Package 'networkBMA'

October 16, 2018

Version 2.20.0

Date 2017--2--06

Author
Chris Fraley, Wm. Chad Young, Ling-Hong Hung, Kaiyuan Shi, Ka Yee Yeung, Adrian Raftery (with contributions from Kenneth Lo)

Title Regression-based network inference using Bayesian Model Averaging

Description An extension of Bayesian Model Averaging (BMA) for network construction using time series gene expression data.
Includes assessment functions and sample test data.

Depends R (>= 2.15.0), stats, utils, BMA, Rcpp (>= 0.10.3), RcppArmadillo (>= 0.3.810.2), RcppEigen (>= 0.3.1.2.1), leaps

LinkingTo Rcpp, RcppArmadillo, RcppEigen, BH

SystemRequirements liblapack-dev

License GPL (>= 2)

Maintainer Ka Yee Yeung <kayee@u.washington.edu>

biocViews GraphsAndNetwork, NetworkInference, GeneExpression, GeneTarget, Network, Bayesian

NeedsCompilation yes

git_url https://git.bioconductor.org/packages/networkBMA

git_branch RELEASE_3_7

git_last_commit b0c0668

git_last_commit_date 2018-04-30

Date/Publication 2018-10-15

R topics documented:

`contabs` *Contingency tables for networks with probabilistic edges.*

Description

Contingency table values relative to known truth for probabilistic networks given thresholds on edge probabilities.

Usage

```
contabs(network, reference, size, thresholds = NULL)
```

Arguments

`network`	A network represented as a data frame in which each row corresponds to an edge with nonzero estimated probability. The first column gives the name of the regulator, the second column gives the name of the regulated gene, and the third column gives the estimated probability for the regulator-gene pair. If the third column is omitted it is assumed that all edges are assigned probability 1.
`reference`	A reference network represented as a two-column data frame in which each row corresponds to an edge, and the columns give the regulator and target genes, respectively. This is the true underlying network for determining the contingency table values.
`size`	The size of the network if all possible edges were included.
`thresholds`	Threshold values on the probability of edges being in the network for which contingency tables are desired. The default is all distinct nonzero probabilities in the network. Thresholds should be specified as probabilities rather than percentages.

Details

Pairs for which edges do not exist in `reference` are not enumerated, because their numbers could be very large. Instead, the `size` parameter is used to account for pairs that are not edges. However, the assumption is that all edges that are in the network but not in the reference are accounted for in `size`.

Care must be taken in specifying `size`. For example, if the reference does not contain self-edges while the network does, and `size` does not count self-edges, then `contabs` will not give the correct number of false negatives.

Value

Outputs a data frame giving the contingency table values for the specified thresholds, with row names being the threshold values. The variables (columns) are of the data frame are as follows:

TP	true positives, or regulator-gene pairs correctly identified as being linked in the reference network,
FN	false negatives, or gene pairs that are linked in the reference network but not in the proposed network,
FP	false positives, or gene pairs that are linked in the proposed network but not in the reference network,
TN	true negatives, or gene pairs that are not linked both networks.

See Also

networkBMA,contabs.netwBMA

Examples

```
data(dream4)

network <- 1

reference <- dream4gold10[[network]]
nGenes <- length(unique(c(reference[,1],reference[,2])))
nPossibleEdges <- nGenes^2

# no self loops in reference (there are none)
any(as.character(reference[,1]) == as.character(reference[,2]))

# restrict reference to edges; first two columns (gene) only
reference <- reference[reference[,3] == 1,1:2]

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10s <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                      nTimePoints = nTimePoints, prior.prob = 0.1,
                      self = TRUE)

# self edges included in contingency table count
size <- nPossibleEdges

contingencyTables <- contabs(network = edges1ts10s, reference = reference,
                             size = size)

matrixFormat(contingencyTables)

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                      nTimePoints = nTimePoints, prior.prob = 0.1,
                      self = FALSE)


# self edges excluded in contingency table count
size <- nPossibleEdges - nGenes

contingencyTables <- contabs(network = edges1ts10, reference = reference,
```

```
                                size = size)

matrixFormat(contingencyTables)
```

`contabs.netwBMA` *Network assessment with incomplete context.*

Description

Contingency table values for the intersection of a probabilistic network of regulator-target gene edges with a reference network with incomplete knowledge.

Usage

```
contabs.netwBMA(network, reference, known = NULL, thresholds = NULL)
```

Arguments

`network`	A network represented as a data frame in which each row corresponds to a directed edge (regulator-gene pair) with nonzero estimated probability. The first column gives the name of the regulator, the second column gives the name of the regulated gene, and the third column gives the estimated probability for the regulator-gene pair. If the third column is omitted it is assumed that all edges are assigned probability 1.
`reference`	A reference network represented as a two-column data frame in which each row corresponds to a regulator-gene pair (or network edge), and the columns give the corresponding regulator and target gene names, respectively. This reference network is used as the standard for determining contingency table entries.
`known`	A 2-column matrix of regulatory relationships that were known (hard-coded) in the modeling process that produced `network`. Each row corresponds to a known regulator-gene pair (edge in the network), and the columns give the corresponding regulator and target gene names, respectively. These known regulatory relationships should all be included in `network` (this is checked), and they may also have regulator-gene pairs in common with `reference`.
`thresholds`	Threshold values on the probability of edges being in the network for which contingency tables are desired. The default is all distinct nonzero probabilities in the network. Nonzero thresholds are treated as inclusive. Thresholds should be specified as probabilities rather than percentages.

Details

For real gene networks, complete information on the true underlying network is not available. `contabs.netwBMA` proceeds by comparing the intersecting subset of the proposed edges with a reference network composed of regulator-genes pairs. Edges that don't exist between any regulator and any target gene in the reference network are used to complete the larger set P of all possible edges for the comparison. Only the subset of network edges that belong to P is considered. There are issues in handling known edges hardcoded in the modeling, for which a detailed explanation is given in the package vignette.

Value

The contingency table for the the intersection of network and reference with edges in known removed.

References

Uncovering regulatory relationships in yeast using networkBMA, networkBMA Bioconductor package vignette.

See Also

networkBMA,contabs.prelim,contabs

Examples

```
## Not run:
data(vignette)

dim(timeSeries)
colnames(timeSeries)
table(timeSeries$replicate)
table(timeSeries$time)

dim(reg.known)
colnames(reg.known)

dim(reg.prob)

edges <- networkBMA(data = timeSeries[,-(1:2)],
                    nTimePoints = length(unique(timeSeries$time)),
                    prior.prob = reg.prob, known = reg.known)

contabs.netwBMA( network = edges, reference = referenceNetwork,
                 known = reg.known)

## End(Not run)
```

---

contabs.prelim *Preliminary calculation for network assessment.*

Description

Computes the quantities to be used for assessment from a proposed and a set of reference edges. For cases where there is incomplete information about the true underlying network.

Usage

```
contabs.prelim(network, reference, known = NULL)
```

Arguments

network	A network represented as a data frame in which each row corresponds to a directed edge (regulator-gene pair) with nonzero estimated probability. The first column gives the name of the regulator, the second column gives the name of the regulated gene, and the third column gives the estimated probability for the regulator-gene pair. If the third column is omitted it is assumed that all edges are assigned probability 1.
reference	A reference network represented as a two-column data frame in which each row corresponds to a regulator-gene pair (or network edge), and the columns give the corresponding regulator and target gene names, respectively. This reference network is used as the standard for determining contingency table entries.
known	A 2-column matrix of regulatory relationships that were known (hard-coded) in the modeling process that produced network. Each row corresponds to a known regulator-gene pair (edge in the network), and the columns give the corresponding regulator and target gene names, respectively. These known regulatory relationships should all be included in network (this is checked), and they may also have regulator-gene pairs in common with reference.

Details

This function accomplishes the preprocessing step from contabs.newtBMA to produce the appropriate input to contabs to produce contingency tables.

Value

A list with the following components:

network	A network represented as a data frame in which each row corresponds to an edge with nonzero estimated probability. This is the input network with edges from the known regulatory relationships and edges outside of the domain defined by the output version of reference removed.
reference	A reference network represented as a two-column data frame in which each row corresponds to an edge, and the columns give the regulator and target genes, respectively. This is the input reference reduced by the known regulatory relationships.
size	The size of the network formed by pairs (r,g) in which r is any regulator g is any gene from the output version of reference, reduced by any known regulatory relationships.

See Also

networkBMA,contabs.netwBMA,contabs

Examples

```
## Not run:
data(vignette)

dim(timeSeries)
colnames(timeSeries)
table(timeSeries$replicate)
table(timeSeries$time)
```

```
dim(reg.known)
colnames(reg.known)

dim(reg.prob)

edges <- networkBMA(data = timeSeries[,-(1:2)],
                    nTimePoints = length(unique(timeSeries$time)),
                    prior.prob = reg.prob, known = reg.known)

prelim <- contabs.prelim( network = edges, reference = referenceNetwork,
                       known = reg.known)
contabs( network = prelim$network, reference = prelim$reference,
         size = prelim$size, thresholds = prelim$thresholds)

contabs.netwBMA( network = edges, reference = referenceNetwork,
                 known = reg.known)

## End(Not run)
```

`dream4` *DREAM 4 (Stolovitsky et al. 2007) 'gold standard' reference network specifications, simulated time series perturbation subsets, and steady state (wild-type) gene expression levels.*

Description

Contains six objects: `dream4gold10`, `dream4gold100`, `dream4ts10`, `dream4ts100`, `dream4wild10`, `dream4wild100`, each in the form of a list of matrices, in which each element corresponds to one of 5 different datasets.

`dream4gold10` and `dream4gold100` are lists of gold standard network specifications for the 10-gene and 100-gene simulated networks. The list elements are three-column data frames, in which the first and second column represent all possible regulator-gene pairs in the network, excluding self edges. The third column is a boolean variable indicating whether or not the genes in the first two columns are a regulatory pair and thus consitute an edge in the network.

`dream4ts10` and `dream4ts100` are lists of time course datasets showing how the simulated network responds to a perturbation and how it relaxes upon removal of the perturbation. For both size 10 and size 100, 5 different datasets are available. For networks of size 10, each of the 5 datasets consists of 5 different time series replicates, and for networks of size 100, each of the 5 datasets consists of 10 time series replicates. Each time series has 21 time points. In both cases, there are 5 list components corresponding to the datasets, each of which is a data frame in which the first two columns give the replicate and time, and the remaining columns give the measured expression levels.

The initial condition always corresponds to a steady-state measurement of the wild-type. At t=0, a perturbation is applied to the network as described below. The first half of the time series (until t=500) shows the response of the network to the perturbation. At t=500, the perturbation is removed (the wild-type network is restored). The second half of the time series (until t=1000) shows how the gene expression levels go back from the perturbed to the wild-type state.

`dream4wild10` and `dream4wild100` are lists giving the steady state (wild-type) expression levels for the genes in each dataset.

Usage

```
data(dream4)
```

Details

The perturbations applied here only affect about a third of all genes, but basal activation of these genes can be strongly increased or decreased. For example, these experiments could correspond to physical or chemical perturbations applied to the cells, which would cause (via regulatory mechanisms not explicitly modeled here) some genes to have an increased or decreased basal activation. The genes that are directly targeted by the perturbation may then cause a change in the expression level of their downstream target genes.
The time series datasets included in this package are a subset of the DREAM 4 data available for the reference networks.

References

Stolovitzky G, Monroe D and Califano A. (2007), Dialogue on Reverse-Engineering Assessment and Methods. Annals of the New York Academy of Sciences,1115(1),1–22.

Stolovitzky G, Califano A, Prill RJ and Saez Rodriguez J.(2009), DREAM: Dialogue for Reverse Engineering Assessments and Methods, http://wiki.c2b2.columbia.edu/dream/

Marbach D, Prill RJ, Schaffter T, Mattiussi C, Floreano D, and Stolovitzky G. Revealing strengths and weaknesses of methods for gene network inference. PNAS, 107(14):6286-6291, 2010.

Marbach D, Schaffter T, Mattiussi C, and Floreano D. Generating Realistic in silico Gene Networks for Performance Assessment of Reverse Engineering Methods. Journal of Computational Biology, 16(2) pp. 229-239, 2009.

Prill RJ, Marbach D, Saez-Rodriguez J, Sorger PK, Alexopoulos LG, Xue X, Clarke ND, Altan-Bonnet G, and Stolovitzky G. Towards a rigorous assessment of systems biology models: the DREAM3 challenges. PLoS ONE, 5(2):e9202, 2010.

See Also

`networkBMA`

`fastBMAcontrol` *Control parameters for* `networkBMA` *when using fastBMA algorithm*

Description

Assigns default control parameters for `networkBMA` when using fastBMA algorithm, and allows setting control parameter values.

Usage

```
fastBMAcontrol(OR = 10000, timeSeries = TRUE, rankOnly = FALSE, noPrune = FALSE,
               edgeMin = 0.5, edgeTol = -1, nThreads = 1,
   selfie = FALSE, showPrune = FALSE, nVars = 0,
   fastgCtrl = fastgControl(), start = -1,
   end = -1, pruneFilterMin = 0, timeout = 0)
```

Arguments

OR	A number specifying the maximum ratio for excluding models in Occam's window.
timeSeries	A logical value indicating whether the input the input data set is a time series data or static data.
rankOnly	A logical value indicating use priors to rank variables but uniform prior otherwise
noPrune	A logical value indicating whether not applying transitive reduction on the output edges or not
edgeMin	Threshold for the posterior probability to be shown
edgeTol	the error tolerance for determining whether an indirect path is as good as a direct path
nThreads	The number of threads used in the parallel computing of fastBMA
selfie	A logical value indicating whether showing self-loop edges or not
showPrune	A logical value indicating whether showing removed edges in transitive reduction or not. Ignored if noPrune is TRUE
nVars	the number of variables analyzed
fastgCtrl	A list of control variables affecting fastBMA computations when using Zellner's g-prior in model likelihhod evaluation. A function called `fastgCtrl` is provided to facilitate this setting, and the default is `fastgCtrl()`.
start	start point of eval subset.
end	end point of eval subset.
pruneFilterMin	minimum posterior prob (0-1) before an edge will be included in the network to be pruned.
timeout	maximum number of seconds for the regression before it stops the search. 0 if not apply.

Value

A list of values for the named control parameters to be passed to `fastBMA`.

References

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2011), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, unpublished manuscript, University of Washington.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

K. Y. Yeung, R. E. Bumgarner and A. E. Raftery (2005). Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21:2394-2402.

J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky (1999). Bayesian Model Averaging: a tutorial, Statistical Science 14(4): 382-417.

L. H. Hong, M. Wu1, A. Lee, W. C. Young, A. E. Raftery and K. Y. Yeung, FastBMA and Transitive Reduction for Gene Network Inference. [in preparation]

See Also

fastgControl, networkBMA

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA(data = dream4ts10[[network]][,-(1:2)],
                         nTimePoints = nTimePoints,
 control=fastBMAcontrol(fastgCtrl=
 fastgControl(optimize=4)))
```

`fastgControl` *Control parameters for using Zellner's g-prior in fastBMA algorithm in* `networkBMA`

Description

Assigns default control parameters for the use of Zellner's g-prior in fastBMA algorithm in `networkBMA`, and allows setting control parameter values.

Usage

```
fastgControl( optimize = 0, g0 = NULL, iterlim = 20 )
```

Arguments

`optimize`	optimize bits - an int value determines how accurate the optimization of g is. 0 means not optimize.
`g0`	An initial value of g to use if optimize is TRUE, or the fixed value to use without optimization.
`iterlim`	If optimize is non-zero, the maximum number of iterations of the optimization algorithm to use. Ignored otherwise.

Value

A list of values for the named control parameters to be passed to `fastBMAcontrol` and `networkBMA` using fastBMA algorithm.

References

A. Zellner (1986), On assessing prior distributions and Bayesian regression analysis with g-prior distributions, Bayesian inference and decision techniques: Essays in Honor of Bruno De Finetti, 6:233-243.

M. Clyde and E.I. George (2004), Model Uncertainty, Statistical Science, 81-94.

L. H. Hong, M. Wu1, A. Lee, W. C. Young, A. E. Raftery and K. Y. Yeung, FastBMA and Transitive Reduction for Gene Network Inference. [in preparation]

See Also

fastBMAcontrol, networkBMA

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA(data = dream4ts10[[network]][,-(1:2)],
                         nTimePoints = nTimePoints,
 control=fastBMAcontrol(fastgCtrl=
 fastgControl(optimize=4)))
```

---

gControl *Control parameters for using Zellner's g-prior in* ScanBMA

Description

Assigns default control parameters for the use of Zellner's g-prior in ScanBMA, and allows setting control parameter values.

Usage

```
gControl( optimize = TRUE, optMethod = "perTarget", g0 = NULL,
          iterlim = 100, epsilon = 0.1 )
```

Arguments

optimize	A logical value indicating whether to optimze g using an iterative EM algorithm or use a fixed value of g.
optMethod	A character string indicating how to optimize g. Currently, only perTarget is supported, indicating that g should be optimized individually for each target.
g0	An initial value of g to use if optimize is TRUE, or the fixed value to use without optimization.
iterlim	If optimize is TRUE, the maximum number of iterations of the EM algorithm to use. Ignored otherwise.
epsilon	If optimize is TRUE, the precision with which to find g using the EM algorithm. Ignored otherwise.

Value

A list of values for the named control parameters to be passed to ScanBMAcontrol and ScanBMA.

References

A. Zellner (1986), On assessing prior distributions and Bayesian regression analysis with g-prior distributions, Bayesian inference and decision techniques: Essays in Honor of Bruno De Finetti, 6:233-243.

M. Clyde and E.I. George (2004), Model Uncertainty, Statistical Science, 81-94.

See Also

ScanBMAcontrol, ScanBMA, networkBMA

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints,
                          control = ScanBMAcontrol(gCtrl =
                          gControl(optimize = TRUE)) )
```

---

iBMAcontrolLM *Control parameters for* iterateBMAlm

Description

Assigns default control parameters for iterateBMAlm, and allows setting control parameter values.

Usage

```
iBMAcontrolLM( OR = 20, nbest = 10, maxNvar = 30, thresProbne0 = 1,
               keepModels = FALSE, maxIter = 200000)
```

Arguments

OR	A number specifying the maximum ratio for excluding models in Occam's window.
nbest	A positive integer specifying the number of models of each size to be considered by leaps-and-bounds in determining the model space for Bayesian Model Averaging. The default value is 10.
maxNvar	A positive integer specifying the maximum number of variables (excluding the intercept) used in each iteration of BMA. The default value is 30.
thresProbne0	Threshold (in percent) for the posterior probability that each variable is has a non-zero coefficient (in percent). Variables with posterior probability less than thresProbne0 are removed in future BMA iterations. The default value is 1 percent.
keepModels	A logical value indicating whether or not to keep the BMA models from all of the iterations and apply Occam's window using OR at the end, or to apply Occam's window in all BMA iterations and return the final model. The default is not to keep the models. Setting the argument to TRUE requires more memory and may slow the computation as a result.
maxIter	A positive integer giving a limit on the number of iterations of iterateBMAlm. The default value is 20000. iterateBMAlm will terminate in fewer than maxIter iterations if the iterative BMA modeling process has seen all available variables.

Value

A list of values for the named control parameters to be passed to a version of the function `bicreg` from the `BMA` package that has been modified to handle prior probabilities.

References

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2012), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, BMC Systems Biology, 6:101.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

K. Y. Yeung (with contributions from A. E. Raftery and I. Painter), iterativeBMA: The Iterative Bayesian Model Averaging (BMA) algorithm, Bioconductor R package, version 1.8.0 posted in 2009.

K. Y. Yeung, R. E. Bumgarner and A. E. Raftery (2005). Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21:2394-2402.

A. E. Raftery, J. A. Hoeting, C. T. Volinsky, I. Painter and K. Y. Yeung (2005), BMA: Bayesian Model Averaging, Comnprehensive R Archhive Network (CRAN), package version 3.15.1 posted in 2012.

J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky (1999). Bayesian Model Averaging: a tutorial, Statistical Science 14(4): 382-417.

See Also

`iterateBMAlm`, `networkBMA` `do.call`

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints,
                          control = iBMAcontrolLM(thresProbne0 = 1))
```

`iterateBMAlm` *Iterative BMA for linear modeling with prior variable probabilities.*

Description

An iterative version of Bayesian Model Averaging (BMA) for linear models with many variables. Incorporates prior probabilities for inclusion of variables in models.

Usage

```
iterateBMAlm( x, y, prior.prob = NULL, control = iBMAcontrolLM(),
              verbose = FALSE)
```

Arguments

x	A matrix of real-valued predictor variables. Rows correspond to observations and columns to variables.
y	A real-valued response vector.
prior.prob	An optional vector of prior probabilities for each predictor variable belonging to a linear model for the data. If not specified, predictor variables are assumed to have equal prior probability.
control	A list of values controling the underlying algorithm. The default is given by iBMAcontrolLM().
verbose	A logical variable indicating whether or not the details of the method's progress shoul be printed during computation. The default value is FALSE.

Details

iterateBMAlm is intended for datasets that have more variables (e.g. gene expression values) than observations (e.g. subjects). There is currently no mechanism for handling factor variables in iterateBMAlm, as there is in the underlying function bicreg in the BMA package. However factors can be encoded by users and included with other variables as input.

Value

A list with the following components, similar to the output of function bicreg in the BMA package:

bic	values of BIC for the models
postprob	the posterior probabilities of the models selected
priorprob	the prior probabilities of the variables in the models
namesx	the names of the variables
label	labels identifying the models selected
r2	R2 values for the models
size	the number of independent variables in each of the models
which	a logical matrix with one row per model and one column per variable indicating whether that variable is in the model
probne0	the posterior probability that each variable is non-zero (in percent)
postmean	the posterior mean of each coefficient (from model averaging)
condpostmean	the posterior mean of each coefficient conditional on the variable being included in the model
condpostsd	the posterior standard deviation of each coefficient conditional on the variable being included in the model
ols	matrix with one row per model and one column per variable giving the OLS estimate of each coefficient for each model
mle	the same as ols
n.models	the number of models
n.vars	the number of variables

References

K. Y. Yeung, R. E. Bumgarner and A. E. Raftery (2005), Bayesian model averaging: development of an improved multi-class, gene selection and classification tool for microarray data, Bioinformatics 21(10) 2394-2402.

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2012), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, BMC Systems Biology, 6:101.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

See Also

iBMAcontrolLM, varord, bicreg

Examples

```
data(dream4)

network <- 1

Time <- as.numeric(dream4ts100[[network]]$time)

xIndex <- which(Time != max(Time))
yIndex <- which(Time != min(Time))

gene <- "G1"

x <- dream4ts100[[network]][xIndex,-(1:2)]
y <- dream4ts100[[network]][yIndex,gene]

nvar <- 50
ord <- varord( x, y, ordering = "bic1")[1:nvar]

result <- iterateBMAlm( x = x[,ord], y = y)
```

`networkBMA` *Gene network inference from time series data via BMA.*

Description

Estimates a probabilistic network of regulator-gene pairs from time series data via ScanBMA or iterative BMA.

Usage

```
networkBMA(data, nTimePoints, prior.prob = NULL, known = NULL,
           ordering = "bic1+prior", nvar = NULL, self = TRUE,
           maxreg = NULL,
           control = ScanBMAcontrol(),
           diff0 = TRUE, diff100 = TRUE,
           verbose = FALSE)
```

Arguments

data	A matrix whose columns correspond to variables or genes and whose rows correspond to the observations at different time points. The row names should be the gene names, to used in conjunction with prob and known as appropriate.
nTimePoints	The number of time points at which expression measurements are available. The number of columns in data should be a multiple of nTimePoints, which could be greater than 1 if there are replicates.
prior.prob	If included as input, either a single positive fraction representing the probability of an regulator-gene pair in the network, or else a matrix in which the (i,j) entry is the estimated prior probability that gene i regulates gene j. The default value is NULL, which implies that no prior information will be used in modeling the network.
known	An optional 2-column matrix of known (hard-coded) regulatory relationships. The first column gives the name of the regulator, and the second column gives the name of the target gene. The gene names should be consistent with the data.
ordering	A character string indicating the ordering to be used for the genes or variables, referring to the options for ordering in function varord. The default is option "bic1+prior" for varord. If a prior is provided, genes with prob=1 are always included first in modeling regardless of the specified ordering, while genes with prior = 0 are excluded.
nvar	The number of top-ranked (see ordering) genes to be considered in the modeling. The default is determined by the length of ordering if it is greater than 1, otherwise it is min(nrow(data),ncol(data)-1). The maximum number of genes for which measurements are available is nrow(data). If fastBMA algorithm is choosen, this parameter ignored, nVars in fastBMAcontrol used instead
self	A logical variable indicating whether or not to allow self edges in modeling. The default is to allow self edges.
maxreg	An optional estimate of the maximum number of regulators for any gene in the network. If provided, this is used to help reduce the amount of memory used for the computations.
control	A list of control variables affecting the BMA computations. The functions ScanBMAcontrol for optimize="ScanBMA" and iBMAcontrolLM for optimize="iBMA" and fastBMAcontrol for optimize="fastBMA"are provided to faciltate this setting, and the default is ScanBMAcontrol().
diff0	A logical variable indicating whether to differentiate between edges with posterior probability of 0. This includes regulators not included from when nvar is less than the total number of possible regulators. If known is not NULL, then diff0 must be FALSE. If fastBMA algorithm used, then diff0 must be FALSE.
diff100	A logical variable indicating whether to differentiate between edges with posterior probability of 1.0. If known is not NULL, then diff100 must be FALSE. If fastBMA algorithm used, then diff100 must be FALSE.
verbose	A logical variable indicating whether or not a detailed information should be output as the computation progresses. The default value is FALSE.

Details

networkBMA is intended for time-series data in which there are more variables (gene expression values) than observations (experiments). For each gene, a linear model is fit to the expression data for all genes at a particular time point to predict the expression of a particular gene at the next

time point. BMA is used to fit the linear model to identify the candidate regulators (variables) in the model. The inferred network consists of candidate regulators and their corresponding posterior probabilities for each gene.
It is assumed that data is available for all replicates at the same set of time points.

Value

A network represented as a data frame in which each row corresponds to a directed edge for which the probability is estimated to be nonzero. The first column gives the name of the regulator, the second column gives the name of the regulated gene, and the third column gives the estimated probability for the regulator-gene pair. Rows are ordered by decreasing probability estimate.
The `summary` function gives the number of inferred edges at posterior probabilities 0, .5, .75, .90, .95 and 1.0

References

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2012), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, BMC Systems Biology, 6:101.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

K. Y. Yeung, A. E. Raftery and C. Fraley (2012), Uncovering regulatory relationships in yeast using `networkBMA`, `networkBMA` Bioconductor package vignette.

L. H. Hong, M. Wu1, A. Lee, W. C. Young, A. E. Raftery and K. Y. Yeung, FastBMA and Transitive Reduction for Gene Network Inference. [in preparation]

See Also

`summary.networkBMA`, `varord`, `ScanBMAcontrol`, `iBMAcontrolLM`, `contabs`

Examples

```
data(dream4)

# there are a total of 5 datasets (networks) in the dream4ts10 data
network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints, prior.prob = 0.01)

summary(edges1ts10)

edges1ts10.fastBMA <- networkBMA(data=dream4ts10[[network]][,-(1:2)], nTimePoints = nTimePoints,
                         control=fastBMAcontrol(optimize <- 4))
summary(edges1ts10.fastBMA)
```

roc *Receiver Operating Characteristic and Precision-Recall Curves*

Description

Computes the Receiver Operating Characteristic (ROC) or precision-recall curves from contingencies tables (confusion matrices) and their associated probabilities, and optionally plots them.

Usage

```
roc( contabs, plotit = TRUE)
prc( contabs, plotit = TRUE, ymax = max(y))
```

Arguments

contabs	A data frame representing a contingency tables (confusion matrices) for a binary classification experiment. The column names should include TP (for true positive), FN (for false negative), FP (for false positive), TN (for true negative); these can be given in any order.
plotit	A logical variable indicating whether or not the ROC curve should be plotted.
ymax	Upper vertical axis value for plotting precision-recall curves with prc. The default is max(y), (the maximum value for precision derived from the contingency table). ymax should be no larger than 1.

Details

The estimated area may be inaccurate when the sector covered by the contingency tables is small, and should be used with caution in such cases.

Value

A vector with the following named components:

area	The area under the extended curve (covered by the black solid and dotted lines).
sector	The estimated area under the sector (below the curve and within the red lines) covered by the contingency table.
width	The width of the sector (within the red vertical lines) covered by the contingency table.

As a side-effect, the ROC (roc) or precision-recall curve (prc) is plotted if plotit = TRUE. The red vertical lines highlight the sector covered by the contingency table. The black lines approximate the curve. They are dotted beyond the endpoints of the sector, where there is no information from the contingency table. The diagonal blue line in the ROC curve indicates the line between (0,0) and (1,1).

References

J. Davis and M. Goadrich, The relationship between Precision-Recall and ROC curves, manuscript, Department of Computer Science, University of Wisconsin.

See Also

contabs

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                      nTimePoints = nTimePoints, prior.prob = 0.1)

# check for self loops in estimated network
selfN <- any(as.character(edges1ts10[,1]) == as.character(edges1ts10[,2]))
selfN

reference <- dream4gold10[[network]]

# check for self loops in reference (there are none)
selfR <- any(as.character(reference[,1]) == as.character(reference[,2]))
selfR

# restrict reference to edges; first two columns (gene) only
reference <- reference[reference[,3] == 1,1:2]

contingencyTables <- contabs(network = edges1ts10, reference = reference,
                             size = 100)

roc(contingencyTables)
prc(contingencyTables)
```

ScanBMA *Bayesian Model Averaging for linear regression models.*

Description

Bayesian Model Averaging accounts for the model uncertainty inherent in the variable selection problem by averaging over the best models in the model class according to approximate posterior model probability.

Usage

```
ScanBMA(x, y, prior.prob = NULL,
        control = ScanBMAcontrol(), verbose = FALSE)
```

Arguments

x	A matrix of independent variables.
y	A vector of values for the dependent variable.

prior.prob	If included as input, either a single positive fraction representing the probability of an independent variable being present in the true model, or else a vector assigning an estimated prior probability for each independent variable individually. The default value is NULL, which implies that no prior information will be used.
control	A list of control variables affecting the ScanBMA computations. The function ScanBMAcontrol is provided to faciltate this setting, and the default is ScanBMAcontrol().
verbose	A logical variable indicating whether or not a detailed information should be output as the computation progresses. The default value is FALSE.

Details

Bayesian Model Averaging accounts for the model uncertainty inherent in the variable selection problem by averaging over the best models in the model class according to approximate posterior model probability. ScanBMA is an algorithm for searching the model space efficiently when a large number of independent variables are present.

Value

Returns an object of classbicreg (see the BMA package). In addition, it adds nmodelschecked, which gives the number of models looked at in the ScanBMA model search, and g, which gives the final value of g used if Zellner's g-prior was used to evaluate model likelihood.

References

Raftery, Adrian E. (1995). Bayesian model selection in social research (with Discussion). Sociological Methodology 1995 (Peter V. Marsden, ed.), pp. 111-196, Cambridge, Mass.: Blackwells.

See Also

networkBMA, ScanBMAcontrol, gControl

Examples

```
data(dream4)

# there are a total of 5 datasets (networks) in the dream4ts10 data
network <- 1

scanBMA.res <- ScanBMA( x = dream4ts10[[network]][,-(1:2)],
                        y = dream4ts10[[network]][,3],
                        prior.prob = 0.01)

summary(scanBMA.res)
```

ScanBMAcontrol *Control parameters for* ScanBMA

Description

Assigns default control parameters for ScanBMA, and allows setting control parameter values.

Usage

```
ScanBMAcontrol( OR = 100, useg = TRUE, gCtrl = gControl(), thresProbne0 = 1 )
```

Arguments

OR	A number specifying the maximum ratio for excluding models in Occam's window.
useg	A logical value indicating whether to use Zellner's g-prior in model likelihood evaluation. If set to FALSE, ScanBMA will use BIC to approximate the likelihood.
gCtrl	A list of control variables affecting ScanBMA computations when using Zellner's g-prior in model likelihhod evaluation. A function called gControl is provided to facilitate this setting, and the default is gControl().
thresProbne0	Threshold (in percent) for the posterior probability that each variable is has a non-zero coefficient (in percent). Variables with posterior probability less than thresProbne0 are removed in future BMA iterations. The default value is 1 percent.

Value

A list of values for the named control parameters to be passed to ScanBMA.

References

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2011), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, unpublished manuscript, University of Washington.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

K. Y. Yeung, R. E. Bumgarner and A. E. Raftery (2005). Bayesian Model Averaging: Development of an improved multi-class, gene selection and classification tool for microarray data. Bioinformatics 21:2394-2402.

J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky (1999). Bayesian Model Averaging: a tutorial, Statistical Science 14(4): 382-417.

See Also

gControl, ScanBMA, networkBMA

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints,
                          control = ScanBMAcontrol(thresProbne0 = 1) )
```

`scores` *Scores for assessment from contingency tables.*

Description

Gives scores for assessment and other values associated with contingency tables for network inference.

Usage

```
scores( contabs, what = c("TP", "FN", "FP", "TN",
       "TPR", "TNR", "FPR", "FDR", "PPV", "NPV",
       "sensitivity", "specificity", "precision", "recall",
       "F1", "MCC", "ACC", "expected", "O/E"))
```

Arguments

`contabs` A data frame representing a contingency tables (confusion matrices) for a binary classification experiment. The column names should include `TP` (for true positive), `FN` (for false negative) `FP` (for false positive), `TN` (for true negative); these can be given in any order.

`what` A character string specifying one or more desired output quantities from among:

"TP" Number of edges that are correctly identified as such (true positives).

"FP" Number of unlinked pairs that are incorrectly identified as edges (false positives).

"TN" Number of unlinked pairs that are correctly identified as such (true negatives).

"FN" Number of edges that are not identified as such (false negatives).

"TPR","sensitivity","recall" True positive rate, sensitivity, or recall. `TP/(# edges)`.

"FPR" False positive or false alarm rate. `FP/(# edges)`.

"TNR","specificity" True negative rate or specificity. `TN/(# unlinked pairs)`.

"FDR" False discovery rate. `FP/(# estimated links)`.

"PPV","precision" Positive predictive value or precision. `TP/(# estimated links)`.

"NPV" Negative predictive value. `TN/(# estimated unlinked pairs)`.

"F1" F1 score, F-measure, balanced F-score. Harmonic mean of precision and recall.

"MCC" Matthews correlation coefficient.

"ACC" Accuracy. (TP + TN)/(# edges + # unlinked pairs).

"expected" Expected number of links under random assortment.

"O/E" Ratio of estimated (observed) number of edges to expected number of links under random assortment. TP/"expected".

Value

A data frame in which the variables are the desired quantities derived the specified contingency tables. The rows correspond to the rows of the contingency tables supplied as input (contabs).

See Also

contabs, roc, prc

Examples

```
data(dream4)

network <- 1

reference <- dream4gold10[[network]]
nGenes <- length(unique(c(reference[,1],reference[,2])))
nPossibleEdges <- nGenes^2
reference <- reference[reference[,3] == 1,1:2]

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints, prior.prob = 0.1,
                          self = FALSE)

size <- nPossibleEdges - nGenes

contingencyTables <- contabs(network = edges1ts10, reference = reference,
                             size = size)

scores(contingencyTables, what = c("sensitivity", "specificity", "FDR"))
```

---

summary.networkBMA *Summarizes a* networkBMA *object.*

Description

Counts the number of edges in a networkBMA object that are at or above specified probability thresholds.

Usage

```
## S3 method for class 'networkBMA'
summary(object, thresholds = c(0,.5,.75,.90,.95,1), ...)
```

Arguments

object	A networkBMA object.
thresholds	Threshold probabilities. The number of edges at or above these values are included in the output.
...	Not used. For generic/method consistency.

Value

A vector giving the number of edges in the object that have probability at or above the values given in thresholds.

See Also

networkBMA

Examples

```
data(dream4)

# there are a total of 5 datasets (networks) in the dream4ts100 data
network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints, prior.prob = 0.01,
                          nvar = 50)

summary(edges1ts10)
```

---

varord *Variable orderings for linear regression.*

Description

Gives variable orderings for linear regression in high-dimensional data by various approaches.

Usage

```
varord(x, y, prior.prob = NULL, ordering = c("bic1","prior","bic1+prior"))
```

Arguments

x	A matrix whose columns correspond to variables or genes and whose rows correspond to the observations.
y	A vector of response values for fitting a linear model to a subset of the variables in x.
prior.prob	A vector of prior probabilities corresponding to the variables (genes).
ordering	A character string specifying the method to be used for the ordering the variables.

"bic1" Variables are ordered in by the BIC for the univariate linear model with that variable as predictor. (Yeung et~al. 2011)

"prior" The variables are ordered according to the log odds of their prior values. (Lo et~al. 2011)

"bic1+prior" The variables are ordered according to sum of the univarite BIC values and the log odds of their prior values.

Value

A vector of integers corresponding to the variable indexes in the specified order.

References

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2012), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, BMC Systems Biology, 6:101.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

See Also

networkBMA, iterateBMAlm

Examples

```
data(dream4)

network <- 1

Time <- as.numeric(dream4ts100[[network]]$time)

xIndex <- which(Time != max(Time))
yIndex <- which(Time != min(Time))

gene <- "G1"

x <- dream4ts100[[network]][xIndex,-(1:2)]
y <- dream4ts100[[network]][yIndex,gene]

nvar <- 50
ord <- varord( x, y, ordering = "bic1")[1:nvar]

## Not run:
result <- iterateBMAlm( x = x[,ord], y = y)

## End(Not run)
```

vignette *The subsets of the yeast-rapamycin time-series data from Yeung et al. (2011) and Lo et al. (2011), and of the static yeast gene-expression data from Brem et al. (2002, 2005), that are used in the* networkBMA *package vignette.*

Description

Contains four data objects used to illustrate the networkBMA package in the accompanying vignette.

brem.data An 85 by 111 subset of the data used for network inference in yeast in Brem et~al. (2002) and Brem and Kruglyak (2005). The rows correspond to genes and the columns to experiments. Provided courtesy of Rachel Brem.

referencePairs A 2-column data frame giving 287 regulator-gene pairs among the selected set of genes reported from the literature. The known regulatory relationships are not necessarily included in the reference network.

reg.known A 2-column matrix of known (hard-coded) regulatory relationships among the 100 gene subset. The first column gives the name of the regulator, and the second column gives the name of the target gene. The gene names should be consistent with the data.

reg.prob A 100 by 100 matrix in which the (i,j) entry is the estimated prior probability that gene i regulates gene j (from Lo et al. 2011).

timeSeries A data frame in which the first two columns are factors identifying the replicate and time (in minutes) after drug perturbation, and the remaining 100 columns are the expression measurements for a subset of 100 genes from the yeast-rapamycin experiment described in Yeung et al. (2011).

Usage

```
data(vignette)
```

References

R. B. Brem and G. Yvert and R. Clinton and L. Kruglyak (2002), Genetic dissection of transcriptional regulation in budding yeast, Science 296(5568):752-755.

R. B. Brem and L. Kruglyak (2005), The landscape of genetic complexity across 5,700 gene expression traits in yeast, Proceedings of the National Academy of Sciences 102(5):1572-1577.

K. Lo, A. E. Raftery, K. M. Dombek, J. Zhu, E. E. Schadt, R. E. Bumgarner and K. Y. Yeung (2012), Integrating External Biological Knowledge in the Construction of Regulatory Networks from Time-series Expression Data, BMC Systems Biology, 6:101.

K. Y. Yeung, K. M. Dombek, K. Lo, J. E. Mittler, J. Zhu, E. E. Schadt, R. E. Bumgarner and A. E. Raftery (2011), Construction of regulatory networks using expression time-series data of a genotyped population, Proceedings of the National Academy of Sciences, 108(48):19436-41.

See Also

iterateBMAlm

Examples

```
data(vignette)

gene <- "YNL037C"
variables <- which(rownames(brem.data) != gene)
control <- iBMAcontrolLM(OR = 50, nbest = 20, thresProbne0 = 5)
iBMAmodel.YNL037C <- iterateBMAlm( x = t(brem.data[variables,]),
                          y = unlist(brem.data[gene,]), control = control)
```

`writeEdges` *Output network edges to text in Cytoscape-readable format.*

Description

Outputs a data frame of probabilistic network edges to a text file suitable as input to Cytoscape for visualization.

Usage

```
writeEdges( network, threshhold = .5, fileName = "edges.txt")
```

Arguments

`network`	A network represented as a data frame in which each row corresponds to an edge for which the probability is estimated to be nonzero. The first column gives the name of the regulator, the second column gives the name of the regulated gene, and the third column gives the estimated probability for the regulator-gene pair.
`threshhold`	A threshhold on the probability that the variables in `x`. The default value is .5
`fileName`	A character string given the desired name of the output text file. The default is `"edges.txt"`.

Value

The data frame of edges with those having probabilities below the specified threshhold removed. As a side effect, this data from is output to a text file named `fileName` in a format suitable for use with Cytoscape for visualization (see `www.cytoscape.org`).

References

Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T (2003), Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Research 13:2498-504

See Also

`networkBMA`

Examples

```
data(dream4)

network <- 1

nTimePoints <- length(unique(dream4ts10[[network]]$time))

edges1ts10 <- networkBMA( data = dream4ts10[[network]][,-(1:2)],
                          nTimePoints = nTimePoints, prior.prob = 0.1)

writeEdges(edges1ts10)
```

Index