Package 'nem'

April 5, 2014

Title (Dynamic) Nested Effects Models and Deterministic Effects Propagation Networks to reconstruct phenotypic hierarchies

Version 2.36.0

Author Holger Froehlich, Florian Markowetz, Achim Tresch, Theresa Niederberger, Christian Bender, Matthias Maneck, Claudio Lottaz, Tim Beissbarth

Description The package 'nem' allows to reconstruct features of pathways from the nested structure of perturbation effects. It takes as input (1.) a set of pathway components, which were perturbed, and (2.) phenotypic readout of these perturbations (e.g. gene expression, protein expression). The output is a directed graph representing the phenotypic hierarchy.

Reference Froehlich H, Beissbarth T, Tresch A, Kostka D, Jacob J,Spang R, Markowetz F: Analyzing gene perturbation screens with
nested effects models in R and bioconductor, Bioinformatics,2008, 24:2549-50.

Maintainer Holger Froehlich <frohlich@bit.uni-bonn.de>

Depends R (>= 2.0), e1071 (>= 1.5), graph (>= 1.24), plotrix, limma,cluster (>= 1.11), statmod, Hmisc, Rgraphviz

Enhances doMC, Rglpk

Imports boot, e1071, graph, graphics, grDevices, methods, RBGL (>= 1.8.1), RColorBrewer, stats, utils, Rgraphviz

Suggests Biobase (>= 1.10)

LazyLoad Yes

biocViews Microarray, Bioinformatics, GraphsAndNetworks, Pathways

URL http://www.bioconductor.org

License GPL (>= 2)

R topics documented:

BFSlevel . . . 2
BoutrosRNAi2002 . . . 3
closest.transitive.greedy . . . 4
enumerate.models . . . 5
generateNetwork . . . 6
getDensityMatrix . . . 8
getRelevantEGenes . . . 9
infer.edge.type . . . 10
internal . . . 12
Ivanova2006RNAiTimeSeries . . . 12
local.model.prior . . . 13
nem . . . 14
nem.bootstrap . . . 17
nem.calcSignificance . . . 18
nem.consensus . . . 19
nem.cont.preprocess . . . 21
nem.discretize . . . 22
nem.jackknife . . . 24
nemModelSelection . . . 25
network.AIC . . . 26
NiederbergerMediator2012 . . . 27
plot.nem . . . 28
plotEffects . . . 29
prior.EgeneAttach.EB . . . 31
prune.graph . . . 32
quicknem . . . 33
SahinRNAi2008 . . . 35
SCCgraph . . . 36
set.default.parameters . . . 37
sim.intervention . . . 40
subsets . . . 41
transitive.closure . . . 42
transitive.projections . . . 43
transitive.reduction . . . 44

Index **45**

`BFSlevel` *Build (generalized) hierarchy by Breath-First Search*

Description

`BFSlevel` builds a (generalized) hierarchy by Breath-First Search as described in (Yu and Gerstein, 2006)

Usage

```
BFSlevel(g,verbose=TRUE)
```

Arguments

g	graphNEL object
verbose	Default: TRUE

Details

Haiyuan Yu and Mark Gerstein: Genomic analysis of the hierarchical structure of regulatory networks, PNAS 103(40):14724-14731, 2006

Value

level	vector of levels for each node

Author(s)

Florian Markowetz

Examples

```
## bla
```

BoutrosRNAi2002 *RNAi data on Drosophila innate immune response*

Description

Data from a study on innate immune response in *Drosophila* (Boutros et al, 2002). Selectively removing signaling components by RNAi blocked induction of all, or only parts, of the transcriptional response to LPS. The nested structure of perturbation effects allows to reconstruct a branching in the Imd pathway.

Usage

```
data(BoutrosRNAi2002)
```

Format

BoutrosRNAiExpression: data matrix: 14010 x 16\ BoutrosRNAiDiscrete: binary matrix: 68 x 16\ BoutrosRNAiDens: data matrix: 68 x 4\ BoutrosRNAiLods: data matrix: 68 x 4\ BoutrosRNAiLogFC: data matrix: 68 x 4

Details

The dataset consists of 16 Affymetrix-microarrays: 4 replicates of control experiments without LPS and without RNAi (negative controls), 4 replicates of expression profiling after stimulation with LPS but without RNAi (positive controls), and 2 replicates each of expression profiling after applying LPS and silencing one of the four candidate genes tak, key, rel, and mkk4/hep.

`BoutrosRNAiExpression`: For preprocessing we performed normalization on probe level using a variance stabilizing transformation (Huber et al, 2002), and probe set summarization using a median polish fit of an additive model (Irizarry et al, 2003).

`BoutrosRNAiDiscrete`: contains only the 68 genes more than two-fold up-regulated between negative and positive controls. The continuous expression values are discretized to `1` (effect: closer to negative controls) and `0` (no effect: closer to positive controls).

`BoutrosRNAiDens`: log p-value density matrix for the 68 genes with more than two-fold up-regulated between negative and positive controls.

`BoutrosRNAiLods`: B-value matrix for the 68 genes with more than two-fold up-regulated between negative and positive controls.

`BoutrosRNAiLogFC`: matrix with log fold changes

References

Boutros M, Agaisse H, Perrimon N, Sequential activation of signaling pathways during innate immune responses in Drosophila. Developmental Cell. 3(5):711-722, 2002

See Also

`nem.discretize`

Examples

```
data("BoutrosRNAi2002")
dim(BoutrosRNAiExpression)
dim(BoutrosRNAiDiscrete)
```

---

closest.transitive.greedy

Find transitively closed graph most similar to the given one

Description

First, from the original graph Φ spurious edges are pruned via `prune.graph`. Then the new graph Φ' is transitively closed. Afterwards, the algorithms successively introduces new edges minimizing the distance to the original graph (defined as $\sum_{ij} |\Phi_{ij} - \Phi'_{ij}|$) most. After each edge addition the graph is transitively closed again.

Usage

```
closest.transitive.greedy(Phi, verbose=TRUE)
```

Arguments

Phi	adjacency matrix
verbose	do you want to see progress statements printed or not? Default: TRUE

Value

adjacency matrix

Author(s)

Holger Froehlich

See Also

prune.graph, transitive.closure, transitive.reduction

---

enumerate.models *Exhaustive enumeration of models*

Description

The function enumerate.models is used to create the model space for inference by exhaustive enumeration. It computes a list of all transitively closed directed graphs on a given number of nodes.

Usage

```
enumerate.models(x,name=NULL,trans.close=TRUE,verbose=TRUE)
```

Arguments

x	either the number of nodes or a vector of node names.
name	optionally the nodenames, if they are not provided in x
trans.close	should graphs be transitively closed?
verbose	if TRUE outputs number of (unique) models. Default: TRUE

Details

The model space of Nested Effects Models consists of all transitively closed directed graphs. The function enumerate.models creates them in three steps: (1.) build all directed graphs on x (or length(x)) nodes, (2.) transitively close each one of them, and (3.) remove redundant models to yield a unique set. So far, enumeration is limited to up to 5 nodes.

I'm aware that this is inefficient! It would be very desirable to enumerate the models directly (i.e. without creating all directed graphs as an intermediate step).

Value

a list of models. Each entry is a transitively closed adjacency matrix with unit main diagonal.

Author(s)

Florian Markowetz

See Also

nem

Examples

```
enumerate.models(2)
enumerate.models(c("Anna","Bert"))
```

---

generateNetwork *Random networks and data sampling*

Description

1. Random network generation; 2. sampling of data from a given network topology

Usage

```
sampleRndNetwork(Sgenes, scaleFree=TRUE, gamma=2.5, maxOutDegree=length(Sgenes), maxInDegree=length(S

sampleData(Phi, m, prob=NULL, uninformative=0, type="binary", replicates=4, typeI.err=0.05, typeII.err
```

Arguments

Sgenes	character vector of S-genes
scaleFree	should the network topology be scale free?
gamma	for scale free networks: out-degrees of nodes are sampled from $\frac{1}{Z} * (0 : maxOutDegree)^{-\gamma}$, where Z is a normalization factor
maxOutDegree	maximal out-degree of nodes
maxInDegree	maximal in-degree of nodes prior to transitive closure
trans.close	Should the transitive closure of the graph be returned? Default: TRUE
DAG	Should only DAGs be sampled? Default: FALSE
Phi	adjacency matrix
m	number of E-genes to sample
prob	probability for each S-gene to get an E-gene attached
uninformative	additional number of uninformative E-genes, i.e. E-genes carrying no information about the nested structure

type	"binary" = binary data; "density" = log 'p-value' densities sampled from beta-uniform mixture model; "lodds" = log odds sampled from two normal distributions
replicates	number of replicate measurements to simulate for binary data
typeI.err	simulated type I error for binary data
typeII.err	simulated type II error for binary data
alpha	parameter for $Beta(\alpha, 1)$ distribution: one parameter per S-gene
beta	parameter for $Beta(1, \beta)$ distribution: one parameter per S-gene
lambda	mixing coefficients for beta-uniform mixture model of the form: $\lambda_1 + \lambda_2 * Beta(\alpha, 1) + \lambda_3 * Beta(1, \beta)$. There is a vector of 3 mixing coefficients per model and one model per S-gene.
meansH1	normal distribution means of log odds ratios under the hypothesis of expecting an effect: one mean per S-gene
meansH0	normal distribution means of log odds ratios under the null hypothesis: one mean per S-gene
sdsH1	normal distribution standard deviations of log odds values under the hypothesis of expecting an effect: one sd per S-gene
sdsH0	normal distribution standard deviations of log odds values under the null hypothesis: one sd per S-gene

Details

Random networks are generated as follows: For each S-gene S_k we randomly choose the number o of outgoing edges between 0 and maxOutDegree. This is either done uniform randomly or, if scale free networks are created, according to a power law distribution specified by gamma. We then select o S-genes having at most maxInDegree ingoing edge and connected S_k to them.

The function `sampleData` samples data from a given network topology as follows: We first attach E-genes to S-genes according to the probabilities prob (default: uniform). We then simulate knockdowns of the individual S-genes. For those E-genes, where no effects are expected, values are sampled from a null distribution, otherwise from an alternative distribution. In the simplest case we only sample binary data, where 1 indicates an effect an 0 no effect. Alternatively, we can sample log "p-value" densities according to a beta-uniform mixture model, where the null distribution is uniform and the alternative a mixture of two beta distributions. A third possibility is to sample log odds ratios, where alternative and null distribution are both normal.

Value

For sampleRndNetwork an adjacency matrix, for sampleData a data matrix, for sampleData.BN a data matrix and a linking of effects to signals.

Author(s)

Holger Froehlich, Cordula Zeller

See Also

`getDensityMatrix`

Examples

```
Phi = sampleRndNetwork(paste("S",1:5,sep=""))
D = sampleData(Phi, 100, type="density")$D
plot(as(transitive.reduction(Phi),"graphNEL"), main="original graph")
x11()
plot.nem(nem(D, control=set.default.parameters(unique(colnames(D)), type="CONTmLLBayes")), transitiveReduction=
```

`getDensityMatrix` *Calculate density matrix from raw p-value matrix*

Description

Fit a 3 component BUM model to each column of a raw p-value matrix.

Usage

```
getDensityMatrix(Porig, dirname=NULL, startab=c(0.3,10), startlam=c(0.6,0.1,0.3), tol=1e-4)
```

Arguments

`Porig`	matrix of raw p-values
`dirname`	name of a directory to save histograms and QQ-plots to. If dirname=NULL, then the plots are made to the screen, and after each fit the user is asked to press a key in order to continue.
`startab`	start values for alpha and beta parameter
`startlam`	start values for mixing coefficients
`tol`	convergence tolerance: If the absolute likelihood ratio -1 becomes smaller than this value, then the EM algorithm is supposed to be converged.

Details

The BUM density model consists of 3 components: $f(x) = lambda_1 + lambda_2 * dbeta(x, alpha, 1) + lambda_3 * dbeta(x, 1, beta)$. The mixing coefficients and the parameters alpha and beta are fitted together via an EM algorithm.

Value

log-density matrix of same dimensions as Porig: The log-densities can be interpreted as log signal-to-noise ratios. A value > 0 means higher signal than noise, and a value < 0 a higher noise than signal.

Note

Note the difference to the previous package version: the LOG-density is returned now!

Author(s)

Holger Froehlich

getRelevantEGenes *Automatic selection of most relevant effect reporters*

Description

1. A-priori filtering of effect reporters/E-genes: Select effect reporters, which show a pattern of differential expression across experiments that is expected to be non-random. 2. Automated effect reporters subset selection: Select those effect reporters, which have the highest likelihood under the given network hypothesis.

Usage

```
filterEGenes(Porig, D, Padj=NULL, ntop=100, fpr=0.05, adjmethod="bonferroni", cutoff=0.05)

getRelevantEGenes(Phi, D, control, nEgenes=min(10*nrow(Phi), nrow(D)))
```

Arguments

	For method filterEGenes:
	matrix of raw p-values, typically from the complete array
Porig	data matrix. Columns correspond to the nodes in the silencing scheme. Rows are effect reporters.
Padj	matrix of false positive rates. If not, provided Benjamini-Hochbergs method for false positive rate computation is used.
ntop	number of top genes to consider from each knock-down experiment
fpr	significance cutoff for the FDR
adjmethod	adjustment method for pattern p-values
cutoff	significance cutoff for patterns
	For method getRelevantEGenes:
Phi	adjacency matrix with unit main diagonal
control	list of parameters: see `set.default.parameters`
nEgenes	no. of E-genes to select

Details

The method filterEGenes performs an a-priori filtering of the complete microarray. It determines how often E-genes are expected to be differentially expressed across experiments just randomly. According to this only E-genes are chosen, which show a pattern of differential expression more often than can be expected by chance.

The method getRelevantEGenes looks for the E-genes, which have the highest likelihood under the given network hypothesis. In case of the scoring type "CONTmLLBayes" these are all E-genes which have a positive contribution to the total log-likelihood. In case of type "CONTmLLMAP" all E-genes not assigned to the "null" S-gene are returned. This involves the prior probability delta/no. S-genes for leaving out an E-gene. For all other cases ("CONTmLL", "FULLmLL", "mLL") the nEgenes E-genes with the highest likelihood under the given network hypothesis are returned.

Value

I	index of selected E-genes
dat	subset of original data according to I
patterns	significant patterns
nobserved	no. of cases per observed pattern
selected	selected E-genes
mLL	marginal likelihood of a phenotypic hierarchy
pos	posterior distribution of effect positions in the hierarchy
mappos	Maximum a posteriori estimate of effect positions
LLperGene	likelihood per selected E-gene

Author(s)

Holger Froehlich

See Also

nem, score, mLL, FULLmLL

Examples

```
# Drosophila RNAi and Microarray Data from Boutros et al, 2002
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]

# enumerate all possible models for 4 genes
Sgenes = unique(colnames(D))
models <- enumerate.models(Sgenes)

getRelevantEGenes(models[[64]], D, control=set.default.parameters(Sgenes, para=c(.13,.05), type="mLL"))
```

---

infer.edge.type *Infer regulation direction for each edge*

Description

The method infers edge types (up-regulation, down-regulation) for a given nem model. Direct approach: For an edge a->b the method looks, whether b is up- or down-regulated in a knock-down of a.

Indirect approach: For an edge a->b the method looks at the fraction of E-genes attached to b (including b itself), which are up- or down-regulated in a knock-down of a. If significantly more genes are down-regulated than up-regulated, the edge a->b is assumed to be an activation. Likewise, if significantly more genes are up-regulated than down-regulated, a->b is assumed to be an inhibition. If there is no significant difference in up- and down-regulated edges, a->b does not have a specified type.

Usage

```
infer.edge.type(x, logFC, alpha=0.05, adj.method="BY", method=c("direct", "indirect"))
```

Arguments

x	nem object
logFC	matrix with fold changes. The rownames of this matrix should correspond to the rownames of the data matrix, which was used to infer the nem model.
alpha	p-value cutoff
adj.method	multiple testing correction method. Default: Benjamini-Yekutieli
method	default: "direct"

Details

Significance in case of the indirect method is calculated using a two-tailed binomial test with null hypothesis p=0.5.

Value

Modified nem object. Each edge in the nem graph now has a "weight" and a "label" attribute. The label attribute corresponds to the original value in the adjacency matrix. The weight attribute encodes up- and down-regulation in the following way: value 2 means up-regulation, value -1 down-regulation and value 1 an unknown effect.

Author(s)

Holger Froehlich

See Also

binom.test

Examples

```
    data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
result = nem(D, control=set.default.parameters(unique(colnames(D)), para=c(0.13,0.05)))
    resEdgeInf = infer.edge.type(result, BoutrosRNAiLogFC)
    plot.nem(resEdgeInf)
```

`internal` *internal functions*

Description

internal functions: do not call these functions directly.

Usage

```
various
```

Arguments

various

Value

various

Author(s)

Holger Froehlich

`Ivanova2006RNAiTimeSeries` *Perturbation Time Series*

Description

The dataset consists of time series microarray measurements (only 1 replicate, 6 time points) for RNAi knock-downs of each of the six genes Nanog, Oct4, Sox2, Esrrb, Tbx3, and Tcl1 in mouse embryonic stem cells.

Usage

```
data(Ivanova2006RNAiTimeSeries)
```

Format

dat: 8 x 122 x 6 array of discretized time series data for E-genes showing a abs(logFC) > 1 in at least one time point (see Anchang et al., 2009).

Details

Mouse embryonic stem cells (ESCs) were grown in the presence of leukemia inhibitory factor (LIF), thus retaining their undifferentiated self-renewing state (positive controls). Differentiation associated changes in gene expression were measured by replacing LIF with retinoic acid (RA), thus inducing differentiation of stem cells (negative controls). RNAi silencing of the 6 afore mentioned genes was done in (LIF+, RA-) cell culturs to investigate their potential for induced cell differentiation. Microarray expression measurements at 6 - 7 time points in one-day intervals were taken for the two controls (positive and negative) and the six RNAi assays. The dataset by Ivanova et al. (2006) was measured once on Affymetrix MOE430A and once on MOE430B chips. The authors normalized both datasets via the MAS 5.0 software (Hubbell et al., 2002). Here, only the Affymetrix MOE430A chip series is included.

References

Ivanova, N., Dobrin, R., Lu, R., Kotenko, I., Levorse, J., DeCoste, C., Schafer, X., Lun, Y., and Lemischka, I. R. (2006). Dissecting self-renewal in stem cells with RNA interference. Nature, 442(7102):533-538.\

Anchang, B., Sadeh, M. J., Jacob, J., Tresch, A., Vlad, M. O., Oefner, P. J., and Spang, R. (2009). Modeling the temporal interplay of molecular signaling and gene expression by using dynamic nested effects models. Proceedings of the National Academy of Sciences of the United States of America, 106(16):6447-52.

See Also

`Ivanova2006RNAiTimeSeries`

Examples

```
data("Ivanova2006RNAiTimeSeries")
dim(dat)
```

`local.model.prior` *Computes a prior to be used for edge-wise model inference*

Description

The function `pairwise.posterior` infers a phenotypic hierarchy edge by edge by choosing between four models (unconnected, subset, superset, undistinguishable). For each edge, `local.model.prior` computes a prior distribution over the four models. It can be used to ensure sparsity of the graph and high confidence in results.

Usage

```
local.model.prior(size,n,bias)
```

Arguments

size	expected number of edges in the graph.
n	number of perturbed genes in the dataset, number of nodes in the graph
bias	the factor by which the double-headed edge is preferred over the single-headed edges

Details

A graph on n nodes has N=n*(n-1)/2 possible directed edges (one- or bi-directional). If each edge occurs with probability p, we expect to see Np edges in the graph. The function local.model.prior takes the number of genes (n) and the expected number of edges (size) as an input and computes a prior distribution for edge occurrence: no edge with probability size/N, and the probability for edge existence being split over the three edge models with a bias towards the conservative double-headed model specified by bias. To ensure sparsity, the size should be chosen small compared to the number of possible edges.

Value

a distribution over four states: a vector of four positive real numbers summing to one

Author(s)

Florian Markowetz

See Also

pairwise.posterior, nem

Examples

```
# uniform over the 3 edge models
local.model.prior(4,4,1)
# bias towards <->
local.model.prior(4,4,2)
```

---

nem *Nested Effects Models - main function*

Description

The main function to perform model learning from data

Usage

```
nem(D,inference="nem.greedy",models=NULL,control=set.default.parameters(setdiff(unique(colnames(D))

## S3 method for class nem
print(x, ...)
```

Arguments

D	data matrix with experiments in the columns (binary or continious)
inference	search to use exhaustive enumeration, triples for triple-based inference, pairwise for the pairwise heuristic, ModuleNetwork for the module based inference, nem.greedy for greedy hillclimbing, nem.greedyMAP for alternating MAP optimization using log odds or log p-value densities, mc.eminem for EM based inference using log odds or log p-value densities, BN.greedy, BN.exhaustive for a conventional Bayesian Network treatment using binomial or normal distribution assumptions, dynoNEM for MCMC based inference from time series data, mc.eminem for EM based inference
models	a list of adjacency matrices for model search. If NULL, an exhaustive enumeration of all possible models is performed.
control	list of parameters: see set.default.parameters
verbose	do you want to see progression statements? Default: TRUE
x	nem object
...	other arguments to pass

Details

If parameter Pm != NULL and parameter lambda == 0, a Bayesian approach to include prior knowledge is used. Alternatively, the regularization parameter lambda can be tuned in a model selection step via the function nemModelSelection using the BIC criterion. If automated subset selection of effect reporters is used (default), the regularization parameter delta can be tuned via the BIC model selection criterion. Per default it is fixed to 1 / (no. S-genes + 1).

The function plot.nem plots the inferred phenotypic hierarchy as a directed graph, the likelihood distribution of the models (only for exhaustive search) or the posterior position of the effected genes.

Value

graph	inferred directed S-gene graph (graphNEL object)
mLL	log posterior marginal likelihood of model(s)
pos	posterior over effect positions
mappos	MAP estimate of effect positions
selected	selected E-gene subset
LLperGene	likelihood per selected E-gene
avg	in case of MCMC: posterior mean S-gene graph (edge weighted adjacency matrix)
control	hyperparameter as in function call

For inference = "mc.eminem" the following additional values are returned:

local.maxima	local maxima of the EM procedure
graphs.sampled	sampled graphs
EB	samples of the empirical Bayes prior
acc_list	list that indicates whether the corresponding sampled S-gene graph has been accepted (new local maximum (1), same local maximum (0)) or rejected(-1) in the MCMC sampling process - length(acc_list)=mcmc.nsamples + mcmc.nburnin

Author(s)

Holger Froehlich, Florian Markowetz

References

Markowetz, F.; Bloch, J. & Spang, R., Non-transcriptional Pathway Features Reconstructed from Secondary Effects of RNA interference. Bioinformatics, 2005, 21, 4026 - 4032\

Markowetz, F.; Kostka, D.; Troyanskaya, O. & Spang, R., Nested Effects Models for High-dimensional Phenotyping Screens. Bioinformatics, 2007, 23, i305 - i312\

Fr\"ohlich, H.; Fellmann, M.; S\"ultmann, H.; Poustka, A. & Beissbarth, T. Large Scale Statistical Inference of Signaling Pathways from RNAi and Microarray Data. BMC Bioinformatics, 2007, 8, 386\

Fr\"ohlich, H.; Fellmann, M.; S\"ultmann, H.; Poustka, A. & Beissbarth, T. Estimating Large Scale Signaling Networks through Nested Effect Models with Intervention Effects from Microarray Data. Bioinformatics, 2008, 24, 2650-2656\

Tresch, A. & Markowetz, F., Structure Learning in Nested Effects Models Statistical Applications in Genetics and Molecular Biology, 2008, 7\

Zeller, C.; Fr\"ohlich, H. & Tresch, A., A Bayesian Network View on Nested Effects Models EURASIP Journal on Bioinformatics and Systems Biology, 2009, 195272\

Fr\"ohlich, H.; Tresch, A. & Beissbarth, T., Nested Effects Models for Learning Signaling Networks from Perturbation Data. Biometrical Journal, 2009, 2, 304 - 323\

Fr\"ohlich, H.; Sahin, \"O.; Arlt, D.; Bender, C. & Beissbarth, T. Deterministic Effects Propagation Networks for Reconstructing Protein Signaling Networks from Multiple Interventions. BMC Bioinformatics, 2009, 10, 322\

Fr\"ohlich, H.; Praveen, P. & Tresch, A., Fast and Efficient Dynamic Nested Effects Models. Bioinformatics, 2011, 27, 238-244\

Niederberger, T.; Etzold, S.; Lidschreiber, M; Maier, K.; Martin, D.; Fr\"ohlich, H.; Cramer, P.; Tresch, A., MC Eminem Maps the Interaction Landscape of the Mediator, PLoS Comp. Biol., 8(6): e1002568, 2012.

See Also

set.default.parameters, nemModelSelection, nem.jackknife, nem.bootstrap, nem.consensus, local.model.prior, plot.nem

Examples

```
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
control = set.default.parameters(unique(colnames(D)), para=c(0.13, 0.05))
res1 <- nem(D,inference="search", control=control)
res2 <- nem(D,inference="pairwise", control=control)
res3 <- nem(D,inference="triples", control=control)
res4 <- nem(D,inference="ModuleNetwork", control=control)
res5 <- nem(D,inference="nem.greedy", control=control)
res6 = nem(BoutrosRNAiLods, inference="nem.greedyMAP", control=control)
```

```
par(mfrow=c(2,3))
plot.nem(res1,main="exhaustive search")
plot.nem(res2,main="pairs")
plot.nem(res3,main="triples")
plot.nem(res4,main="module network")
plot.nem(res5,main="greedy hillclimber")
plot.nem(res6,main="alternating MAP optimization")
```

`nem.bootstrap` *Bootstrapping for nested effect models*

Description

Performs bootstrapping (resampling with replacement) on effect reporters to assess the statistical stability of networks

Usage

```
nem.bootstrap(D, thresh=0.5, nboot=1000,inference="nem.greedy",models=NULL,control=set.default.param

## S3 method for class nem.bootstrap
print(x, ...)
```

Arguments

`D`	data matrix with experiments in the columns (binary or continous)
`thresh`	only edges appearing with a higher frequency than "thresh" are returned
`nboot`	number of bootstrap samples desired
`inference`	`search` to use exhaustive enumeration, `triples` for triple-based inference, `pairwise` for the pairwise heuristic, `ModuleNetwork` for the module based inference, `nem.greedy` for greedy hillclimbing, `nem.greedyMAP` for alternating MAP optimization using log odds or log p-value densities
`models`	a list of adjacency matrices for model search. If NULL, an exhaustive enumeration of all possible models is performed.
`control`	list of parameters: see `set.default.parameters`
`verbose`	do you want to see progression statements? Default: TRUE
`x`	nem object
`...`	other arguments to pass

Details

Calls `nem` or `nemModelSelection` internally, depending on whether or not lambda is a vector and Pm != NULL. For DEPNs a stratified bootstrap is carried out, where strate are defined on each replicate group for each time point.

Value

nem object with edge weights being the bootstrap probabilities

Author(s)

Holger Froehlich

See Also

nem.jackknife, nem.consensus, nem.calcSignificance, nem

Examples

```
## Not run:
   data("BoutrosRNAi2002")
   D <- BoutrosRNAiDiscrete[,9:16]
  nem.bootstrap(D, control=set.default.parameters(unique(colnames(D)), para=c(0.13,0.05)))

## End(Not run)
```

---

nem.calcSignificance *Statistical significance of network hypotheses*

Description

Assess statistical significance of a network hypothesis by comparing it to a null hypothesis.

Usage

```
nem.calcSignificance(D, x, N=1000, which.test=c("permutation", "rand.net", "rand.modify"), seed=1, mc.
```

Arguments

D	data matrix with experiments in the columns (binary or continious)
x	nem object
N	number of random networks to sample
which.test	type of significance test to conduct (see details)
seed	random seed
mc.cores	number of cores to be used on a multicore processor

Details

Given data, N random network hypotheses from a null distribution are drawn as follows: For each S-gene S_k we randomly choose a number o of outgoing edges between 0 and 3. We then select o S-genes having at most 1 ingoing edge, connected S_k to them and transitively closed the graph. For all random network hypotheses it is counted, how often their likelihood is bigger than that of the given network. This yields an exact p-value.

Another way of assessing the statistical significance of the network hypothesis is to draw random permutations of the node labels. Note that in this case the node degree distribution is the same as in the given network. Again, we can obtain an exact p-value by counting, how often the likelihood of the permuted network is bigger than that of the given network.

Finally, comparison to randomly perturbed networks (insertion, deletion or reversion of 1 edge) yields an exact p-value describing the stability of the network. For dynoNEMs network modification operations are edge weight increase, decrease and swap.

Value

For "random" network: p-value of the network according to the null hypothesis that it is random. For permuted nework: p-value of the network according to the null hypothesis that a network with permuted node labels is at least as good. For perturbed / modified network: p-value of the network according to the null hypothesis a randomly peturbed network is at least as good.

Author(s)

Holger Froehlich

See Also

nem.consensus, nem.jackknife, nem.bootstrap, nem

Examples

```
## Not run:
   data("BoutrosRNAi2002")
   D <- BoutrosRNAiDiscrete[,9:16]
  res = nem(D, control=set.default.parameters(unique(colnames(D)), para=c(0.13,0.05))) # get best network
   nem.calcSignificance(D,res) # assess its statistical significance

## End(Not run)
```

---

nem.consensus *Statistically stabile nested effects models*

Description

Performs bootstrapping (resampling with replacement) on E-genes and jackknife on S-genes to assess the statistical stability of networks. Only edges appearing with a higher frequency than a predescribed threshold in both procedures are regarded as statistical stable and appear in the so-called consensus network.

Usage

```
nem.consensus(D,thresh=0.5, nboot=1000,inference="nem.greedy",models=NULL,control=set.default.parame

## S3 method for class nem.consensus
print(x, ...)
```

Arguments

D	data matrix with experiments in the columns (binary or continous)
thresh	only edges appearing with a higher frequency than "thresh" in both, bootstrap and jackknife procedure, are regarded as statistically stable and trust worthy
nboot	number of bootstrap samples desired
inference	search to use exhaustive enumeration, triples for triple-based inference, pairwise for the pairwise heuristic, ModuleNetwork for the module based inference, nem.greedy for greedy hillclimbing, nem.greedyMAP for alternating MAP optimization using log odds or log p-value densities
models	a list of adjacency matrices for model search. If NULL, an exhaustive enumeration of all possible models is performed.
control	list of parameters: see set.default.parameters
verbose	do you want to see progression statements? Default: TRUE
x	nem object
...	other arguments to pass

Details

Calls nem or nemModelSelection internally, depending on whether or not lambda is a vector and Pm != NULL.

Value

consensus network (nem object)

Author(s)

Holger Froehlich

See Also

nem.bootstrap, nem.jackknife, nem.calcSignificance, nem

Examples

```
## Not run:
   data("BoutrosRNAi2002")
   D <- BoutrosRNAiDiscrete[,9:16]
  nem.consensus(D, control=set.default.parameters(unique(colnames(D)), para=c(0.13,0.05)))

## End(Not run)
```

`nem.cont.preprocess` *Calculate classification probabilities of perturbation data according to control experiments*

Description

Calculates probabilities of data to define effects of interventions with respect to wildtype/control measurements

Usage

```
nem.cont.preprocess(D,neg.control=NULL,pos.control=NULL,nfold=2, influencefactor=NULL, empPval=.05,
```

Arguments

`D`	matrix with experiments as columns and effect reporters as rows
`neg.control`	either indices of columns in `D` or a matrix with the same number of rows as `D`
`pos.control`	either indices of columns in `D` or a matrix with the same number of rows as `D`
`nfold`	fold-change between neg. and pos. controls for selecting effect reporters. Default: 2
`influencefactor`	factor multiplied onto the probabilities, so that all negative control genes are treated as influenced, usually automatically determined
`empPval`	empirical p-value cutoff for effects if only one control is available
`verbose`	Default: TRUE

Details

Determines the empirical distributions of the controls and calculates the probabilities of pertubartion data to belong to the control distribution(s).

Value

`dat`	data matrix
`pos`	positive controls [in the two-controls setting]
`neg`	negative controls [in the two-controls setting]
`sel`	effect reporters selected [in the two-controls setting]
`prob.influenced`	probability of a reporter to be influenced
`influencefactor`	factor multiplied onto the probabilities, so that all negative control genes are treated as influenced

Note

preliminary! will be developed to be more generally applicable

Author(s)

Florian Markowetz

References

Markowetz F, Bloch J, Spang R, Non-transcriptional pathway features reconstructed from secondary effects of RNA interference, Bioinformatics, 2005

See Also

BoutrosRNAi2002

Examples

```
 data("BoutrosRNAi2002")
preprocessed <- nem.cont.preprocess(BoutrosRNAiExpression,neg.control=1:4,pos.control=5:8)
```

---

nem.discretize *Discretize perturbation data according to control experiments*

Description

discretizes raw data to define effects of interventions with respect to wildtype/control measurements

Usage

```
nem.discretize(D,neg.control=NULL,pos.control=NULL,nfold=2,cutoff=0:10/10, pCounts=20, empPval=.05,
```

Arguments

D	matrix with experiments as columns and effect reporters as rows
neg.control	either indices of columns in D or a matrix with the same number of rows as D
pos.control	either indices of columns in D or a matrix with the same number of rows as D
nfold	fold-change between neg. and pos. controls for selecting effect reporters. Default: 2
cutoff	a (vector of) cutoff value(s) weighting the pos. controls versus the neg. controls. Default: 0:10/10
pCounts	pseudo-counts to guard against unreasonable low error estimates
empPval	empirical p-value cutoff for effects if only one control is available
verbose	Default: TRUE

Details

Chooses cutoff such that separation between negative and positive controls becomes optimal.

Value

`dat`	discretized data matrix
`pos`	discretized positive controls [in the two-controls setting]
`neg`	discretized negative controls [in the two-controls setting]
`sel`	effect reporters selected [in the two-controls setting]
`cutoff`	error rates for different cutoff values [in the two-controls setting]
`para`	estimated error rates [in the two-controls setting]

Note

preliminary! will be developed to be more generally applicable

Author(s)

Florian Markowetz

References

Markowetz F, Bloch J, Spang R, Non-transcriptional pathway features reconstructed from secondary effects of RNA interference, Bioinformatics, 2005

See Also

BoutrosRNAi2002

Examples

```
 # discretize Boutros data as in
 # Markowetz et al, 2005
 data("BoutrosRNAi2002")
disc <- nem.discretize(BoutrosRNAiExpression,neg.control=1:4,pos.control=5:8,cutoff=.7)
 stopifnot(disc$dat==BoutrosRNAiDiscrete[,9:16])
```

`nem.jackknife` *Jackknife for nested effect models*

Description

Assesses the statistical stability of a network via a jackknife procedure: Each S-gene is left out once and the network reconstructed on the remaining ones. The relative frequency of each edge to appear in n-1 jackknife samples is returned.

Usage

```
nem.jackknife(D, thresh=0.5, inference="nem.greedy",models=NULL,control=set.default.parameters(uniqu

## S3 method for class nem.jackknife
print(x, ...)
```

Arguments

`D`	data matrix with experiments in the columns (binary or continious)
`thresh`	only edges appearing with a higher frequency than "thresh" are returned
`inference`	`search` to use exhaustive enumeration, `triples` for triple-based inference, `pairwise` for the pairwise heuristic, `ModuleNetwork` for the module based inference, `nem.greedy` for greedy hillclimbing, `nem.greedyMAP` for alternating MAP optimization using log odds or log p-value densities
`models`	a list of adjacency matrices for model search. If NULL, an exhaustive enumeration of all possible models is performed.
`control`	list of parameters: see `set.default.parameters`
`verbose`	do you want to see progression statements? Default: TRUE
`x`	nem object
`...`	other arguments to pass

Details

Calls `nem` or `nemModelSelection` internally, depending on whether or not parameter lambda is a vector and parameter Pm != NULL.

Value

nem object with edge weights being the jackknife probabilities

Author(s)

Holger Froehlich

See Also

nem.bootstrap, nem.consensus, nem, nemModelSelection

Examples

```
## Not run:
   data("BoutrosRNAi2002")
   D <- BoutrosRNAiDiscrete[,9:16]
  nem.jackknife(D, control=set.default.parameters(unique(colnames(D)), para=c(0.13,0.05)))

## End(Not run)
```

---

nemModelSelection *Model selection for nested effect models*

Description

Infers models with different regularization constants, compares them via the BIC or AIC criterion and returns the highest scoring one

Usage

```
nemModelSelection(lambdas,D,inference="nem.greedy",models=NULL,control=set.default.parameters(uniqu
```

Arguments

lambdas	vector of regularization constants
D	data matrix with experiments in the columns (binary or continious)
inference	search to use exhaustive enumeration, triples for triple-based inference, pairwise for the pairwise heuristic, ModuleNetwork for the module based inference, nem.greedy for greedy hillclimbing, nem.greedyMAP for alternating MAP optimization using log odds or log p-value densities
models	a list of adjacency matrices for model search. If NULL, an exhaustive enumeration of all possible models is performed.
control	list of parameters: see set.default.parameters
verbose	do you want to see progression statements? Default: TRUE
...	other arguments to pass to function nem or network.AIC

Details

nemModelSelection internally calls nem to infer a model with a given regularization constant. The comparison between models is based on the BIC or AIC criterion, depending on the parameters passed to network.AIC.

Value

nem object

Author(s)

Holger Froehlich

See Also

set.default.parameters, nem, network.AIC

Examples

```
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
hyper = set.default.parameters(unique(colnames(D)), para=c(0.13, 0.05), Pm=diag(4))
res <- nemModelSelection(c(0.1,1,10), D, control=hyper)

plot.nem(res,main="highest scoring model")
```

---

network.AIC *AIC/BIC criterion for network graph*

Description

Calclate AIC/BIC for a given network graph (should be transitively closed). The number of free parameters equals the number of unknown edges in the network graph.

Usage

```
network.AIC(network,Pm=NULL,k=length(nodes(network$graph)),verbose=TRUE)
```

Arguments

network	a nem object (e.g. 'pairwise')
Pm	prior over models (n x n matrix). If NULL, then a matrix of 0s is assumed
k	penalty per parameter in the AIC/BIC calculation. k = 2 for classical AIC
verbose	print out the result

Details

For k = log(n) the BIC (Schwarz criterion) is computed. Usually this function is not called directly but from nemModelSelection

Value

AIC/BIC value

Author(s)

Holger Froehlich

See Also

nemModelSelection

Examples

```
data("BoutrosRNAi2002")
D = BoutrosRNAiDiscrete[,9:16]
control = set.default.parameters(unique(colnames(D)), para=c(0.13,0.05))
res1 <- nem(D, control=control)
network.AIC(res1)
control$lambda=100 # enforce sparsity
res2 <- nem(D,control=control)
network.AIC(res2)
```

NiederbergerMediator2012

Expression measurements upon perturbation of Mediator subunits

---

Description

This dataset combines expression data from 4 gene perturbation studies in S. cerevisae, including the perturbations of the 9 Mediator subunits Med2, Med7 (N-terminus), Med7 (C-terminus), Med10, Med15, Med19, Med20, Med21 and Med31. Med10 and Med21 have been combined due to indistinguishable results.

Usage

```
data(NiederbergerMediator2012)
```

Format

NiederbergerMediatorLods: data matrix(log-odds ratios): 2578 x 8 NiederbergerMediatorLogFC: data matrix(log2 fold-changes): 2578 x 8 NiederbergerMediatorPVals: data matrix(p-values): 2578 x 8

Details

The data has been preprocessed using the R/Bioconductor package limma. The results have been filtered as follows: The original data contains a double-knockout Med7N/Med31, which can't be assessed by EMiNEM and has thus been removed. Analysis of the data revealed genes showing batch-specific effects, they have been excluded from further analysis. Finally, genes that do not react to any perturbation have been removed since they do not contain additional information. For a more detailed description of the individual studies and datasets as well as further information on

data processing and results, please refer to Niederberger et al, 2012. The raw data is available on Array Express (accession number [provided upon publication]).

NiederbergerMediatorLods: the log-odds ratios NiederbergerMediatorLogFC: the log2 fold-changes NiederbergerMediatorPVals: the corresponding p-values

References

Niederberger T., Etzold S., Lidschreiber M., Maier K.C., Martin D.E., Fr\"o hlich H., Cramer P., Tresch A., MC EMiNEM Maps the Interaction Landscape of the Mediator. PLoS Computational Biology, 8(6): e1002568, 2012-

Examples

```
data(NiederbergerMediator2012)
dim(NiederbergerMediatorLods)
dim(NiederbergerMediatorLogFC)
dim(NiederbergerMediatorPVals)
```

---

plot.nem *plot nested effect model*

Description

plot graph of nested effects model, the marginal likelihood distribution or the posterior position of the effected genes

Usage

```
## S3 method for class nem
plot(x, what="graph", remove.singletons=FALSE, PDF=FALSE, filename="nemplot.pdf", thresh=0, transitiv
```

Arguments

x	nem object to plot
what	(i), "graph", (ii) "mLL" = likelihood distribution, (iii) "pos" = posterior position of effected genes
remove.singletons	remove unconnected nodes from the graph plot
PDF	output as PDF-file
filename	filename of PDF-file
thresh	if x has a real valued adjacency matrix (weight matrix), don't plot edges with |weight| <= thresh
transitiveReduction	plot a transitively reduced graph

plot.probs	plot edge weights/probabilities. If regulation directions have been inferred (see infer.edge.type), upregulated edges are drawn red and downregulated edges blue. Edges, were no clear direction could be inferred, are drawn in black.
SCC	plot the strongly connected components graph
D	Visualize the nested subset structure of the dataset via plotEffects along with the graph and show the linking of E-genes to S-genes in the dataset. Should only be used for small networks. Default: Just plot the graph
draw.lines	If the nested subset structure is shown, should additionally lines connecting S-genes and their associated E-genes be drawn? WARNING: For larger datasets than e.g. 5 S-genes this most probably does not work, because the nested subset structure picture then partially overlaps with the graph picture. Default: Do not draw these lines
palette	color palette to use: either 'BlueRed' (default) or 'Grey'
...	other arguments to be passed to the Rgraphviz plot function or to the graphics 'image' function.

Value

none

Author(s)

Florian Markowetz, Holger Froehlich

See Also

nem, plotEffects, infer.edge.type

---

plotEffects *Plots data according to a phenotypic hierarchy*

Description

plotEffects visualizes the subset structure in the data by reordering rows and columns according to the topological order given by a phenotypic hierarchy.

Usage

```
plotEffects(D,nem,border=TRUE,legend=TRUE,order=NULL,orderSCC=TRUE,palette="BlueRed",...)
```

Arguments

D	data matrix
nem	phenotypic hierarchy (object of class 'score' or 'pairwise')
border	draw red lines to indicate gene-specific effect reporters. Default: TRUE
legend	plot a legend. Default: TRUE
order	pre-define an order of the S-genes instead of the topological order to visualize the subset structure. Default: Use topological order.
orderSCC	Is the pre-defined order given on strongly connected components rather than on individual nodes?
palette	color palette to use: either 'BlueRed' (default) or 'Grey'
...	additional parameters for the graphics function 'image'

Details

The experiments in the columns are reordered according to the topological order given by a phenotypic hierarchy. The effect reporters in the rows are grouped together by their position in the hierarchy. The groups are then arranged by topological order. Within each group the rows are hierarchically clustered.

Value

ordering of the E-genes according to the hierarchy (vector of indices)

Note

This function was formerly named `plot.effects`. This naming is not possible any more, since S3 classes were used for the function `plot.nem`.

Author(s)

Florian Markowetz, Holger Froehlich

Examples

```
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
res <- nem(D,control=set.default.parameters(unique(colnames(D)), para=c(.13,.05)))
plotEffects(D,res)
```

prior.EgeneAttach.EB *Initialize E-gene attachment prior for empirical Bayes*

Description

This function computes an initial prior for the attachment of E-genes based on observed log-pvalue densities or log-odds ratios. This prior can then be updated within an empirical Bayes procedure (MC.EMiNEM).

Usage

```
prior.EgeneAttach.EB(ratioMat)
```

Arguments

ratioMat	data matrix with experiments in the columns (log-odds ratios or log-pvalue densities)

Value

|E-genes| x (|S-genes| + 1) matrix with prior E-gene attachment probabilities. The last column denotes the virtual 'null' S-gene, which is there to filter E-genes that have no obvious attachment to any of the real S-genes.

Author(s)

Theresa Niederberger, Holger Froehlich

References

Niederberger, T.; Etzold, S.; Lidschreiber, M; Maier, K.; Martin, D.; Fr\"ohlich, H.; Cramer, P.; Tresch, A., MC Eminem Maps the Interaction Landscape of the Mediator, PLoS Comp. Biol., 2012, submitted.

See Also

set.default.parameters, nem

Examples

```
# only for test purposes
   data("BoutrosRNAi2002")
   D <- BoutrosRNAiDens
  control = set.default.parameters(unique(colnames(D)), Pe=prior.EgeneAttach.EB(D), mcmc.nsamples=100, mcmc.nbur
   res <- nem(D,inference="mc.eminem", control=control)

   plot(res)
```

prune.graph *Prunes spurious edges in a phenotypic hierarchy*

Description

A heuristic to prune spurious edges in a pehnotypic hierarchy

Usage

```
prune.graph(g,cutIN=NULL,cutOUT=NULL,quant=.95,verbose=TRUE)
```

Arguments

g	an adjacency matrix or a 'graphNEL' object
cutIN	minimum number of missing in-edges required to cut all in-edges. Default
cutOUT	minimum number of missing out-edges required to cut all out-edges
quant	if 'cutIN' or 'cutOUT' are not assigned, a quantile 'quant' of the distribution of missing in- or out-edges for all nodes is used
verbose	Default: TRUE

Details

prune.graph provides a heuristic approach to prune surious edges. prune.graph compares the input graph to its transitive closure, and counts for each node how many incoming and outgoing edges are missing. If the number is bigger than a user-defined cutoff, all incoming (outgoing) edges are removed.

Value

graph	the pruned phenotypic hierarchy (a 'graphNEL' object)
removed	number of removed edges
missing.in	number of missing in-edges for each node
missing.out	number of missing out-edges for each node

Author(s)

Florian Markowetz

Examples

```
# a transitively closed core with two spurious edges
g <- matrix(0,5,5)
g[1,2] <- 1
g[2,c(3,4)] <- 1
g[3,4] <- 1
g[4,5] <- 1
```

```
dimnames(g) <- list(LETTERS[1:5],LETTERS[1:5])
g <- as(g,"graphNEL")

# prune graph
gP <- prune.graph(g)

# plot
par(mfrow=c(1,2))
plot(g,main="two spurious edges")
plot(gP$graph,main="pruned")
```

---

quicknem *Quick run of Nested Effects Models inference*

Description

Interface to learn NEM models from data

Usage

```
quicknem(D,type="CONTmLLDens",inference="nem.greedy",controls.name=NULL,contrasts=NULL,normalize=FA
```

Arguments

D	ExpressionSet object or data matrix with raw or normalized expression data.
type	Parameter estimation, either mLL, FULLmLL, CONTmLL, CONTmLLBayes, CONTmLLMAP, depn.
inference	search to use exhaustive enumeration, triples for triple-based inference, pairwise for the pairwise heuristic, ModuleNetwork for the module based inference, nem.greedy for greedy hillclimbing, nem.greedyMAP for alternating MAP optimization using log odds or log p-value densities
controls.name	Pattern to search for in the columnnames of D. Defines which columns in D should be regarded as controls.
contrasts	String defining the contrasts to estimate via limma
normalize	boolean value, should quantile normalization be performed
cutoff	P-value cutoff for differential expression using adjusted p.values from limma.
DIR	Directory name, where additional informative plots should be stored. Created if not present.
plot	Should the inferred network be plotted?
bootstrap	Integer defining the number of bootstrapping samples to be performed. Defaults to 0.
...	other arguments to pass

Details

Wrapper function for call of nem inference. Extracts differential genes for given contrasts and infers a NEM - graph for the given inference type.

D Is either an ExpressionSet Object or a matrix/data.frame containing the expression values from the siRNA knockdown experiments. If an ExpressionSet, the data is extracted via exprs(ExpressionSet). The knockdowns must be in the columns, the measured effect genes in the rows of the expression matrix.

type `mLL` or `FULLmLL` or `CONTmLL` or `CONTmLLBayes` or `CONTmLLMAP` or `depn`. `CONTmLLDens` and `CONTmLLRatio` are identical to `CONTmLLBayes` and `CONTmLLMAP` and are still supported for compatibility reasons. `mLL` and `FULLmLL` are used for binary data (see `BoutrosRNAiDiscrete`) and `CONTmLL` for a matrix of effect probabilities. `CONTmLLBayes` and `CONTmLLMAP` are used, if log-odds ratios, p-value densities or any other model specifies effect likelihoods. `CONTmLLBayes` refers to an inference scheme, were the linking positions of effect reporters to network nodes are integrated out, and `CONTmLLMAP` to an inference scheme, were a MAP estimate for the linking positions is calculated. `depn` indicates Deterministic Effects Propagation Networks (DEPNs).

inference Type of network reconstruction. `search` enumerates all possible networks. Set to `triples`, `pairwise`, `ModuleNetwork`, `nem.greedy` or `nem.greedyMAP` for heuristic search of the network.

controls.name Defines a pattern to search for in the column names of D, which describes the control experiments. Each remaining experiment is then compared via limma to these controls by defining the appropriate contrasts. If NULL, then `controls.name` must be given, except for using `type="depn"`, where neither `controls.name` nor `contrasts` needs to be defined.

contrasts Defines the contrasts of interest that should be used for the limma analysis. If NULL, then `controls.name` must be given, except for using `type="depn"`, where neither `controls.name` nor `contrasts` needs to be defined.

DIR In case of `type="CONTmLLDens"` or `type="CONTmLLBayes"` some additional plots for the BUM model fits are created and stored here.

Value

`graph`	the inferred directed graph (graphNEL object)
`mLL`	log posterior marginal likelihood of final model
`pos`	posterior over effect positions
`mappos`	MAP estimate of effect positions
`selected`	selected E-gene subset
`LLperGene`	likelihood per selected E-gene
`control`	hyperparameter as in function call
`bootstrap`	Integer number defining how many bootstrap samples should be drawn. If 0, no bootstrapping will be performed. Else, `nem.bootstrap` will be called internally.

Author(s)

Christian Bender, Holger Froehlich, Florian Markowetz

See Also

nem, set.default.parameters, nemModelSelection, nem.jackknife, nem.bootstrap, nem.consensus, local.model.prior, plot.nem

Examples

```
## Not run:
data(BoutrosRNAi2002)
exps <- colnames(BoutrosRNAiExpression)
res <- quicknem(BoutrosRNAiExpression,controls="control")
res <- quicknem(BoutrosRNAiExpression,controls="control",type="CONTmLLRatio")
res <- quicknem(BoutrosRNAiExpression,controls="control",type="CONTmLLRatio",inference="ModuleNetwork")
contrasts <- c("rel-control","rel-LPS","key-control","key-LPS","tak-control","tak-LPS","mkk4hep-control","mkk4h
res <- quicknem(BoutrosRNAiExpression,contrasts=contrasts)

data(SahinRNAi2008)
dat <- dat.unnormalized #[,sample(1:17,5)]
res <- quicknem(dat,type="depn")

## End(Not run)
```

SahinRNAi2008 *Combinatorial Protein Knockdowns in the ERBB Signaling Pathway*

Description

Sixteen RNAi knockdowns (including 3 double knockdowns) of proteins in the ERBB signaling pathway of trastuzumab resistant breast cancer cells were conducted. Reverse Phase Protein Array (RPPA) measurments for 10 signaling intermediates are available before and after EGF stimulation with 4 technical and 3 biological replicates.

Usage

```
data(SahinRNAi2008)
```

Format

dat.unnormalized: 408 x 17 matrix (rows = RPPA measurements for (16 KOs + MOCK) x 4 technical x 3 biological replicates, columns = 10 antibodies + 6 proteins without measurements + time) dat.normalized: 408 x 17 matrix (measurements from technical and biological replicates are quantile normalized for each RNAi experiment) map.int2node: list with names being interventions (=names of dat.normalized) and entries being node names (=column names of dat.normalized)

Details

The cells were lysed on ice by scraping the cells in M-PER lysis buffer (Pierce, Rockford, IL) containing protease inhibitor Complete Mini (Roche, Basel), anti-phosphatase PhosSTOP (Roche, Basel), 10 mM NaF and 1mM Na4VO3. Protein concentrations were determined with a BCA Protein Assay Reagent Kit (Pierce, Rockford, IL). Lysates were mixed 1:2 with 2 times Protein Arraying Buffer (Whatman, Brentfort, UK) to obtain a final protein concentration of 1.5 mug/muL. Briefly, these lysates were printed onto nitrocellulose coated ONCYTE-slides (Grace Bio Labs, Bend, USA) using a non-contact piezo spotter, sciFlexxarrayer S5 (Scienion, Berlin, Germany). After primary and near-infrared (NIR)-dye labeled secondary antibodies applied, spots were analysed using an Odyssey scanner (LI-COR, Lincoln, USA) and signal intensities were quantified using Odyssey 2.0 software (For detailed information and an antibody list, see Sahin et al., 2008). Since no antibody against MEK1 was available, we measured protein expression of pERK1/2, which is downstream of MEK1.

References

Oezguer Sahin, Holger Froehlich, Christian Loebke, Ulrike Korf, Sara Burmester, Meher Majety, Jens Mattern, Ingo Schupp, Claudine Chaouiya, Denis Thieffry, Annemarie Poustka, Stefan Wiemann, Tim Beissbarth, Dorit Arlt, Modeling ERBB receptor-regulated G1/S transition to find novel targets for de novo trastuzumab resistance, BMC Systems Biology, 2008

See Also

BoutrosRNAi2002

Examples

```
data("SahinRNAi2008")
dim(dat.normalized)
dim(dat.unnormalized)
```

SCCgraph *Combines Strongly Connected Components into single nodes*

Description

SCCgraph is used to identify all nodes which are not distinguishable given the data.

Usage

```
SCCgraph(x,name=TRUE,nlength=20)
```

Arguments

x	graphNEL object or an adjacency matrix
name	Concatenate all names of summarized nodes, if TRUE, or number nodes, if FALSE. Default: TRUE
nlength	maximum length of names

Details

A graph inferred by either nem or nemModelSelection may have cycles if some phenotypic profiles are not distinguishable. The function SCCgraph identifies cycles in the graph (the strongly conneced components) and summarizes them in a single node. The resulting graph is then acyclic.

Value

graph	a graphNEL object with connected components of the input graph summarized into single nodes
scc	a list mapping SCCs to nodes
which.scc	a vector mapping nodes to SCCs

Author(s)

Florian Markowetz, Holger Froehlich

See Also

nem, transitive.reduction

Examples

```
data("BoutrosRNAi2002")
D   <- BoutrosRNAiDiscrete[,9:16]
res <- nem(D,control=set.default.parameters(unique(colnames(D)), para=c(.13,.05)))
#
sccg <- SCCgraph(res$graph,name=TRUE)
#
par(mfrow=c(1,2))
plot.nem(res, main="inferred from data")
plot(sccg$graph, main="condensed (rel,key)")
```

---

set.default.parameters

Get/set hyperparameters

Description

Allows to set and retrieve various hyperparameters for different inference methods.

Usage

```
set.default.parameters(Sgenes, ...)
```

Arguments

Sgenes	character vector of S-gene identifiers
...	parameters to set (see details)

Details

Since version 2.5.4 functions in the nem package do not have any more a large amount of individual parameters. Instead there is just one hyperparameter, which is passed to all functions. Parameter values with the hyperparameter can be set with this function.

type `mLL` or `FULLmLL` or `CONTmLL` or `CONTmLLBayes` or `CONTmLLMAP` or `depn`. `CONTmLLDens` and `CONTmLLRatio` are identical to `CONTmLLBayes` and `CONTmLLMAP` and are still supported for compatibility reasons. `mLL` and `FULLmLL` are used for binary data (see `BoutrosRNAiDiscrete`) and `CONTmLL` for a matrix of effect probabilities. `CONTmLLBayes` and `CONTmLLMAP` are used, if log-odds ratios, p-value densities or any other model specifies effect likelihoods. `CONTmLLBayes` refers to an inference scheme, were the linking positions of effect reporters to network nodes are integrated out, and `CONTmLLMAP` to an inference scheme, were a MAP estimate for the linking positions is calculated. `depn` indicates Deterministic Effects Propagation Networks (DEPNs).

para vector of length two: false positive rate and false negative rate for binary data. Used by `mLL`

hyperpara vector of length four: used by `FULLmLL()` for binary data

Pe prior of effect reporter positions in the phenotypic hierarchy (same dimension as D). Not used type `depn`. Default: NULL

Pm prior over models (n x n matrix). Default: NULL

Pm.frac_edges expected fraction of edges in the true S-gene graph

Pmlocal local model prior for pairwise and triple learning. For pairwise learning generated by `local.model.prior` according to arguments `local.prior.size` and `local.prior.bias`

local.prior.size prior expected number of edges in the graph (for pairwise learning). Default: no. nodes

local.prior.bias bias towards double-headed edges. Default: 1 (no bias; for pairwise learning)

triples.thrsh threshold for model averaging to combine triple models for each edge. Default: 0.5

lambda regularization parameter to incorporate prior assumptions. May also be a vector of possible values, if `nemModelSelection` is used, Default: 0 (no regularization)

delta regularization parameter for automated subset selection of effect reporters. If no E-gene selection is wanted, set delta to 0. Default: 1/ (no. S-genes + 1)

selEGenes.method If "regularization", E-gene selection is performed by introducing a "null" S-gene to which E-genes are attached with probability delta/ (no. S-genes + 1). If "iterative" and selEGenes=TRUE, `getRelevantEGenes` is called and a new model is trained on the selected E-genes. The process is then repeated until convergence. Default: "regularization"

selEGenes Tune parameter delta for automated selection of E-genes. Default: FALSE. NOTE: Since version > 2.18.0 E-gene selection is now performed per default by using the regularization mechanism with parameter delta. If no E-gene selection is wanted, set delta to 0.

trans.close Should always transitive closed graphs be computed? Default: TRUE. NOTE: This has only an impact for type `nem.greedyMAP` and `depn`. Default: TRUE

backward.elimination For module networks and greedy hillclimbing inference: Try to eliminate edges increasing the likelihood. Works only, if trans.close=FALSE. Default: FALSE

mode For Bayesian network inference and DEPNs: `binary_ML`: effects come from a binomial distribution - ML learning of parameters (Bayesian networks only); `binary_Bayesian`: effects come from a binomial distribution - Bayesian learning of parameters (Bayesian networks

only); `continuous_ML`: effects come from a normal distribution - ML learning of parameters; `continuous_Bayesian`: effects come from a normal distribution - Bayesian learning of parameters.

nu.intervention, lambda.intervention For `depn`: For any perturbed node we suppose the unknown mean mu given its unknown variance sigma2 to be drawn from N(nu.intervention, sigma2/lambda.intervention). Default: nu.intervention=0.6, lambda.intervention=4

nu.no_intervention, lambda.no_intervention The same parameters for unperturbed nodes. Default: nu.no_intervention=0.95, lambda.no_intervention=4

df.intervention, scale.intervention For `depn`: The unknown variance sigma2 for perturbed nodes is supposed to be drawn from Inv-χ^2(df.intervention, scale.intervention). Default: df.intervention=4.4, scale.intervention=4.4

df.no_intervention, scale.no_intervention The same parameters for unperturbed nodes. Default: df.no_intervention=4.4, scale.no_intervention=0.023

map For `depn`: Mapping of interventions to network nodes. The format is a named list of strings with names being the interventions and entries being the network nodes. Default: Entries and names are the network nodes.

outputdir Directory where to put diagnostic plots. Default: folder "QualityControl" in current working directory

debug Print out or plot diagnostic information. Default: FALSE

mc.cores number of cores to be used on a multicore processor. Default: 8

mcmc.nsamples Number of MCMC samples to take. Default: 1e6

mcmc.nburnin Number of additional samples for burnin phase. Default: 1e6

mcmc.seed random seed. Default: 1234

mcmc.hyperprior Parameter for exponential distribution hyperprior for regularization parameter 1/lambda. Default: 1

eminem.maxsteps Maximum number of iterations for the EM algorithm (MC.EMINEM). Default: 1000

eminem.sdVal positive number, between 1 and ncol(D)*(ncol(D)-1): number of edges to change in one MCMC step; see paper for the author's choice. Default: 1

eminem.changeHfreq positive number, mcmc.nsamples must be a multiple: the Empirical Bayes step is performed every <changeHfreq> steps (see paper for the author's choice); set >= mcmc.nsamples (or leave to default) to exclude this step. Default: NULL

prob.cutoff Only edges with probability > prob.cutoff are assumed to be present. Default: 0.5

Value

A list containing all parameters described above.

Author(s)

Holger Froehlich

References

Markowetz, F.; Bloch, J. & Spang, R., Non-transcriptional Pathway Features Reconstructed from Secondary Effects of RNA interference. Bioinformatics, 2005, 21, 4026 - 4032\

Markowetz, F.; Kostka, D.; Troyanskaya, O. & Spang, R., Nested Effects Models for High-dimensional Phenotyping Screens. Bioinformatics, 2007, 23, i305 - i312\

Fr\"ohlich, H.; Fellmann, M.; S\"ultmann, H.; Poustka, A. & Beissbarth, T. Large Scale Statistical Inference of Signaling Pathways from RNAi and Microarray Data. BMC Bioinformatics, 2007, 8, 386\

Fr\"ohlich, H.; Fellmann, M.; S\"ultmann, H.; Poustka, A. & Beissbarth, T. Estimating Large Scale Signaling Networks through Nested Effect Models with Intervention Effects from Microarray Data. Bioinformatics, 2008, 24, 2650-2656\

Tresch, A. & Markowetz, F., Structure Learning in Nested Effects Models Statistical Applications in Genetics and Molecular Biology, 2008, 7\

Zeller, C.; Fr\"ohlich, H. & Tresch, A., A Bayesian Network View on Nested Effects Models EURASIP Journal on Bioinformatics and Systems Biology, 2009, 195272\

Fr\"ohlich, H.; Tresch, A. & Beissbarth, T., Nested Effects Models for Learning Signaling Networks from Perturbation Data. Biometrical Journal, 2009, 2, 304 - 323\

Fr\"ohlich, H.; Sahin, \"O.; Arlt, D.; Bender, C. & Beissbarth, T. Deterministic Effects Propagation Networks for Reconstructing Protein Signaling Networks from Multiple Interventions. BMC Bioinformatics, 2009, 10, 322\

Fr\"ohlich, H.; Praveen, P. & Tresch, A., Fast and Efficient Dynamic Nested Effects Models. Bioinformatics, 2011, 27, 238-244\

Niederberger, T.; Etzold, S.; Lidschreiber, M; Maier, K.; Martin, D.; Fr\"ohlich, H.; Cramer, P.; Tresch, A., MC Eminem Maps the Interaction Landscape of the Mediator, PLoS Comp. Biol., 2012, submitted.

Examples

```
control = set.default.parameters(LETTERS[1:5], type="CONTmLLBayes", selEGenes=TRUE) # set inference type and wheth
```

`sim.intervention` *Simulate interventions in a Nested Effects Model*

Description

Simulates a knock-down of a list of network nodes and returns the network nodes and effect reporters, where effects are expected.

Usage

```
sim.intervention(x, int, T=1)
```

Arguments

x	nem object
int	a character vector of nodes in the network
T	number of time steps to be simulated (only for dynoNEMs)

Value

list with two slots:

Sgenes.effected
: effected network nodes

Egenes.effected
: effected downstream effect reporters

Author(s)

Holger Froehlich

Examples

```
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
res = nem(D, control=set.default.parameters(unique(colnames(D)), para=c(0.13,0.05)))
sim.intervention(res, "rel") # simulate knock-down of rel
```

---

subsets *Subsets*

Description

subsets

Usage

```
subsets(n, r, v = 1:n, set = TRUE)
```

Arguments

n	bli
r	bla
v	blo
set	blu

Details

taken from the programmers corner of some R-News issue by Dennis

Value

n	bli
r	bla
v	blo

Author(s)

Dennis Kostka

Examples

```
## bla
```

---

transitive.closure *Computes the transitive closure of a directed graph*

Description

Computes the transitive closure of a graph. Introduces a direct edge whenever there is a path between two nodes in a digraph.

Usage

```
transitive.closure(g, mat=FALSE, loops=TRUE)
```

Arguments

g	graphNEL object or adjacency matrix.
mat	convert result to adjacency matrix.
loops	Add loops from each node to itself?

Details

This function calculates the transitive closure of a given graph. We use the matrix exponential to find the transitive closure.

Value

returns a graphNEL object or adjacency matrix

Author(s)

Florian Markowetz

See Also

transitive.reduction

Examples

```
V <- LETTERS[1:3]
edL <- list(A=list(edges="B"),B=list(edges="C"),C=list(edges=NULL))
g <- new("graphNEL",nodes=V,edgeL=edL,edgemode="directed")
gc <- transitive.closure(g,loops=FALSE)

par(mfrow=c(1,2))
plot(g,main="NOT transitively closed")
plot(gc,main="transitively closed")
```

`transitive.projections` *Computes the transitive approximation of a directed graph*

Description

Computes the transitive approximation of a graph. The transitive approximation of a graph is a graph that is "almost" transitively closed and has minimal distance to the input graph.

Usage

```
transitive.projections(adjmat)
```

Arguments

`adjmat`	graphNEL object or adjacency matrix.

Value

returns adjacency matrices and having minimal graph distance to the input graph matrix

Author(s)

Juby Jacob

See Also

`transitive.projections`

`transitive.reduction` *Computes the transitive reduction of a graph*

Description

`transitive.reduction` removes direct edges, which can be explained by another path in the graph. Regulation directions inferred via `infer.edge.type` are taken into account.

Usage

```
transitive.reduction(g)
```

Arguments

g	adjacency matrix

Details

`transitive.reduction` uses a modification of the classical algorithm from the Sedgewick book for computing transitive closures. The so-called "transitive reduction" is neither necessarily unique (only for DAGs) nor minimal in the number of edges (this could be improved).

Value

returns an adjacency matrix with shortcuts removed

Author(s)

Holger Froehlich

References

R. Sedgewick, Algorithms, Pearson, 2002.

See Also

`transitive.closure`, `infer.edge.type`

Examples

```
V <- LETTERS[1:3]
edL <- list(A=list(edges=c("B","C")),B=list(edges="C"),C=list(edges=NULL))
gc <- new("graphNEL",nodes=V,edgeL=edL,edgemode="directed")
g <- transitive.reduction(gc)

par(mfrow=c(1,2))
plot(gc,main="shortcut A->C")
plot(as(g,"graphNEL"),main="shortcut removed")
```

Index

∗Topic **datasets**
BoutrosRNAi2002, 3
Ivanova2006RNAiTimeSeries, 12
NiederbergerMediator2012, 27
SahinRNAi2008, 35
∗Topic **graphs**
BFSlevel, 2
enumerate.models, 5
generateNetwork, 6
nem, 14
nem.bootstrap, 17
nem.consensus, 19
nem.cont.preprocess, 21
nem.discretize, 22
nemModelSelection, 25
network.AIC, 26
plotEffects, 29
prior.EgeneAttach.EB, 31
prune.graph, 32
quicknem, 33
SCCgraph, 36
set.default.parameters, 37
subsets, 41
transitive.closure, 42
transitive.projections, 43
transitive.reduction, 44
∗Topic **models**
closest.transitive.greedy, 4
generateNetwork, 6
getDensityMatrix, 8
getRelevantEGenes, 9
infer.edge.type, 10
internal, 12
local.model.prior, 13
nem, 14
nem.bootstrap, 17
nem.calcSignificance, 18
nem.consensus, 19
nem.cont.preprocess, 21
nem.discretize, 22
nem.jackknife, 24
nemModelSelection, 25
plot.nem, 28
prior.EgeneAttach.EB, 31
quicknem, 33
set.default.parameters, 37
sim.intervention, 40

BFSlevel, 2
binom.test, *11*
BoutrosRNAi2002, 3, *22, 23*, *36*
BoutrosRNAiDens *(*BoutrosRNAi2002*)*, 3
BoutrosRNAiDiscrete *(*BoutrosRNAi2002*)*, 3
BoutrosRNAiExpression *(*BoutrosRNAi2002*)*, 3
BoutrosRNAiLods *(*BoutrosRNAi2002*)*, 3
BoutrosRNAiLogFC *(*BoutrosRNAi2002*)*, 3
bum.dalt *(*internal*)*, 12
bum.EM *(*internal*)*, 12
bum.histogram *(*internal*)*, 12
bum.mle *(*internal*)*, 12
bum.negLogLik *(*internal*)*, 12
bum.palt *(*internal*)*, 12
bum.qalt *(*internal*)*, 12
bum.ralt *(*internal*)*, 12

CheckEdge *(*transitive.projections*)*, 43
closest.transitive.greedy, 4
connectModules *(*internal*)*, 12
createBN *(*internal*)*, 12

dat *(*Ivanova2006RNAiTimeSeries*)*, 12
dat.normalized *(*SahinRNAi2008*)*, 35
dat.unnormalized *(*SahinRNAi2008*)*, 35
data.likelihood *(*internal*)*, 12
dbum *(*internal*)*, 12
distdecrease *(*transitive.projections*)*, 43

distincrease (transitive.projections), 43
distincrease1 (transitive.projections), 43
distsame (transitive.projections), 43

EdgeEk (transitive.projections), 43
effect.likelihood (internal), 12
encode.interventions (internal), 12
enumerate.models, 5
enumerate.models2 (internal), 12
erase.cycles (internal), 12
exhaustive_BN (internal), 12

filterEGenes (getRelevantEGenes), 9
fit.BN (internal), 12
fitBUM (internal), 12
FourNeighborhood (transitive.projections), 43
FULLmLL, *10*
FULLmLL (internal), 12

generateNetwork, 6
getComponent (internal), 12
getDensityMatrix, *7*, 8
getRelevantEGenes, 9
graychange (internal), 12

infer.edge.type, 10, *29*, *44*
ingreed_BN (internal), 12
internal, 12
inv.logit (internal), 12
is.dag (internal), 12
is.transitive (transitive.projections), 43
Ivanova2006RNAiTimeSeries, 12, *13*

learn (internal), 12
local.model.prior, 13, *16*, *35*
logit (internal), 12

map.int2node (SahinRNAi2008), 35
mLL, *10*
mLL (internal), 12
moduleNetwork (internal), 12

nem, *6*, *10*, 14, *14*, *17–20*, *24–26*, *29*, *31*, *35*, *37*
nem.BN (internal), 12
nem.bootstrap, *16*, 17, *19, 20*, *25*, *34, 35*
nem.calcSignificance, 18, *18*, *20*
nem.consensus, *16*, *18*, 19, *19*, *25*, *35*
nem.cont.preprocess, 21
nem.discretize, *4*, 22
nem.featureselection (internal), 12
nem.greedy (internal), 12
nem.greedyMAP (internal), 12
nem.jackknife, *16*, *18–20*, 24, *35*
nemModelSelection, *16, 17*, *20*, *24*, 25, *25*, *27*, *35*
network.AIC, 26, *26*
NiederbergerMediator2012, 27
NiederbergerMediatorLods (NiederbergerMediator2012), 27
NiederbergerMediatorLogFC (NiederbergerMediator2012), 27
NiederbergerMediatorPVals (NiederbergerMediator2012), 27

OneNeighborhood (transitive.projections), 43
optimizecoregraph (internal), 12
optimizemarginal (internal), 12

pairwise.posterior, *14*
pairwise.posterior (internal), 12
parameters_continuous_Bayesian (internal), 12
parameters_continuous_ML (internal), 12
parameters_discrete_Bayesian (internal), 12
parameters_discrete_ML (internal), 12
pbum (internal), 12
PhiDistr (internal), 12
plot.dynoNEM (plot.nem), 28
plot.mc.eminem (plot.nem), 28
plot.ModuleNetwork (plot.nem), 28
plot.nem, *16*, 28, *35*
plot.pairwise (plot.nem), 28
plot.score (plot.nem), 28
plot.triples (plot.nem), 28
plotEffects, 29, *29*
plotnem (plot.nem), 28
print.dynoNEM (nem), 14
print.mc.eminem (nem), 14
print.ModuleNetwork (nem), 14
print.nem (nem), 14
print.nem.bootstrap (nem.bootstrap), 17
print.nem.consensus (nem.consensus), 19
print.nem.jackknife (nem.jackknife), 24

print.pairwise *(nem)*, 14
print.score *(nem)*, 14
print.triples *(nem)*, 14
prior.EgeneAttach.EB, 31
prune.graph, *5*, 32

qbum *(internal)*, 12
qqbum *(internal)*, 12
quicknem, 33

rbum *(internal)*, 12
remTwoEdges *(transitive.projections)*, 43

SahinRNAi2008, 35
sample.effect.likelihood *(internal)*, 12
sample.likelihood *(internal)*, 12
sampleData *(generateNetwork)*, 6
sampleRndNetwork *(generateNetwork)*, 6
SCCgraph, 36
score, *10*
score *(internal)*, 12
score_BN *(internal)*, 12
score_continuous_Bayesian *(internal)*, 12
score_continuous_ML *(internal)*, 12
score_discrete_Bayesian *(internal)*, 12
score_discrete_ML *(internal)*, 12
selectEGenes *(getRelevantEGenes)*, 9
set.default.parameters, *16*, *26*, *31*, *35*, 37
sim.intervention, 40
sim.interventions *(internal)*, 12
subsets, 41

ThreeNeighborhood
        (transitive.projections), 43
transitive.closure, *5*, 42, *44*
transitive.projections, 43, *43*
transitive.reduction, *5*, *37*, *42*, 44
transSubGr *(transitive.projections)*, 43
triples.posterior *(internal)*, 12
TwoNeighborhood
        (transitive.projections), 43

VecToMat *(transitive.projections)*, 43

which.is.max *(internal)*, 12