Description

BFSlevel builds a (generalized) hierarchy by Breath-First Search as described in (Yu and Ger-
stein, 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 net-
works, PNAS 103(40):14724-14731, 2006

Value

level: vector of levels for each node

Author(s)

Florian Markowetz <URL: http://genomics.princeton.edu/~florian>

Examples

```r
# bla
```
**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**

```r
data(BoutrosRNAi2002)
```

**Format**

- **BoutrosRNAiExpression**: data matrix: 14010 x 16
- **BoutrosRNAiDiscrete**: binary matrix: 68 x 16

**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**


**See Also**

- `nem.discretize`

**Examples**

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

Full marginal likelihood of a phenotypic hierarchy

Description
The function the full marginal likelihood of a phenotypic hierarchy. The full marginal likelihood equals the marginal likelihood mLL averaged over the error probabilities α and β.

Usage
FULLmLL(Phi, D1, D0, a0, b0, a1, b1, Pe, Pm=NULL, lambda=0)

Arguments
- **Phi**: an adjacency matrix with unit main diagonal
- **D1**: count matrix: phenotypes x genes. How often did we see an effect after interventions?
- **D0**: count matrix: phenotypes x genes. How often did we NOT see an effect after intervention?
- **a0, b0, a1, b1**: Hyperparameters
- **Pe**: prior of effect positions in the hierarchy. A matrix of size phenotypes x genes, where each row contains positive numbers summing to 1.
- **Pm**: prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.
- **lambda**: regularization parameter to incorporate prior assumptions.

Details
Additionally to the marginal likelihood introduced in Markowetz et al (2005), we can average over the error probabilities α and β assuming Beta priors. The parameters of the two Beta priors are hyperparameters of the full marginal likelihood score. The four hyperparameters fall into two categories: a1 and b0 are weights for observing the predicted state, while a0 and b1 are weights for observing errors. We suggest setting a1=b0 and a0=b1. The ratio between the two values should correspond to our assessment of the noise level. See the example section for an application. The function FULLmLL is usually called from within function score.

Value
- **mLL**: full marginal likelihood of a model
- **pos**: posterior distribution of effect positions in the hierarchy
- **mappos**: maximum a posteriori estimate of effect positions
- **LLperGene**: likelihood per E-gene

Author(s)
Florian Markowetz <URL: http://genomics.princeton.edu/~florian>
SCCgraph

References


See Also

nem, score, mLL

Examples

data("BoutrosRNAi2002")
res <- nem(BoutrosRNAiDiscrete[,9:16], type="FULLmLL", hyperpara=c(1,9,9,1))

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 connected 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 <URL: http://genomics.princeton.edu/~florian>, Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>

See Also

nem, transitive.reduction
**closest.transitive.greedy**

Find transitive closed graph most similar to the given one

**Description**

First, from the original graph $\Phi$ spurious edges are pruned via `prune.graph`. Then the new graph $\Phi'$ 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`, `prune.graph`
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, 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`
- `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 <URL: http://genomics.princeton.edu/~florian>

See Also

`score`, `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**

```r
sampleRndNetwork(Sgenes, scaleFree=TRUE, gamma=2.5, maxOutDegree=length(Sgenes),
                  trans.close=TRUE, DAG=FALSE)

sampleData(Phi, m, prob=NULL, uninformative=0, type="binary", replicates=4, typeI.err=0.05, typeII.err=0.2, ...
```

**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 $\frac{1}{Z} \cdot (0 : \text{maxOutDegree})^{-\gamma}$, 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: $\lambda_1 + \lambda_2 \ast Beta(\alpha, 1) + \lambda_3 \ast Beta(1, \beta)$. 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
getDensityMatrix

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 \texttt{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 \texttt{sampleRndNetwork} an adjacency matrix, for \texttt{sampleData} a data matrix, for \texttt{sampleData.BN} a data matrix and a linking of effects to signals.

Author(s)

Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>, Cordula Zeller

See Also

getchDensityMatrix

Examples

```r
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(D, type="CONTLLBayes"), transitiveReduction=TRUE, SCC=FALSE, main="inferred graph")
```

---

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

```r
getDensityMatrix(Porig, dirname=NULL, startab=c(0.3,10), startlam=c(0.6,0.1,0.3),
```
infer.edge.type

Arguments

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

Details

The BUM density model consists of 3 components:

\[ f(x) = \lambda_1 + \lambda_2 \cdot \text{dbeta}(x, \alpha, 1) + \lambda_3 \cdot \text{dbeta}(x, 1, \beta). \]

The mixing coefficients and the parameters alpha and beta are fitted together via an EM algorithm.

Value

log-density matrix of the same dimensions as Porig

Note

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

Author(s)

Holger Froehlich

infer.edge.type Infer regulation direction for each edge

Description

The method infers edge types (up-regulation, down-regulation) for a given nem model. 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")

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>nem object</td>
</tr>
<tr>
<td>logFC</td>
<td>matrix with fold changes. The rownames of this matrix should correspond to</td>
</tr>
<tr>
<td></td>
<td>the rownames of the data matrix, which was used to infer the nem model.</td>
</tr>
<tr>
<td>alpha</td>
<td>p-value cutoff</td>
</tr>
<tr>
<td>adj.method</td>
<td>multiple testing correction method. Default: Benjamini-Yekutieli</td>
</tr>
</tbody>
</table>
Details

Significance 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 corresponds to an unknown regulation type.

Author(s)

Holger Froehlich

See Also

binom.test

Examples

data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
p <- c(.13,.05)
result = nem(D, para=p)
resEdgeInf = infer.edge.type(result, BoutrosRNAiLogFC)
plot(resEdgeInf)

Description

internal functions: do not call these functions directly.

Usage

various

Arguments

various

Value

various

Author(s)

Holger Froehlich
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 <URL: http://genomics.princeton.edu/~florian>

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)
mLL

Marginal likelihood of a phenotypic hierarchy

Description
computes the marginal likelihood of observed phenotypic data given a phenotypic hierarchy.

Usage
mLL(Phi,D1,D0=NULL,a=0.15,b=0.05,Pe=NULL,Pm=NULL,lambda=0,type="mLL")

Arguments
Phi an adjacency matrix with unit main diagonal
D1 (i) count matrix for discrete data: phenotypes x genes. How often did we see an effect after interventions? (ii) matrix describing the PROBABILITIES of an effect (iii) matrix describing the log-LIKELIHOOD of an effect (e.g. log-density matrix, log-odds matrix)
D0 count matrix: phenotypes x genes. How often did we NOT see an effect after intervention? Not used for continuous data
a false positive rate: how probable is it to miss an effect? (for count matrix)
b false negative rate: how probable is it to see a spurious effect? (for count matrix)
Pe prior of effect reporter positions in the phenotypic hierarchy
Pm prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.
lambda regularization parameter to incorporate prior assumptions.
type see nem

Details
It computes the marginal likelihood of a single phenotypic hierarchy. Usually called from within the function score.

Value
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 E-gene

Author(s)
Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>, Florian Markowetz <URL: http://genomics.princeton.edu/~florian>

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

See Also

nem, score, FULLmLL

Examples

```r
data("BoutrosRNAi2002")
result <- nem(BoutrosRNAiDiscrete[,9:16],type="mLL",para=c(.15,.05))
```

---

moduleNetwork **Infers a phenotypic hierarchy using the module network**

**Description**

Function `moduleNetwork` estimates the hierarchy using a divide and conquer approach. In each step only a subset of nodes (called module) is involved and no exhaustive enumeration of model space is needed as in function `score`.

**Usage**

```r
moduleNetwork(D,type="mLL",Pe=NULL,Pm=NULL,lambda=0,delta=1,para=NULL,hyperpara=NULL,verbose=TRUE)
## S3 method for class 'ModuleNetwork':
print(x,...)
```

**Arguments**

- `D` data matrix. Columns correspond to the nodes in the silencing scheme. Rows are phenotypes.
- `type` see `nem`
- `Pe` prior position of effect reporters. Default: uniform over nodes in hierarchy
- `Pm` prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.
- `lambda` regularization parameter to incorporate prior assumptions.
- `delta` regularization parameter for automated E-gene subset selection (CONTmLLRatio only)
- `para` vector with parameters a and b for "mLL", if count matrices are used
- `hyperpara` vector with hyperparameters a0, b0, a1, b1 for "FULLmLL"
- `verbose` do you want to see progress statements printed or not? Default: TRUE
- `x` nem object
- `...` other arguments to pass

**Details**

`moduleNetwork` is an alternative to exhaustive search by the function `score` and more accurate than `pairwise.posterior` and `triples.posterior`. It uses clustering to successively split the network into smaller modules, which can then be estimated completely. Connections between modules are estimated by performing a constraint greedy hillclimbing.
nem.BN

Value
nem object

Author(s)
Holger Froehlich

References


See Also
score, nem

Examples

```r
data("BoutrosRNAi2002")
res <- nem(BoutrosRNAiDiscrete[,9:16],para=c(.13,.05),inference="ModuleNetwork")

# plot graph
plot(res,what="graph")

# plot posterior over effect positions
plot(res,what="pos")

# estimate of effect positions
res$mappos
```

---

**nem.BN**

*Bayesian Network Nested Effects Models*

**Description**

Uses a Bayesian network interpretation of Nested Effects Models to estimate the signals graph.

**Usage**

```r
nem.BN(D, inference="greedy", mode="binary_ML", lambda=0, verbose=TRUE)
```
Arguments

D  data matrix with experiments in the columns (binary or continuous)

inference  exhaustive to use exhaustive enumeration; or greedy for optimizing the linking of effects to signals and the signals graph in an alternating fashion

mode  binary_ML: effects come from a binomial distribution - ML learning of parameters; binary_Bayesian: effects come from a binomial distribution - Bayesian learning of parameters with beta distribution prior; continuous_ML: effects come from a normal distribution - ML learning of parameters; continuous_Bayesian: effects come from a normal distribution - Bayesian learning of parameters with gamma distribution prior.

lambda  regularization parameter to incorporate prior assumptions. Not used so far.

verbose  do you want to see progression statements? Default: TRUE

Details

plot.nem plots the inferred phenotypic hierarchy as a directed graph.

Value

An object of class ‘nem.BN’

graph  the inferred phenotypic hierarchy
mLL  log (posterior) marginal likelihood
mappos  estimated position of effects in the phenotypic hierarchy
selected  selected E-gene subset
type  = mode in function call
lambda  see above

Author(s)

Cordula Zeller, Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>

See Also

plot.nem

Description

The main function to infer a phenotypic hierarchy from data

Usage

nem(D,inference="nem.greedy",models=NULL,type="mLL",para=NULL,hyperpara=NULL,Pe=NULL,Pm=NULL,Pmlocal=NULL,local.prior.size ... (unique(colnames(D))),local.prior.bias=1,triples.thrsh=0.5,lambda=0,delta=1,selEGenes=FALSE,trans.close=TRUE,verbose=TRUE)

## S3 method for class 'nem':
print(x, ...)
Arguments

D data matrix with experiments in the columns (binary or continuous)
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, enumerate.models is used for exhaustive enumeration of all possible models.
type mLL or FULLmLL or CONTmLL or CONTmLLBayes or CONTmLLMAP, 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, where the linking positions of E-genes to S-Genes are integrated out, and CONTmLLMAP to an inference scheme, were a MAP estimate for the linking positions is calculated.
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)
Pm prior over models (n x n matrix)
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)
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
lambda regularization parameter to incorporate prior assumptions. Default: 0 (no regularization)
delta regularization parameter for automated E-gene subset selection (CONTmLLMAP only)
selectEGenes automated E-gene subset selection (includes tuning of delta for CONTmLLMAP)
trans.close Should always transitive closed graphs be computed? Default: TRUE. NOTE: This has only an impact for the nem.greedyMAP method.
verbose do you want to see progression statements? Default: TRUE
x nem object
... other arguments to pass
**nem**

**Details**

nem is an interface to the functions score, pairwise.posterior, triple.posterior, moduleNetwork, nem.greedy and nem.greedyMAP. If Pm != NULL and 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 E-gene subset selection is used and type == CONTmLLMAP, the regularization parameter delta is tuned via the AIC model selection criterion. Otherwise, an iterative algorithm is executed, which in an alternating optimization scheme reconstructs a network given the current set of E-gene and then selects the E-genes having the highest likelihood under the given network. The procedure is run until convergence.

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**: 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
- **type**: as used in function call
- **para**: as used in function call
- **hyperpara**: as used in function call
- **lambda**: as in function call
- **delta**: as in function call
- **selected**: selected E-gene subset
- **LLperGene**: likelihood per selected E-gene

**Author(s)**

Florian Markowetz <URL: http://genomics.princeton.edu/~florian>, Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>

**See Also**

nemModelSelection, nem.jackknife, nem.bootstrap, nem.consensus, score, moduleNetwork, nem.greedy, triples.posterior, pairwise.posterior, nem.greedyMAP, local.model.prior, enumerate.models, plot.nem

**Examples**

```r
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
p <- c(.13,.05)
res1 <- nem(D,para=p,inference="search")
res2 <- nem(D,para=p,inference="pairwise")
res3 <- nem(D,para=p,inference="triples")
res4 <- nem(D,para=p,inference="ModuleNetwork")
res5 <- nem(D,para=p,inference="nem.greedy")
res6 = nem(BoutrosRNAiLods, inference="nem.greedyMAP")
```
nem.bootstrap

Bootstrapping for nested effect models

Description
Performs bootstrapping (resampling with replacement) on E-genes to assess the statistical stability of networks.

Usage
nem.bootstrap(D, thresh=0.5, nboot=1000, inference="nem.greedy", models=NULL, type="mLL")

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

Arguments
- **D**: data matrix with experiments in the columns (binary or continuous)
- **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
- **type**: mLL or FULLmLL or CONTmLL or CONTmLLBayes or CONTmLLMAP, see nem
- **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)
- **Pm**: prior over models (n x n matrix)
- **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)
- **local.prior.bias**: bias towards double-headed edges. Default: 1 (no bias; for pairwise learning)
nem.calcSignificance

triples.thresh
    threshold for model averaging to combine triple models for each edge
lambda
    regularization parameter to include prior assumptions
delta
    regularization parameter for automated E-gene subset selection (CONTmLLRatio only)
selEGenes
    automated E-gene subset selection (includes tuning of delta for CONTmLLRatio)
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

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]
p <- c(.13,.05)
nem.bootstrap(D, para=p)
## End(Not run)
Arguments

- **D**: data matrix with experiments in the columns (binary or continuous)
- **x**: nem object
- **N**: number of random networks to sample
- **seed**: random seed
- **Pe**: prior of effect reporter positions in the phenotypic hierarchy (same dimension as D)
- **Pm**: prior over models (n x n matrix)
- **selEGenes**: automated E-gene subset selection yes/no

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 or deletion of 1 edge) yields an exact p-value describing the stability of the network.

Value

- **p.value.rnd**: p-value of the network according to the null hypothesis that it is random
- **p.value.perm**: p-value of the network according to the null hypothesis that a network with permuted node labels is at least as good
- **p.value.mod**: p-value of the network according to the null hypothesis a randomly perturbed network is at least as good

Author(s)

- Holger Froehlich

See Also

- `nem.consensus`, `nem.jackknife`, `nem.bootstrap`, `nem`

Examples

```r
## Not run:
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
p <- c(.13,.05)
res = nem(D, para=p) # 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, type="mLL", para=NULL, hyperpara=NULL, Pe=NULL, Pm=NULL, Pmlocal=NULL, local.prior.size=5000, local.prior.bias=1, triples.thrsh=0.5, lambda=0, delta=1, selEGenes=FALSE, verbose=TRUE)

## 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; or triples for triple-based inference; or pairwise for the pairwise heuristic; or ModuleNetwork for the module based inference; or nem.greedy for greedy hillclimbing
- **models**
  - a list of adjacency matrices for model search
- **type**
  - mLL or FULLmLL or CONTmLL or CONTmLLBayes or CONTmLLMAP, see nem
- **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)
- **Pm**
  - prior over models (n x n matrix)
- **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)
- **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
- **lambda**
  - regularization parameter to include prior assumptions
- **delta**
  - regularization parameter for automated E-gene subset selection (CONTmLLMAP only)
nem.cont.preprocess

Description

Calculate classification probabilities of perturbation data according to control experiments

Usage

nem.cont.preprocess(D, neg.control=NULL, pos.control=NULL, nfold=2, influencefactor=NULL)

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]
 p <- c(.13,.05)
 nem.consensus(D, para=p)
## End(Not run)
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 perturbation 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 <URL: http://genomics.princeton.edu/~florian>

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, verbose=TRUE)

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 <URL: http://genomics.princeton.edu/~florian>

References

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

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.greedy  

Infers a phenotypic hierarchy using a greedy search strategy

Description

Starting from an initial graph (default: no edges), this strategy successively adds those edges, which most increase the likelihood of the data under the model.

Usage

nem.greedy(D,initial=NULL,type="mLL",Pe=NULL,Pm=NULL,lambda=0,delta=1,para=NULL,hyperpara=NULL,verbose=TRUE)
## S3 method for class 'nem.greedy':
print(x,...)

Arguments

D  
data matrix. Columns correspond to the nodes in the silencing scheme. Rows are phenotypes.

initial  
initial model to start greedy hillclimbing from (default: no edges)

type  
see nem

Pe  
prior position of effect reporters. Default: uniform over nodes in hierarchy

Pm  
prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.

lambda  
regularization parameter to incorporate prior assumptions.

delta  
regularization parameter for automated E-gene subset selection (CONTmLLRatio only)

para  
vector with parameters a and b for "mLL", if count matrices are used

hyperpara  
vector with hyperparameters a0, b0, a1, b1 for "FULLmLL"

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

x  
nem object

...  
other arguments to pass

Value

nem object
nem.greedyMAP

Author(s)
Holger Froehlich

See Also
nem

Examples

# Drosophila RNAi and Microarray Data from Boutros et al, 2002
data("BoutrosRNAi2002")
D <- BoutrosRNAIDiscrete[,9:16]
nem(D, para=c(.13,.05), inference="nem.greedy")

nem.greedyMAP

Infers a phenotypic hierarchy using an alternating MAP optimization

Description
Starting with an initial estimate of the linking of E-genes to S-genes from the data, this method performs an alternating MAP optimization of the S-genes graph and the linking graph until convergence. As a final step the function closest.transitive.greedy can be invoked to find a transitivity closed graph most similar to the original one.

Usage
nem.greedyMAP(D,Pe=NULL,Pm=NULL,lambda=0,delta=1, trans.close=TRUE, verbose=TRUE)

## S3 method for class 'nem.greedyMAP':
print(x, ...)

Arguments

D data matrix. Columns correspond to the nodes in the silencing scheme. Rows are phenotypes.
Pe prior position of effect reporters. Default: uniform over nodes in hierarchy
Pm prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.
lambda regularization parameter to incorporate prior assumptions.
delta regularization parameter for automated E-gene subset selection
trans.close find a similar transitively closed graph
verbose do you want to see progress statements printed or not? Default: TRUE
x nem object
...
other arguments to pass

Value
nem object
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, type="mLL", para=NULL)

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

Arguments

D data matrix with experiments in the columns (binary or continuous)
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, enumerate.models is used for exhaustive enumeration of all possible models.
type mLL or FULLmLL or CONTmLL or CONTmLLBayes or CONTmLLMAP, see nem
nem.jackknife

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)
Pm prior over models (n x n matrix)
Pmlocal local model prior for pairwise and triplets 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)
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
lambda regularization parameter to incorporate prior assumptions. Default: 0 (no regularization)
delta regularization parameter for automated E-gene subset selection (CONTmLLRatio only)
 selEGenes automated E-gene subset selection (includes tuning of delta for CONTmLLRatio)
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

nem object with edge weights being the jackknife probabilities

Author(s)

Holger Froehlich

See Also

nem.bootstrap, nem.consensus, nem, nemModelSelection

Examples

```r
## Not run:
data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
p <- c(.13,.05)
nem.jackknife(D, para=p)
## End(Not run)
```
nemModelSelection  

**model selection for nested effect models**

**Description**

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

**Usage**

```r
nemModelSelection(lambdas, D, inference="nem.greedy", models=NULL, type="mLL", para=NULL, hyperpara=NULL, Pe=NULL, Pmlocal=NULL, Pm NULL, triple...)
```

**Arguments**

- `lambdas`: vector of regularization constants
- `D`: data matrix with experiments in the columns (binary or continuous)
- `inference`: search to use exhaustive enumeration; or triples for triple-based inference; or pairwise for the pairwise heuristic; or ModuleNetwork for the module based inference; or nem.greedy for the greedy hillclimbing
- `models`: a list of adjacency matrices for model search. If NULL, enumerate.models is used for exhaustive enumeration of all possible models.
- `type`: mLL or FULLmLL or CONTmLL or CONTmLLBayes or CONTmLLMAP, see nem
- `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)
- `Pm`: prior over models (n x n matrix)
- `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)
- `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
- `delta`: regularization parameter for automated E-gene subset selection (CONTmLLMAP only)
- `selEGenes`: automated E-gene subset selection (includes tuning of delta for CONTmLLMAP)
- `trans.close`: Should always transitive closed graphs be computed? Default: TRUE. NOTE: This has only an impact for the nem.greedyMAP method.
- `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

nem, network.AIC

Examples

data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
p <- c(.13,.05)
res <- nemModelSelection(c(0.1,1,10),D, para=p, Pm=matrix(0,ncol=4,nrow=4))

plot(res,main="highest scoring model")

network.AIC

AIC criterion for network graph

Description

Calculate AIC 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=2,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 calculation. Default = 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
pairwise.posterior

Value

AIC value

Author(s)

Holger Froehlich

See Also

nemModelSelection

Examples

data("BoutrosRNAi2002")
res1 <- nem.greedy(BoutrosRNAiDiscrete[,9:16],para=c(.13,.05))
network.AIC(res1)
res2 <- nem.greedy(BoutrosRNAiDiscrete[,9:16],para=c(.13,.05),lambda=10)
network.AIC(res2)

pairwise.posterior  Infers a phenotypic hierarchy edge by edge

Description

Function pairwise.posterior estimates the hierarchy edge by edge. In each step only a pair of nodes is involved and no exhaustive enumeration of model space is needed as in function score.

Usage

pairwise.posterior(D, type = "mLL", para = NULL, hyperpara = NULL,
Pe = NULL, Pmlocal = NULL, Pm = NULL, lambda = 0, delta=1, verbose = TRUE)

## S3 method for class 'pairwise':
print(x,...)

Arguments

D           data matrix. Columns correspond to the nodes in the silencing scheme. Rows are phenotypes.
type        see nem
para         vector with parameters a and b for "mLL", if count matrices are used
hyperpara    vector with hyperparameters a0, b0, a1, b1 for "FULLmLL"
Pe           prior position of effect reporters. Default: uniform over nodes in hierarchy
Pmlocal      local model prior for the four models tested at each node: a vector of length 4 with positive entries summing to one
Pm           prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.
lambda       regularization parameter to incorporate prior assumptions.
plot.nem

delta

regularization parameter for automated E-gene subset selection (CONTmLLRa-tio only)

verbose

do you want to see progress statements printed or not? Default: TRUE

x

nem object

... other arguments to pass

Details

pairwise.posterior is a fast(er) heuristic alternative to exhaustive search by the function score. For each pair \((A,B)\) of perturbed genes it chooses between four possible models: \(A..B\) (unconnected), \(A->B\) (superset), \(A<-B\) (subset), or \(A<->B\) (indistinguishable). The result is the graph built from the maximum aposteriori models for each edge.

print.pairwise gives an overview over the 'pairwise' object.

Value

nem object

Author(s)

Florian Markowetz <URL: http://genomics.princeton.edu/~florian>

See Also

score, nem

Examples

data("BoutrosRNAi2002")
res <- nem(BoutrosRNAiDiscrete[,9:16],para=c(.13,.05),inference="pairwise")

# plot graph
plot(res,what="graph")

# plot posterior over effect positions
plot(res,what="pos")

# estimate of effect positions
res$mappos

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", ...)
plotEffects

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
...
other arguments to be passed to the Rgraphviz plot function or to the graphics 'image' function.

Value
none

Author(s)
Florian Markowetz <URL: http://genomics.princeton.edu/~florian>, Holger Froehlich <URL: http://www.dkfz.de/mga2/people/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,...)

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?
...     additional parameters for the graphics function 'image'

Details

The experiments in the columns are reordered according to the topological order given by a pheno-
notypic 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 <URL: http://genomics.princeton.edu/~florian>, Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>

Examples

data("BoutrosRNAi2002")
D <- BoutrosRNAiDiscrete[,9:16]
res <- nem(D,para=c(.13,.05))
plotEffects(D,res)
**prune.graph**  
*Prunes spurious edges in a phenotypic hierarchy*

**Description**
A heuristic to prune spurious edges in a phenotypic hierarchy

**Usage**
```r
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 spurious 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 <URL: http://genomics.princeton.edu/~florian>

**Examples**
```r
# 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
```
```r
gP <- prune.graph(g)

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

---

**score**

Computes the marginal likelihood of phenotypic hierarchies

### Description

Function to compute the marginal likelihood of a set of phenotypic hierarchies.

### Usage

```r
score(models, D, type="mLL", para=NULL, hyperpara=NULL, Pe=NULL, Pm=NULL, lambda=0, delta=1, verbose=TRUE, graphClass="graphNEL")
```

### Arguments

- **models**: a list of adjacency matrices with unit main diagonal
- **D**: data matrix. Columns correspond to the nodes in the silencing scheme. Rows are effect reporters.
- **type**: `mLL`, `FULLmLL` or `CONTmLL` for binary data (see `BoutrosRNAiDiscrete`) and `CONTmLLBayes`, `CONTmLLMAP`, `CONTmLLDens` and `CONTmLLRatio` 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.
- **para**: Vector with parameters `a` and `b` (for "mLL" with count data)
- **hyperpara**: Vector with hyperparameters `a0, b0, a1, b1` for "FULLmLL"
- **Pe**: prior position of effect reporters. Default: uniform over nodes in silencing scheme
- **Pm**: prior on model graph (n x n matrix) with entries 0 <= `priorPhi[i,j]` <= 1 describing the probability of an edge between gene i and gene j.
- **lambda**: regularization parameter to incorporate prior assumptions.
- **delta**: regularization parameter for automated E-gene subset selection (CONTmLLRatio only)
- **verbose**: output while running or not
Score models by marginal log-likelihood is implemented in function `score`. Input consists of models and data, the type of the score ("mLL", "FULLmLL", "CONTmLL" or "CONTmLLBayes" or "CONTmLLMAP"), the corresponding parameters (para) or hyperparameters (hyperpara), a prior for phenotype positions (Pe) and model structures (Pm) with regularization parameter lambda. If a structure prior (Pm) is provided, but no regularization parameter lambda, Bayesian model averaging with an inverse gamma prior on 1/lambda is performed. With type "CONTmLLMAP" usually an automated selection of most relevant E-genes is performed by introducing a "null" S-gene. The corresponding prior probability of leaving out an E-gene is set to delta/no. S-genes.

`score` is usually called within function `nem`.

**Value**

nem object

**Author(s)**

Holger Froehlich <URL: http://www.dkfz.de/mga2/people/froehlich>, Florian Markowetz <URL: http://genomics.princeton.edu/~florian>

**References**


**See Also**

nem, mLL, FULLmLL, enumerate.models
getRelevantEGenes

Automatic selection of most relevant E-genes

Description

1. A-priori filtering of E-genes: Select E-genes, which show a pattern of differential expression across experiments that is expected to be non-random. 2. Automated E-gene subset selection: Select those E-genes, 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, para=NULL, hyperpara=NULL, Pe=NULL, Pm=NULL, lambda=0, delta=1, type="CONTmLLDens", nEgenes=min(10*nrow(Phi), nrow(D))

Arguments

For method filterEGenes:

Porig matrix of raw p-values, typically from the complete array
D 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
Phi      adjacency matrix with unit main diagonal
type     mLL or FULLmLL or CONTmLL or CONTmLLBayes or CONTmLLMAP. CONTmLLDens and CONTmLLRatio are identical to CONTmLLBayes and CONTmLLMAP and are still supported for compatibility reasons, see nem.
para     Vector with parameters a and b (for "mLL" with count data)
hyperpara Vector with hyperparameters a0, b0, a1, b1 for "FULLmLL"
Pe       prior position of effect reporters. Default: uniform over nodes in silencing scheme
Pm       prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.
lambda   regularization parameter to incorporate prior assumptions.
delta    regularization parameter for automated E-gene subset selection (CONTmLLMAP only)
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
models <- enumerate.models(unique(colnames(D)))

getRelevantEGenes(models[[64]], D, para=c(.13,.05), type="mLL")

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 <URL: http://www.molgen.mpg.de/~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 <URL: http://genomics.princeton.edu/~florian>

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


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")
```

triples.posterior

Infers a phenotypic hierarchy from triples

Description

Function `triples.posterior` estimates the hierarchy triple-wise. In each step only a triple of nodes is involved and no exhaustive enumeration of model space is needed as in function `score`.

Usage

```
triples.posterior(D, type="mLL", para=NULL, hyperpara=NULL, Pe=NULL, Pmlocal=NULL, Pm=NULL, lambda=0, delta=1, triples.thrsh=.5, verbose=TRUE)
```

## S3 method for class 'triples':

```
print(x,...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>data matrix. Columns correspond to the nodes in the silencing scheme. Rows are phenotypes.</td>
</tr>
<tr>
<td>type</td>
<td>see <code>nem</code></td>
</tr>
<tr>
<td>para</td>
<td>vector with parameters a and b for &quot;mLL&quot;, if count matrices are used</td>
</tr>
<tr>
<td>hyperpara</td>
<td>vector with hyperparameters a0, b0, a1, b1 for &quot;FULLmLL&quot;</td>
</tr>
<tr>
<td>Pe</td>
<td>prior position of effect reporters. Default: uniform over nodes in hierarchy</td>
</tr>
<tr>
<td>Pmlocal</td>
<td>local model prior for the four models tested at each node: a vector of length 4 with positive entries summing to one</td>
</tr>
</tbody>
</table>
triples.posterior

threshold used when combining triple models for each edge. Default: only edges appearing in more than half of triples are included in the final graph.

prior on model graph (n x n matrix) with entries 0 <= priorPhi[i,j] <= 1 describing the probability of an edge between gene i and gene j.

regularization parameter to incorporate prior assumptions.

regularization parameter for automated E-gene subset selection (CONTmLLRatio only)

do you want to see progress statements printed or not? Default: TRUE

nem object

details

triples.posterior is an alternative to exhaustive search by the function score and more accurate than pairwise.posterior. For each triple of perturbed genes it chooses between the 29 possible models. It then uses model averaging to combine the triple-models into a final graph.

print.triples gives an overview over the 'triples' object.

Value

nem object

Author(s)

Florian Markowetz <URL: http://genomics.princeton.edu/~florian>

References


See Also

score,nem

Examples

data("BoutrosRNAi2002")
res <- nem(BoutrosRNAiDiscrete[,9:16],para=c(.13,.05),inference="triples")

# plot graph
plot(res,what="graph")

# plot posterior over effect positions
plot(res,what="pos")

# estimate of effect positions
res$mappos
Index

*Topic datasets
  BoutrosRNAi2002, 2

*Topic graphs
  BFSlevel, 1
  enumerate.models, 6
  generateNetwork, 7
  moduleNetwork, 13
  nem, 15
  nem.BN, 14
  nem.bootstrap, 18
  nem.consensus, 21
  nem.cont.preprocess, 22
  nem.discretize, 24
  nemModelSelection, 29
  network.AIC, 30
  pairwise.posterior, 31
  plotEffects, 33
  prune.graph, 35
  SCCgraph, 4
  subsets, 40
  transitive.closure, 41
  transitive.projections, 42
  transitive.reduction, 42
  triples.posterior, 43

*Topic models
  closest.transitive.greedy, 5
  FULLmLL, 3
  generateNetwork, 7
  getDensityMatrix, 8
  getRelevantEGenes, 38
  infer.edge.type, 9
  internal, 10
  local.model.type, 11
  mLL, 12
  nem, 15
  nem.BN, 14
  nem.bootstrap, 18
  nem.calcSignificance, 19
  nem.consensus, 21
  nem.cont.preprocess, 22
  nem.discretize, 24
  nem.greedy, 25
  nem.greedyMAP, 26
  nem.jackknife, 27
  nemModelSelection, 29
  plot.nem, 32
  score, 36

BFSlevel, 1
  binom.test, 10
  BoutrosRNAi2002, 2, 23, 25
  BoutrosRNAiDens
    (BoutrosRNAi2002), 2
  BoutrosRNAiDiscrete
    (BoutrosRNAi2002), 2
  BoutrosRNAiExpression
    (BoutrosRNAi2002), 2
  BoutrosRNAiLods
    (BoutrosRNAi2002), 2
  BoutrosRNAiLogFC
    (BoutrosRNAi2002), 2
  bum.dalt (internal), 10
  bum.EM (internal), 10
  bum.histogram (internal), 10
  bum.mle (internal), 10
  bum.negLogLik (internal), 10
  bum.palt (internal), 10
  bum.qalt (internal), 10
  bum.ralt (internal), 10
  CheckEdge
    (transitive.projections), 42
  closest.transitive.greedy, 5, 27
  connectModules (internal), 10
  createBN (internal), 10
  dbum (internal), 10
  distdecrease
    (transitive.projections), 42
  distincrease
    (transitive.projections), 42
  distincreasel
    (transitive.projections), 42
distsame
  (transitive.projections), 42
EdgeEk (transitive.projections), 42
enumerate.models, 6, 17, 37
exhaustive_BN (internal), 10
filterEGenes (getRelevantEGenes), 38
fit.BN (internal), 10
fitBUM (internal), 10
FourNeighborhood
  (transitive.projections), 42
FULLmLL, 3, 13, 37, 39
generateNetwork, 7
component (internal), 10
densityMatrix, 8, 8
getRelevantEGenes, 38
graychange (internal), 10
infer.edge.type, 9, 33, 43
ingreed_BN (internal), 10
internal, 10
inv.logit (internal), 10
is.dag (internal), 10
is.transitive
  (transitive.projections), 42
local.model.prior, 11, 17
logit (internal), 10
mLL, 4, 12, 37, 39
moduleNetwork, 13, 17
moduleNetwork.aux (internal), 10
nem, 4, 6, 11, 13, 14, 15, 19, 20, 22, 26–28, 30, 32, 33, 37, 39, 44
nem.BN, 14
nem.bootstrap, 17, 18, 20, 22, 28
nem.calcSignificance, 19, 19, 22
nem.consensus, 17, 19, 20, 21, 28
nem.cont.preprocess, 22
nem.discretize, 2, 24
nem.featureselection (internal), 10
nem.greedy, 17, 25
nem.greedyMAP, 17, 26
nem.jackknife, 17, 19, 20, 22, 27
nemModelSelection, 17, 19, 22, 28, 29, 31
network.AIC, 30, 30
OneNeighborhood
  (transitive.projections), 42
optimizecoregraph (internal), 10
optimizemarginal (internal), 10
pairwise.posterior, 11, 17, 31
parameters_continuous_Bayesian (internal), 10
parameters_continuous_ML (internal), 10
parameters_discrete_Bayesian (internal), 10
parameters_discrete_ML (internal), 10
pbum (internal), 10
PhiDistr (score), 36
plot.ModuleNetwork (plot.nem), 32
plot.nem, 17, 32
plot.pairwise (plot.nem), 32
plot.score (plot.nem), 32
plot.triples (plot.nem), 32
plotEffects, 33, 33
plotnem (plot.nem), 32
print.ModuleNetwork
  (moduleNetwork), 13
print.nem, 15
print.nem.BN (nem.BN), 14
print.nem.bootstrap
  (nem.bootstrap), 18
print.nem.consensus
  (nem.consensus), 21
print.nem.greedy (nem.greedy), 25
print.nem.greedyMAP (nem.greedyMAP), 26
print.nem.jackknife
  (nem.jackknife), 27
print.pairwise
  (pairwise.posterior), 31
print.score (score), 36
print.triples
  (triples.posterior), 43
prune.graph, 5, 35
qbum (internal), 10
qqbum (internal), 10
rbum (internal), 10
INDEX

remTwoEdges
  (transitive.projections), 42

sampleData (generateNetwork), 7
sampleRndNetwork
  (generateNetwork), 7
SCCgraph, 4
score, 4, 6, 13, 14, 17, 32, 36, 39, 44
score_continuous_Bayesian
  (internal), 10
score_continuous_ML (internal), 10
score_discrete_Bayesian
  (internal), 10
score_discrete_ML (internal), 10
selectEGenes (getRelevantEGenes), 38
subsets, 40

ThreeNeighborhood
  (transitive.projections), 42
transitive.closure, 41, 43
transitive.projections, 42, 42
transitive.reduction, 4, 41, 42
transSubGr
  (transitive.projections), 42
triples.posterior, 17, 43
TwoNeighborhood
  (transitive.projections), 42

VecToMat
  (transitive.projections), 42

which.is.max (internal), 10