Package ‘covglasso’

October 12, 2022

Version 1.0.3
Date 2021-07-14
Title Sparse Covariance Matrix Estimation
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
covglasso ................................................................. 2
control ................................................................. 3
Index 7
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:


Author(s)

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

References


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)
**covglasso**

**Sparse covariance matrix estimation**

**Description**

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

**Usage**

```r
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**

- **iter.out**: Maximum number of iterations in the outer loop of the coordinate descent algorithm.
- **iter.in**: Maximum number of iterations in the inner loop of the coordinate descent algorithm.
- **tol.out**: Tolerance value for judging when convergence has been reached. Used in the outer loop of the coordinate descent algorithm.
- **tol.in**: 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**

Arguments

- **data**: A numerical dataframe or matrix, where rows correspond to observations and columns to variables. If `data = NULL`, the sample covariance `S` must be provided in input.

- **S**: The sample covariance matrix of the data. If `S = NULL`, the maximum likelihood estimate of the covariance matrix is used in the estimation of the sparse covariance matrix.

- **n**: The number of observations. If `data = NULL` and `S` is provided in input, `n` must be provided in input as well.

- **lambda**: 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 `lambda = NULL`, an alternative penalization based on thresholding of the empirical correlation matrix is used; see "Details".

- **rho**: 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.

- **duplicated**: Remove duplicated penalty matrices when the default penalty term based on the thresholded correlation matrix is used. Suggest to leave this argument to `TRUE` all the time as several redundant matrices giving the same penalty term are discarded.

- **L**: The number of `rho` values. Only used when `lambda` and `rho` are `NULL`. Default is `L = 10`.

- **crit**: The model selection criterion employed to select the optimal covariance graph model. Can be "bic" or "ebic"; see "Details".

- **gamma**: A penalty parameter used when `crit = "ebic"` and EBIC is used to select the optimal graph covariance model. The value of `gamma` must be in the range `[0,1]`. Default is `gamma = 0.5`, which encourages sparser models.

- **penalize.diag**: A logical argument indicating if the diagonal of the covariance matrix should be penalized. Default to `FALSE`.

- **start**: A starting matrix for the estimation algorithm. If `NULL`, the starting value is the diagonal sample covariance matrix.

- **ctrl**: A list of control parameters for the coordinate descent algorithm employed for estimation. See also `control`.

- **path**: A logical argument controlling whether all the estimated covariance matrices along the path defined by `lambda` or `rho` 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 $\Sigma$ being positive definite. In the penalty term, the $L_1$ norm and the matrix multiplication between $\Lambda$ and $\Sigma$ is elementwise.

By default (when $\lambda$ = NULL), the penalization matrix $\Lambda$ is defined in terms of a sequential thresholding of the sample correlation matrix. Given $\rho_l$ a threshold value and $R$ the sample correlation matrix, the penalty term matrix $\Lambda$ 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.

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

References


**See Also**

control

**Examples**

```r
# 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

covglasso-package, 2
control, 2, 4, 6
covglasso, 3
covglasso-package, 2