

Package ‘covglasso’

July 14, 2021

Version 1.0.3

Date 2021-07-14

Title Sparse Covariance Matrix Estimation

Description Direct sparse covariance matrix estimation via the covariance graphical lasso by Bien, Tibshirani (2011) <[doi:10.1093/biomet/asr054](https://doi.org/10.1093/biomet/asr054)> using the fast coordinate descent algorithm of Wang (2014) <[doi:10.1007/s11222-013-9385-5](https://doi.org/10.1007/s11222-013-9385-5)>.

Maintainer Michael Fop <michael.fop@ucd.ie>

Depends R (>= 3.4)

Imports Rcpp (>= 1.0)

Suggests MASS

LinkingTo Rcpp, RcppArmadillo

License GPL-3

Repository CRAN

ByteCompile true

NeedsCompilation yes

Encoding UTF-8

Author Michael Fop [aut, cre] (<<https://orcid.org/0000-0003-3936-2757>>),
Hao Wang [ctb]

Date/Publication 2021-07-14 15:10:16 UTC

R topics documented:

covglasso-package	2
control	2
covglasso	3
Index	7

covglasso-package *Sparse covariance matrix estimation*

Description

Fast and direct estimation of a sparse covariance matrix via covariance graphical lasso and coordinate descent algorithm.

Details

A package implementing direct estimation of a sparse covariance matrix corresponding to a Gaussian covariance graphical model. Estimation is performed by solving the covariance graphical lasso using a fast coordinate descent algorithm.

How to cite this package

To cite **covglasso** in publications use:

Fop, M. (2021). covglasso: Sparse Covariance Matrix Estimation, R package version 1.0.3, <https://CRAN.R-project.org/package=covglasso>

Author(s)

Michael Fop.

Maintainer: Michael Fop <michael.fop@ucd.ie>

References

Bien, J., Tibshirani, R.J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4), 807–820.

Wang, H. (2014). Coordinate descent algorithm for covariance graphical lasso. *Statistics and Computing*, 24:521.

control *Set control parameters*

Description

Set control parameters of the coordinate descent algorithm for the graphical lasso for sparse covariance matrix estimation.

Usage

```
control(iter.out = 1e04, iter.in = 1e03, tol.out = 1e-04, tol.in = 1e-03)
```

Arguments

<code>iter.out</code>	Maximum number of iterations in the in the outer loop of the coordinate descent algorithm.
<code>iter.in</code>	Maximum number of iterations in the in the inner loop of the coordinate descent algorithm.
<code>tol.out</code>	Tolerance value for judging when convergence has been reached. Used in the outer loop of the coordinate descent algorithm.
<code>tol.in</code>	Tolerance value for judging when convergence has been reached. Used in the inner loop of the coordinate descent algorithm.

Details

Function `control` is used to set control parameters of the coordinate descent algorithm employed for solving the covariance graphical lasso.

Value

A list of parameters values.

References

Wang, H. (2014). Coordinate descent algorithm for covariance graphical lasso. *Statistics and Computing*, 24:521.

covglasso

Sparse covariance matrix estimation

Description

Direct estimation of a sparse covariance matrix using the covariance graphical lasso.

Usage

```
covglasso(data = NULL,
           S = NULL, n = NULL,
           lambda = NULL,
           rho = NULL,
           duplicated = TRUE,
           L = 10,
           crit = c("bic", "ebic"),
           gamma = 0.5,
           penalize.diag = FALSE,
           start = NULL,
           ctrl = control(),
           path = FALSE)
```

Arguments

<code>data</code>	A numerical dataframe or matrix, where rows correspond to observations and columns to variables. If <code>data = NULL</code> , the sample covariance S must be provided in input.
<code>S</code>	The sample covariance matrix of the data. If <code>S = NULL</code> , the maximum likelihood estimate of the covariance matrix is used in the estimation of the sparse covariance matrix.
<code>n</code>	The number of observations. If <code>data = NULL</code> and <code>S</code> is provided in input, <code>n</code> must be provided in input as well.
<code>lambda</code>	A vector or array of non-negative lasso regularization parameters. Penalization is applied elementwise to all entries of the covariance matrix. If an array, each entry must be a matrix with same dimensions of the sample covariance matrix. Values should be increasing from the smallest to the largest. If <code>lambda = NULL</code> , an alternative penalization based on thresholding of the empirical correlation matrix is used; see "Details".
<code>rho</code>	A vector of correlation values used to define the penalization in terms of the thresholded sample correlation matrix. See "Details". Note that this penalization is used by default.
<code>duplicated</code>	Remove duplicated penalty matrices when the default penalty term based on the thresholded correlation matrix is used. Suggest to leave this argument to <code>TRUE</code> all the time as several redundant matrices giving the same penalty term are discarded.
<code>L</code>	The number of <code>rho</code> values. Only used when <code>lambda</code> and <code>rho</code> are <code>NULL</code> . Default is <code>L = 10</code> .
<code>crit</code>	The model selection criterion employed to select the optimal covariance graph model. Can be "bic" or "ebic"; see "Details".
<code>gamma</code>	A penalty parameter used when <code>crit = "ebic"</code> and EBIC is used to select the optimal graph covariance model. The value of <code>gamma</code> must be in the range $[0, 1]$. Default is <code>gamma = 0.5</code> , which encourages sparser models.
<code>penalize.diag</code>	A logical argument indicating if the diagonal of the covariance matrix should be penalized. Default to <code>FALSE</code> .
<code>start</code>	A starting matrix for the estimation algorithm. If <code>NULL</code> , the starting value is the diagonal sample covariance matrix.
<code>ctrl</code>	A list of control parameters for the coordinate descent algorithm employed for estimation. See also control .
<code>path</code>	A logical argument controlling whether all the estimated covariance matrices along the path defined by <code>lambda</code> or <code>rho</code> should be included in the output.

Details

The function estimates a sparse covariance matrix using a fast coordinate descent algorithm to solve the covariance graphical lasso. The estimated sparse covariance matrix is obtained by optimizing the following penalized log-likelihood:

$$-\frac{n}{2} \{ \log \det(\Sigma) + \text{trace}(S\Sigma^{-1}) \} - \|\Lambda * \Sigma\|_1$$

subject to Σ being positive definite. In the penalty term, the L_1 norm and the matrix multiplication between Λ and Σ is elementwise.

By default (when `lambda = NULL`), the penalization matrix Λ is defined in terms of a sequential thresholding of the sample correlation matrix. Given ρ_l a threshold value and R the sample correlation matrix, the penalty term matrix Λ is defined by the values $(1/s_{ij})I(r_{ij} < \rho_l)$, that is:

$$\Lambda = \frac{1}{S}I(R < \rho_l)$$

where the inequality is taken elementwise. Such choice of penalty matrix provides a framework related to the adaptive lasso of Fan et al. (2009) and the method of Chaudhuri et al. (2007). If the vector `rho` is not given in input, the sequence of threshold values is defined as the L quantiles of the absolute values of the sample correlations in R . If `lambda` is provided in input, the penalization corresponds to the standard covariance graphical lasso of Bien, Tibshirani (2011).

The sparse covariance matrix corresponds to a Gaussian covariance graphical model of marginal independence, where in the sparse covariance matrix a zero entry corresponds to two variables being marginally independent. Different penalizations `lambda` imply different models, and selection of the optimal graphical model is performed using "`bic`" (default) or "`ebic`". In the latter case, the argument `gamma` controls the additional penalty term in the model selection criterion; see Foygel, Drton, (2010).

Value

A list containing the following elements.

<code>sigma</code>	The estimated covariance matrix.
<code>omega</code>	The estimated concentration (inverse covariance) matrix.
<code>graph</code>	The adjacency matrix given in input corresponding to the marginal or conditional independence graph.
<code>loglik</code>	Value of the maximized log-likelihood.
<code>npar</code>	Number of estimated non-zero parameters.
<code>penalty</code>	Value of the penalty term.
<code>bic</code>	Optimal BIC or EBIC value.
<code>BIC</code>	All BIC or EBIC values along the path defined by <code>lambda</code> or <code>rho</code> .
<code>path</code>	A list containing all the estimated sparse covariance models. Provided in output only when <code>path = TRUE</code> .
<code>rho</code>	The values of <code>rho</code> thresholds used to define the penalization based on the thresholded sample correlation matrix.
<code>lambda</code>	The values of <code>lambda</code> penalty parameters for the penalization.

References

- Bien, J., Tibshirani, R.J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4), 807–820.
- Chaudhuri, S., Drton M., Richardson, T. S. (2007). Estimation of a covariance matrix with zeros. *Biometrika*, 94(1), 199-216.

Fan, J., Feng, Y., Wu, Y. (2009). Network exploration via the adaptive lasso and scad penalties. *The Annals of Applied Statistics*, 3(2), 521.

Foygel, R., Drton, M. (2010). Extended Bayesian information criteria for Gaussian graphical models. In *Advances in neural information processing systems*, pages 604–612.

Wang, H. (2014). Coordinate descent algorithm for covariance graphical lasso. *Statistics and Computing*, 24:521.

See Also

[control](#)

Examples

```
# a simple example with a 3-block diagonal matrix
library(MASS)
p <- 3
n <- 300
sig <- matrix(0.8, p,p)
diag(sig) <- 1
set.seed(190188)
tmp <- replicate( 3, mvrnorm(n, rep(0,p), sig) )
x <- matrix(c(tmp), n, p*3)

fit1 <- covglasso(x)
plot(fit1$rho, fit1$BIC)
image(fit1$sigma != 0)

# refine search
fit2 <- covglasso(x, rho = seq(0.1, 0.4, length = 50))
image(fit2$sigma != 0)

fit1$bic
fit2$bic

# Cars93 data in MASS package
data("Cars93", package = "MASS")
dat <- na.omit( Cars93[,c(4:8,12:15,17,19:25)] )

fit1 <- covglasso(dat, L = 50)

# more sparse
fit2 <- covglasso(dat, L = 50,
                  crit = "ebic", gamma = 1)

oldpar <- par(no.readonly = TRUE)
par(mfrow = c(1,2))
plot(fit1$rho, fit1$BIC, main = "BIC")
plot(fit2$rho, fit2$BIC, main = "EBIC")
image(fit1$sigma != 0, col = c("white", "black"), main = "BIC")
image(fit2$sigma != 0, col = c("white", "black"), main = "EBIC")
par(oldpar) # reset par
```

Index

*** package**

covglasso-package, [2](#)

control, [2](#), [4](#), [6](#)

covglasso, [3](#)

covglasso-package, [2](#)