 0, 0, 0.75),
  col3 = rgb(0, 0, 1, 0.75),
  blim = NULL,
  p = NULL,
  lwd = 3,
  test = wilcox.test,
  vis = "matrix",
  verbose = FALSE,
  ...
)
```

**Arguments**

data	m times l matrix with m observed genes and l variables with numeric values to rank the genes
list	list of of vectors of genes
list2	optional list with same length as list
n	length of the gradient (maximum: m)
main	character string for main header; if NULL uses the column names of data by default
col1	color of the gradient
col2	color of the first list
col3	color of the second list2
blim	numeric vector of length two with the lower and upper bounds for the gradient
p	numeric adjustment (length four) of the left side of the gradient (low means more to the left, high more to the right) the right side of the enrichment lines and the top positions of the additional matrices in case of vis='matrices'
lwd	line width of the enrichment lines
test	test function for the enrichment p-value; must have input argument and output values same as perm.rank.test; e.g., wilcox.test or ks.test (here 'less' and 'greater' are switched!)
vis	method for visualisation: 'matrix' uses one matrix heatmap for; 'matrices' uses several matrices (experimental), 'colside' uses the colSideColors argument for the ticks of genes in list/list2 (can use a lot of memory; experimental)
verbose	if TRUE gives prints additional output
...	additional arguments for epiNEM::HeatmapOP

**Value**

transitively closed matrix or graphNEL

**Author(s)**

Martin Pirkl

**Examples**

```
data <- matrix(rnorm(100*2),100,2)
rownames(data) <- 1:100
colnames(data) <- LETTERS[1:2]
list <- list(first = as.character(sample(1:100, 10)), second = as.character(sample(1:100, 20)))
rank.enrichment(data,list)
```

---

sameith_GO	<i>graph-based GO similarity scores, string GO annotations for Sameith et al., 2015 data</i>
------------	--

---

**Description**

The data consists of lists including epiNEM identified and general similarity scores and GO annotations for each triple. For details see the vignette.

**Examples**

```
data(sameith_GO)
```

---

sameith_string	<i>sig. of string interaction scores for Sameith et al., 2015 data</i>
----------------	--

---

**Description**

The data consists of a list including a vectors of pairs (for interactions) and a corresponding list of interaction scores derived from the string database. For details see the vignette.

**Examples**

```
data(sameith_string)
```

---

samscreen	<i>Example data: epiNEM results for the Sameith et al., 2015 knock-out screen</i>
-----------	---

---

**Description**

The result of the epiNEM analysis of the data from "[http://www.holstegelab.nl/publications/sv/signaling\\_redundancy/dov](http://www.holstegelab.nl/publications/sv/signaling_redundancy/dov)"

The data consists of a list of matrices with the likelihoods (ll) for each analysed triple of signalling genes and the inferred logic (logic) for each triple. The signalling genes or modulators C are the rows and the signalling genes from the double knock-downs are in the columns. For details see the vignette.

**Examples**

```
data(samscreen)
```



sim

*Example data: simulation results***Description**

Contains simulation results. How they were acquired is explained in the vignette. The data consists of a list of data matrices holding sensitivity and specificity (spec, sens) of network edges for the various methods compared to the ground truth, sensitivity and specificity (sens2, spec2) of the expected data for epiNEM and Boolean NEMs and accuracy of the inferred logics for both. The different methods are in the rows and the columns denote the different independent simulation runs.

**Examples**

```
data(sim)
```

SimEpiNEM

*Compare algorithms.***Description**

Compares different network reconstruction algorithm on simulated data.

**Usage**

```
SimEpiNEM(
  runs = 10,
  do = c("n", "e"),
  random = list(FPrate = 0.1, FNrate = c(0.1, 0.5), single = 3, double = 1, reporters =
    10, replicates = 2),
  maxTime = FALSE,
  forcelogic = TRUE,
  epinemsearch = "greedy",
  bnemsearch = "genetic",
  ...
)
```

**Arguments**

runs	number simulation runs
do	string vector of algorithms to compare: e (epiNEM), n (Nested Effects Models), b (B-NEM), p (PC algorithm), a (Aracne), e.g. c("e", "n", "p")
random	list of false positive rate FPrate, false negative rates FNrate, number of single knock-downs single, number of double knock-downs double, number of effect reporters reporters and number of replicates replicates
maxTime	TRUE if the algorithms are bound to a maximum running time in respect to epiNEM
forcelogic	if TRUE the randomly sampled ground truth network includes a complex logic with probability 1

```
epinemsearch  greedy or exhaustive search for epiNEM
bnemsearch    genetic or greedy search for B-NEM
...          additional parameters
```

**Value**

returns list of specificity and sensitivity of inferred edges (spec, sens) and inferred expected data (spec2, sens2) and accuracy of logics (logics) and running time (time)

**Author(s)**

Martin Pirkl

**Examples**

```
res <- SimEpiNEM(runs = 1)
```

---

wageningen_GO	<i>graph-based GO similarity scores, string GO annotations for van Wageningen et al., 2015 data</i>
---------------	---

---

**Description**

The data consists of lists including epiNEM identified and general similarity scores and GO annotations for each triple. For details see the vignette.

**Examples**

```
data(wageningen_GO)
```

---

wageningen_string	<i>sig. of string interaction scores for van Wageningen et al., 2010 data</i>
-------------------	---

---

**Description**

The data consists of a list including a vectors of pairs (for interactions) and a corresponding list of interaction scores derived from the string database. For details see the vignette.

**Examples**

```
data(wageningen_string)
```

---

wagscreen

*Example data: epiNEM results for the Wageningen et al., 2010 knock-out screen*  
*"[http://www.holstegelab.nl/publications/GSTF\\_geneticinteractions/downloads/del\\_mutants\\_limma.txt](http://www.holstegelab.nl/publications/GSTF_geneticinteractions/downloads/del_mutants_limma.txt)"*

---

### **Description**

The data consists of a list of matrices with the likelihoods (ll) for each analysed triple of signalling genes and the inferred logic (logic) for each triple. The signalling genes or modulators C are the rows and the signalling genes from the double knock-downs are in the columns. For details see the vignette.

### **Examples**

```
data(wagscreen)
```

# Index

AddLogicGates, [2](#)

CreateExtendedAdjacency, [3](#)  
CreateRandomGraph, [3](#)  
CreateTopology, [4](#)

epiAnno, [4](#)  
epiNEM, [5](#)  
epiScreen, [6](#)  
ExtendTopology, [7](#)

GenerateData, [8](#)

HeatmapOP, [9](#)

M11, [11](#)

perm.rank.test, [12](#)  
plot.epiNEM, [12](#)  
plot.epiScreen, [13](#)  
plot.epiSim, [14](#)

rank.enrichment, [14](#)

sameith\_GO, [16](#)  
sameith\_string, [16](#)  
samscreen, [16](#)  
sim, [17](#)  
SimEpiNEM, [17](#)

wageningen\_GO, [18](#)  
wageningen\_string, [18](#)  
wagscreen, [19](#)