Package ‘mapggm’

October 11, 2015

Type Package

Title Multi-attribute network construction and perturbation detection

Version 0.0.0

Date 2015-09-30

Author Paula J Griffin

Maintainer Paula J Griffin <paulajgriffin@gmail.com>

Description
Companion package to 'Detection of multiple perturbations in multi-omics biological networks'

License Apache License

Imports Matrix, mvtnorm, igraph

R topics documented:

`blockNorms` *Get Frobenius norms of submatrices with optional weights*

Description

Given a square matrix `M` and a vector `id` that distinguishes sections of `M`, return a matrix with the Frobenius norm of the specified submatrices. Optionally, provide a square weight matrix `W` with number of rows and columns equal to the unique entries in id that multiplies these norms.

Usage

```
blockNorms(M, id, W = NULL)
```

Arguments

`M`	square matrix of interest
`id`	vector grouping elements of M
`W`	optional weights (square matrix, dimensions equal to unique id

Value

matrix of Frobenius norms of submatrices

`getAIC` *Get AIC of a given network configuration*

Description

In a zero-mean Gaussian graphical model, calculate the Akaike information criterion (AIC) of a model with the proposed precision matrix Omega.

Usage

```
getAIC(S, n, Omega, id)
```

Arguments

`S`	sample covariance matrix
`n`	number of samples
`Omega`	estimated precision
`id`	vector grouping elements of S, Omega

Value

Akaike information criterion for model implied by Omega

`getBIC` *Get BIC of a given network configuration*

Description

In a zero-mean Gaussian graphical model, calculate the Bayesian information criterion (BIC) of a model with the proposed precision matrix Omega.

Usage

```
getBIC(S, n, Omega, id)
```

Arguments

`S`	sample covariance matrix
`n`	number of samples
`Omega`	estimated precision
`id`	vector grouping elements of S, Omega

Value

Bayesian information criterion for model implied by Omega

`getEBIC` *Get extended BIC of a given network configuration*

Description

In a zero-mean Gaussian graphical model, calculate the extended Bayesian information criterion (EBIC; Chen and Chen, 2008) of a model with the proposed precision matrix Omega. The parameter gamma controls the weight given towards the component of EBIC related to model size.

Usage

```
getEBIC(S, n, Omega, id, gamma = 0.5)
```

Arguments

`S`	sample covariance matrix
`n`	number of samples
`Omega`	estimated precision
`id`	vector grouping elements of S, Omega
`gamma`	EBIC parameter (default 0.5)

Value

extended Bayesian information criterion for model implied by Omega

References

Chen, J. and Chen, Z. (2008). Extended Bayesian information criteria for model selection with large model spaces. Biometrika 95, 759-771.

`getHammingDist` *Get Hamming distance of a given network configuration (truth required)*

Description

Calculate the Hamming distance of a given network construction (as provided by Omega), compared to a true network.

Usage

```
getHammingDist(Omega, Theta, id)
```

Arguments

`Omega`	estimated precision
`Theta`	true network graph of joint nodes
`id`	vector grouping elements of Omega

Value

Hamming distance between Theta and graph implied by Omega

`getLambdaRange` *Generate lambdas to try*

Description

Generate penalty parameters spanning a range of 1 to `length(unique(id))` submatrix blocks. This function is provided to establish a range of reasonable penalty parameters for optimization according to the algorithm of Kolar et al (2014).

Usage

```
getLambdaRange(S, id, length.out = 10)
```

Arguments

`S`	sample covariance
`id`	vector of node identifiers
`length.out`	number of lambda parameters to return

Value

vector of `length.out` equally spaced lambdas

References

Kolar, M., Liu, H., and Xing, E. P. (2014). Graph estimation from multi-attribute data. The Journal of Machine Learning Research 15, 1713-1750.

Examples

```
Y <- matrix(rnorm(120), nrow=20, ncol=6)
S <- crossprod(Y)
id <- rep(1:3, each=2)
getLambdaRange(S, id, 5)
```

`multiAttEstimate` *Wrapper function for multi-attribute network estimation via lasso*

Description

Performs estimation of a zero-mean multi-attribute Gaussian graphical model as described by Kolar et al (2014) when `mode=1`.

Usage

```
multiAttEstimate(S, n, id, lambda, W = NULL, mode = 1, update = 100,
  max.gap = 0.5, max.iter = 100, min.t = .Machine$double.eps)
```

Arguments

`S`	sample covariance matrix
`n`	number of samples
`id`	vector assigning variables to nodes
`lambda`	penalty tuning parameter
`W`	optional weight matrix
`mode`	estimation mode. Can be 1 (multiattribute; default), 2 (unstructured together), or 3 (separately; only valid for 2-attribute data)
`update`	how often to print updates in optimization
`max.gap`	maximum allowable primal/dual gap
`max.iter`	maximum number of iterations to optimize (overrides max.gap)
`min.t`	minimum step size (overrides max.gap, max.iter)

Details

If `mode=2`, the `id` argument is essentially disregarded, and all rows/columns in `S` are treated as if they represent individual nodes. If `mode=3`, the optimization of `mode=2` is performed separately for each attribute type, and a matrix with 0 on all cross-attribute entries is returned.

Value

list of precision, covariance, optimization status, lambda, and number components

References

Kolar, M., Liu, H., and Xing, E. P. (2014). Graph estimation from multi-attribute data. The Journal of Machine Learning Research 15, 1713-1750.

Examples

```
library(mvtnorm)
id <- rep(1:3, each=2)
Omega <- matrix(0, nrow=6, ncol=6)
Omega[1:4,1:4] <- 1
diag(Omega) <- 6
Sigma <- solve(Omega)
n <- 1000
Y <- rmvnorm(n, sigma=Sigma)
S <- crossprod(Y)

lambdas <- getLambdaRange(S, id, 10)
result <- multiAttEstimate(S, n, id, lambdas[1])
```

`multiAttLasso` *Multi-attribute network estimation*

Description

Estimates block precisions and covariances for a multi-attribute network based on a Gaussian graphical model with zero mean vector. This is according to the method described by Kolar et al (2015).

Usage

```
multiAttLasso(S, n, id, lambda, W = NULL, update = 100, max.gap = 0.5,
  max.iter = 100, min.t = .Machine$double.eps)
```

Arguments

`S`	sample covariance matrix
`n`	number of samples
`id`	vector of node identifiers
`lambda`	tuning parameter for penalty
`W`	optional penalty weight matrix (same dimensions as S)
`update`	how often to print updates
`max.gap`	maximum allowable primal-dual gap
`max.iter`	maximum number of iterations to complete (overriedes max.gap)
`min.t`	minimum step size (overrides max.gap, max.iter)

Value

list of precision, covariance, optimization status, lambda, and number components

References

Kolar, M., Liu, H., and Xing, E. P. (2014). Graph estimation from multi-attribute data. The Journal of Machine Learning Research 15, 1713-1750.

`multiAttSelect` — *Wrapper function for multi-attribute network estimation plus selection via lasso*

Description

Performs estimation of a zero-mean multi-attribute Gaussian graphical model as described by Kolar et al (2014) when `mode=1` with a selection procedure to determine optimal `lambda`. Selection may be performed according to extended BIC, BIC, AIC, or Hamming distance to true graph. The true graph `Theta` is only required if Hamming distance is being used for the selection procedure.

Usage

```
multiAttSelect(S, n, id, lambda.range, W = NULL, mode = 1, update = 100,
  max.gap = 0.5, max.iter = 100, min.t = .Machine$double.eps,
  method = "BIC", Theta.true = NULL, plot = NULL, gamma = 0.5)
```

Arguments

`S`	sample covariance matrix
`n`	number of samples
`id`	vector assigning variables to nodes
`lambda.range`	vector of lambdas to try
`W`	optional weight matrix
`mode`	estimation mode. Can be 1 (multiattribute; default), 2 (unstructured together), or 3 (separately; only valid for 2-attribute data)
`update`	how often to print updates in optimization
`max.gap`	maximum allowable primal/dual gap
`max.iter`	maximum number of iterations to optimize (overrides max.gap)
`min.t`	minimum step size (overrides max.gap, max.iter
`method`	how to select the best model (`EBIC`, `BIC`, `AIC`, or `hamming`)
`Theta.true`	true underlying graph (`hamming` method only)
`plot`	boolean, whether or not to make diagnostic plot
`gamma`	gamma parameter for EBIC method (default 0.5)

Details

If `mode=2`, the `id` argument is essentially disregarded, and all rows/columns in `S` are treated as if they represent individual nodes. If `mode=3`, the optimization of `mode=2` is performed separately for each attribute type, and a matrix with 0 on all cross-attribute entries is returned.

Value

list of precision, covariance, optimization status, lambda, and number components

References

Kolar, M., Liu, H., and Xing, E. P. (2014). Graph estimation from multi-attribute data. The Journal of Machine Learning Research 15, 1713-1750.

Chen, J. and Chen, Z. (2008). Extended Bayesian information criteria for model selection with large model spaces. Biometrika 95, 759-771.

Examples

```
library(mvtnorm)
id <- rep(1:3, each=2)
Omega <- matrix(0, nrow=6, ncol=6)
Omega[1:4,1:4] <- -1
diag(Omega) <- 6
Sigma <- solve(Omega)
n <- 1000
Y <- rmvnorm(n, sigma=Sigma)
S <- crossprod(Y)

lambdas <- getLambdaRange(S, id, 3)
result <- multiAttSelect(S, n, id, lambdas)
```

`perturbNodeTests` *Get node-wise test statistics*

Description

Given a vector `id` indicating node membership, test for perturbations at each node as described by Griffin et al. This function performs either the single-target test (`sequential=FALSE`, default) or the multi-target adjusted procedure (`sequential=TRUE`)

Usage

```
perturbNodeTests(Y, Omega, Sigma, id, sequential = FALSE,
  return.value = "stat")
```

Arguments

`Y`	matrix or data frame (rows=subjects, columns=variables)
`Omega`	precision matrix
`Sigma`	covariance matrix
`id`	vector mapping variables to nodes
`sequential`	boolean, whether or not to perform sequential adjustments
`return.value`	return statistics (`stat`) or p-values (`pval`)

Value

test statistics or p-values for perturbations at each individual node

References

Griffin, P.J., Johnson, W.E., and Kolaczyk, E.K. Detection of multiple perturbations in multi-omics biological networks (under review)

Examples

```
library(mvtnorm)
Omega <- matrix(0, nrow=6, ncol=6)
Omega[1:4,1:4] <- -1
diag(Omega) <- 6
Sigma <- solve(Omega)
mu <- c(1,1,0,0,0,0)
Y <- rmvnorm(20, mean=Sigma %*% mu, sigma=Sigma)
id <- c(1,1,2,2,3,3)
perturbNodeTests(Y, Omega, Sigma, id, sequential=FALSE) # unadjusted
perturbNodeTests(Y, Omega, Sigma, id, sequential=TRUE) # sequential
```

`perturbTests` *Test for perturbations at a series of locations*

Description

Given a matrix of possible perturbation locations, perform the single-target (non-sequential) testing procedure described by Griffin et al. Return either test statistics or p-values.

Usage

```
perturbTests(Y, Omega, Sigma, perturb.mat, return.value = "stat")
```

Arguments

`Y`	matrix or data frame (rows=subjects, columns=variables)
`Omega`	precision matrix
`Sigma`	covariance matrix
`perturb.mat`	matrix indicating sets of which variables to test for perturbations
`return.value`	value to return (`stat` or `pval`)

Value

test statistics for perturbations at each of perturb.mat columns

References

Griffin, P.J., Johnson, W.E., and Kolaczyk, E.K. Detection of multiple perturbations in multi-omics biological networks (under review)

Examples

```
library(mvtnorm)
Omega <- matrix(0, nrow=6, ncol=6)
Omega[1:4,1:4] <- -1
diag(Omega) <- 6
Sigma <- solve(Omega)
mu <- c(1,1,0,0,0,0)
Y <- rmvnorm(20, mean=Sigma %*% mu, sigma=Sigma)
perturb.mat <- matrix(c(TRUE, TRUE, FALSE, FALSE, FALSE, FALSE,
                        FALSE, FALSE, TRUE, TRUE, FALSE, FALSE), ncol=2)
perturbTests(Y, Omega, Sigma, perturb.mat, return.value=stat)
```

`seqPerturbTests` *Sequentially test for perturbations at a series of locations*

Description

Given a matrix of possible perturbation locations, perform the sequential testing procedure described by Griffin et al. Return either test statistics or p-values.

Usage

```
seqPerturbTests(Y, Omega, Sigma, perturb.mat, return.value = "stat")
```

Arguments

`Y`	matrix or data frame (rows=subjects, columns=variables)
`Omega`	precision matrix
`Sigma`	covariance matrix
`perturb.mat`	matrix indicating sets of which variables to test for perturbations
`return.value`	return statistics (`stat`) or p-values (`pval`)

Value

test statistics for perturbations at each of perturb.mat columns

References

Griffin, P.J., Johnson, W.E., and Kolaczyk, E.K. Detection of multiple perturbations in multi-omics biological networks (under review)

Examples

```
library(mvtnorm)
Omega <- matrix(0, nrow=6, ncol=6)
Omega[1:4,1:4] <- -1
diag(Omega) <- 6
Sigma <- solve(Omega)
mu <- c(1,1,0,0,0,0)
Y <- rmvnorm(20, mean=Sigma %*% mu, sigma=Sigma)
perturb.mat <- matrix(c(TRUE, TRUE, FALSE, FALSE, FALSE, FALSE,
                        FALSE, FALSE, TRUE, TRUE, FALSE, FALSE), ncol=2)
```

```
perturbTests(Y, Omega, Sigma, perturb.mat, return.value=stat) # initial ranking
seqPerturbTests(Y, Omega, Sigma, perturb.mat, return.value=stat) # sequential adjusted
```

Index