Package ‘dpcid’

December 30, 2018

Type Package

Title Differential Partial Correlation IDentification

Version 1.0

Date 2018-12-15

Author Donghyeon Yu <dyu@inha.ac.kr>
Sang Han Lee <shlee@nki.rfmh.org>
Johan Lim <johanlim@stats.snu.ac.kr>
Guanghua Xiao <guanghua.xiao@utsouthwestern.edu>
R. Cameron Craddock <cameron.craddock@gmail.com>
Bharat B. Biswal <bharat.biswal@njit.edu>

Maintainer Donghyeon Yu <dyu@inha.ac.kr>

Suggests MASS

Description Differential partial correlation identification with the ridge and the fusion penalties.

License GPL (>= 2)

URL https://sites.google.com/site/dhyeonyu/software

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-12-30 17:20:09 UTC

R topics documented:

crit.dpcid .................................................. 2
cv.lambda1 .................................................. 3
dpcid ...................................................... 5
dpcid_core .................................................. 7
dpcid_l1_core ............................................... 9
lshr.cov .................................................... 11

Index 13
crit.dpcid

Description

aic.dpcid returns the AIC values corresponding to the given lambda1 and lambda2 values for the DPCID.

Usage

crit.dpcid(A,B,l1,seq_l2,wd1,wd2,rho1_init,rho2_init,niter=1000,tol=1e-6,scaling=FALSE)

Arguments

- A: An observed dataset from the first condition.
- B: An observed dataset from the second condition.
- l1: The selected lambda1 in cv.lambda1.
- seq_l2: A sequence of tuning parameter lambda2 for the fusion penalty.
- wd1: The estimate of diagonal elements of the precision matrix of the first condition.
- wd2: The estimate of diagonal elements of the precision matrix of the second condition.
- rho1_init: An initial value for the partial correlation matrix of the first condition.
- rho2_init: An initial value for the partial correlation matrix of the second condition.
- niter: A total number of iterations in the block-wise coordinate descent.
- tol: A tolerance for the convergence.
- scaling: A logical flag for scaling variable to have unit variance. Default is FALSE.

Details

crit.dpcid needs the estimates of the diagonal elements of two precision matrices.

Value

- aic: A vector of aic values corresponding to a given sequence of tuning parameters.
- bic: A vector of bic values corresponding to a given sequence of tuning parameters.

References

Examples

library(MASS)

## True precision matrix
omega1 <- matrix(0,5,5)
omega1[1,2] <- omega1[1,3] <- omega1[1,4] <- 1
omega1[2,3] <- omega1[3,4] <- 1.5
omega1 <- t(omega1) + omega1
diag(omega1) <- 3

omega2 <- matrix(0,5,5)
omega2[1,3] <- omega2[1,5] <- 1.5
omega2 <- t(omega2) + omega2
diag(omega2) <- 3

Sig1 = solve(omega1)
Sig2 = solve(omega2)
X1 = mvrnorm(50,rep(0,5),Sig1)
X2 = mvrnorm(50,rep(0,5),Sig2)

A = scale(X1,center=TRUE,scale=FALSE)
B = scale(X2,center=TRUE,scale=FALSE)

shr_res = lshr.cov(A)
PM1 = shr_res$shr_inv

shr_res = lshr.cov(B)
PM2 = shr_res$shr_inv

wd1 = diag(PM1)
wd2 = diag(PM2)

rho1_init = -(1/sqrt(wd1))*PM1
rho1_init = t((1/sqrt(wd1))*t(rho1_init))
diag(rho1_init) = 1

rho2_init = -(1/sqrt(wd2))*PM2
rho2_init = t((1/sqrt(wd2))*t(rho2_init))
diag(rho2_init) = 1

l1 = 0.3
seq_l2 = seq(0.1,1,by=0.2)

crit = crit.dpcid(A,B,l1,seq_l2,wd1,wd2,rho1_init,rho2_init)
crit$aic
crit$bic

---

K-fold Crossvalidation for the ridge step
Description

K-fold crossvalidation for the choice of the lambda1.

Usage

\[ \text{cv.lambda1}(A, B, nfold, seq_lambda1, niter=1000, tol=1e-6, scaling=\text{FALSE}) \]

Arguments

- **A**: An observed dataset from the first condition.
- **B**: An observed dataset from the second condition.
- **nfold**: the number of folds in the crossvalidation (i.e., K in K-fold cross validation)
- **seq_lambda1**: A sequence of tuning parameters for the ridge penalty
- **niter**: A total number of iterations in the block-wise coordinate descent.
- **tol**: A tolerance for the convergence.
- **scaling**: a logical flag for scaling variable to have unit variance. Default is \text{FALSE}.

Details

\text{cv.lambda1} returns a vector of the K-fold crossvalidated errors and matrices of the initial estimates of the precision matrices.

Value

- **cv**: A vector of crossvalidated errors corresponding to a given sequence of tuning parameters.
- **pm1**: A matrix of the inverse of the linear shrinkage covariance estimates for the first condition.
- **pm2**: A matrix of the inverse of the linear shrinkage covariance estimates for the second condition.

References


Examples

library(MASS)

```r
## True precision matrix
omega1 <- matrix(0, 5, 5)
omega1[1, 2] <- omega1[1, 3] <- omega1[1, 4] <- 1
omega1[2, 3] <- omega1[3, 4] <- 1.5
omega1 <- t(omega1) + omega1
diag(omega1) <- 3
```
omegaR <- matrix(0.5, 5)  # Create a 5x5 matrix omegaR
omegaR[1, 3] <- omegaR[1, 5] <- 1.5
omegaR <- t(omegaR) + omegaR
diag(omegaR) <- 3

Sig1 = solve(omega1)
Sig2 = solve(omega2)
X1 = mvrnorm(50, rep(0, 5), Sig1)
X2 = mvrnorm(50, rep(0, 5), Sig2)

nfold = 5
seq_lam1 = seq(0.5, 3, by=0.5)
cv_vec = cv.lambdal(X1, X2, nfold, seq_lam1, niter=1000, tol=1e-6)$cv

dpcid <- function(A, B, lambda1, lambda2, niter=1000, tol=1e-6, scaling=FALSE) {
  # Differential partial correlation identification with the ridge and the fusion penalties

  # Description
  DPCID is a procedure for the differential partial correlation identification with the ridge and the fusion penalties. This function conducts the two stage procedure (diagonal and partial correlation steps).

  # Usage
  dpcid(A, B, lambda1, lambda2, niter=1000, tol=1e-6, scaling=FALSE)

  # Arguments
  A An observed dataset from the first condition.
  B An observed dataset from the second condition.
  lambda1 A tuning parameter for the ridge penalty.
  lambda2 A tuning parameter for the fusion penalty between two precision matrices.
  niter A total number of iterations in the block-wise coordinate descent.
  tol A tolerance for the convergence.
  scaling a logical flag for scaling variable to have unit variance. Default is FALSE.

  # Details
  In the first step (lshr.cov), each precision matrix is estimated from the optimal linear shrinkage covariance matrix. In the second step (dpcid_core), two partial correlation matrices are jointly estimated with a given tuning parameters lambda1 and lambda2 and fixed diagonal elements of two precision matrices.
Value

- **rho1**: An estimated partial correlation matrix of the first condition.
- **rho2**: An estimated partial correlation matrix of the second condition.
- **wd1**: A vector of estimated diagonal elements of the first precision matrices.
- **wd2**: A vector of estimated diagonal elements of the second precision matrices.
- **diff_edge**: An index matrix of different edges between two conditions.
- **n_diff**: The number of different edges between two conditions.

References


Examples

```r
library(MASS)

## True precision matrix
omega1 <- matrix(0,5,5)
omega1[1,2] <- omega1[1,3] <- omega1[1,4] <- 1
omega1[2,3] <- omega1[3,4] <- 1.5
omega1 <- t(omega1) + omega1
diag(omega1) <- 3

omega2 <- matrix(0,5,5)
omega2[1,3] <- omega2[1,5] <- 1.5
omega2 <- t(omega2) + omega2
diag(omega2) <- 3

Sig1 = solve(omega1)
Sig2 = solve(omega2)
X1 = mvrnorm(50,rep(0,5),Sig1)
X2 = mvrnorm(50,rep(0,5),Sig2)

lambda1 = 0.2
lambda2 = 0.2

res = dpcid(X1,X2,lambda1,lambda2,niter=1000,tol=1e-6)
```


**dpcid_core**

*Identification of two partial correlation matrices having sparse differences*

**Description**

dpcid_core estimates two partial correlation matrices by applying the regression approach with the ridge penalty and the fusion penalty.

**Usage**

dpcid_core(A,B,lambda1,lambda2,wd1,wd2,rho1_init,rho2_init,niter=1000,tol=1e-6)

**Arguments**

- **A** An observed dataset from the first condition.
- **B** An observed dataset from the second condition.
- **lambda1** A tuning parameter for the ridge penalty.
- **lambda2** A tuning parameter for the fusion penalty between two precision matrices.
- **wd1** The estimate of diagonal elements of the precision matrix of the first condition.
- **wd2** The estimate of diagonal elements of the precision matrix of the second condition.
- **rho1_init** An initial value for the partial correlation matrix of the first condition.
- **rho2_init** An initial value for the partial correlation matrix of the second condition.
- **niter** A total number of iterations in the block-wise coordinate descent.
- **tol** A tolerance for the convergence.

**Details**

Dpcid_core is the partial correlation step of the differential partial correlation identification method by Yu et al. (2018). The dpcid_core estimates two partial correlation matrices with the estimated diagonal elements of two precision matrices from the optimal linear shrinkage estimates. The estimated precision matrices by the optimal linear shrinkage estimates can simply be used as the initial values of PM1 and PM2 in the dpcid_core.

**Value**

- **rho1** An estimated partial correlation matrix of the first condition.
- **rho2** An estimated partial correlation matrix of the second condition.
- **resid1** Residuals of the first condition.
- **resid2** Residuals of the second condition.
References

Examples

```r
library(MASS)

## true precision matrix
omega1 <- matrix(0,5,5)
omega1[1,2] <- omega1[1,3] <- omega1[1,4] <- 1
omega1[2,3] <- omega1[3,4] <- 1.5
omega1 <- t(omega1) + omega1
diag(omega1) <- 3

omega2 <- matrix(0,5,5)
omega2[1,3] <- omega2[1,5] <- 1.5
omega2 <- t(omega2) + omega2
diag(omega2) <- 3

Sig1 = solve(omega1)
Sig2 = solve(omega2)
X1 = mvrnorm(50,rep(0,5),Sig1)
X2 = mvrnorm(50,rep(0,5),Sig2)

A = scale(X1,center=TRUE,scale=FALSE)
B = scale(X2,center=TRUE,scale=FALSE)

lambda1 = 1
lambda2 = 1

shr_res = lshr.cov(A)
PM1 = shr_res$shr_inv
shr_res = lshr.cov(B)
PM2 = shr_res$shr_inv

wd1 = diag(PM1)
wd2 = diag(PM2)

rho1_init = -(1/sqrt(wd1))*PM1
rho1_init = t( 1/sqrt(wd1)*t(rho1_init))
diag(rho1_init) = 1

rho2_init = -(1/sqrt(wd2))*PM2
rho2_init = t( 1/sqrt(wd2)*t(rho2_init))
diag(rho2_init) = 1

res = dpcid_core(A, B, lambda1, lambda2, wd1, wd2, rho1_init, rho2_init)
```

```
Identification of two partial correlation matrices having sparse differences

**Description**

dpcid\_l1\_core estimates two partial correlation matrices by applying the regression approach with the lasso penalty and the fusion penalty.

**Usage**

dpcid\_l1\_core(A,B,lambda1,lambda2,wd1,wd2,rho1\_init,rho2\_init,niter=1000,tol=1e-6)

**Arguments**

- **A**: An observed dataset from the first condition.
- **B**: An observed dataset from the second condition.
- **lambda1**: A tuning parameter for the lasso penalty.
- **lambda2**: A tuning parameter for the fusion penalty between two precision matrices.
- **wd1**: The estimate of diagonal elements of the precision matrix of the first condition.
- **wd2**: The estimate of diagonal elements of the precision matrix of the second condition.
- **rho1\_init**: An initial value for the partial correlation matrix of the first condition.
- **rho2\_init**: An initial value for the partial correlation matrix of the second condition.
- **niter**: A total number of iterations in the block-wise coordinate descent.
- **tol**: A tolerance for the convergence.

**Details**

Dpcid\_l1\_core is the partial correlation step of the differential partial correlation identification method with the lasso penalty instead of the ridge penalty by Yu et al.(2018). The dpcid\_l1\_core estimates two partial correlation matrices with the estimated diagonal elements of two precision matrices from the optimal linear shrinkage estimates. The estimated precision matrices by the optimal linear shrinkage estimates can simply be used as the initial values of PM1 and PM2 in the dpcid\_core.

**Value**

- **rho1**: An estimated partial correlation matrix of the first condition.
- **rho2**: An estimated partial correlation matrix of the second condition.
- **resid1**: Residuals of the first condition.
- **resid2**: Residuals of the second condition.
References

Examples

```r
library(MASS)

## true precision matrix
omega1 <- matrix(0,5,5)
omega1[1,2] <- omega1[1,3] <- omega1[1,4] <- 1
omega1[2,3] <- omega1[3,4] <- 1.5
omega1 <- t(omega1) + omega1
diag(omega1) <- 3

omega2 <- matrix(0,5,5)
omega2[1,3] <- omega2[1,5] <- 1.5
omega2 <- t(omega2) + omega2
diag(omega2) <- 3

Sig1 = solve(omega1)
Sig2 = solve(omega2)
X1 = mvrnorm(50, rep(0,5), Sig1)
X2 = mvrnorm(50, rep(0,5), Sig2)

A = scale(X1, center=TRUE, scale=FALSE)
B = scale(X2, center=TRUE, scale=FALSE)

lambda1 = 1
lambda2 = 1

shr_res = lshr.cov(A)
PM1 = shr_res$shr_inv
shr_res = lshr.cov(B)
PM2 = shr_res$shr_inv

wd1 = diag(PM1)
wd2 = diag(PM2)

rho1_init = -(1/sqrt(wd1)) * PM1
rho1_init = t(1/sqrt(wd1) * t(rho1_init))
diag(rho1_init) = 1

rho2_init = -(1/sqrt(wd2)) * PM2
rho2_init = t(1/sqrt(wd2) * t(rho2_init))
diag(rho2_init) = 1

res = dpcid_11_core(A, B, lambda1, lambda2, wd1, wd2, rho1_init, rho2_init)
```
Linear shrinkage estimates of covariance and inverse covariance matrix

Description

Linear shrinkage estimates of covariance and inverse covariance matrix.

Usage

lshr.cov(X, scaling=FALSE)

Arguments

- **X**: An observed dataset from a specific condition.
- **scaling**: A logical flag for scaling variable to have unit variance. Default is FALSE.

Details

`shr_covp` returns the optimal linear shrinkage parameter, the linear shrinkage estimates of the covariance and the precision matrix.

Value

- **shr_cov**: Linear shrinkage estimate of the covariance matrix.
- **shr_inv**: Linear shrinkage estimate of the inverse covariance matrix.

References


Examples

```r
library(MASS)

## True precision matrix
omega <- matrix(0, 5, 5)
omega[1, 2] <- omega[1, 3] <- omega[1, 4] <- 1
omega[2, 3] <- omega[3, 4] <- 1.5
omega <- t(omega) + omega
diag(omega) <- 3
```
Sig = solve(omega)
X = mvrnorm(50, rep(0,5), Sig)
lshr.cov(X)
Index

crit.dpcid, 2
cv.lambda1, 3
dpcid, 5
dpcid_core, 7
dpcid_l1_core, 9
lshr.cov, 11