Package 'Tlasso'

October 12, 2022

Type Package

Title Non-Convex Optimization and Statistical Inference for Sparse Tensor Graphical Models

Version 1.0.2

Description An optimal alternating optimization algorithm for estimation of precision matrices of sparse tensor graphical models, and an efficient inference procedure for support recovery of the precision matrices.

Imports huge, expm, rTensor, igraph, stats, graphics

Suggests knitr, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

Author Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng

Maintainer Xiang Lyu <xianglyu@berkeley.edu>

Depends R (>= 3.1.1)

License GPL (>= 2)

LazyData false

RoxygenNote 7.1.2

NeedsCompilation no

Repository CRAN

Date/Publication 2022-02-01 08:20:08 UTC

R topics documented:

`biascor` *Bias Correction of Sample Covariance of Residuals*

Description

Generate a matrix of bias-corrected sample covariance of residuals (excludes diagnoal) described in Lyu et al. (2019).

Usage

```
biascor(rho, Omega.list, k = 1)
```

Arguments

`rho`	matrix of sample covariance of residuals (includes diagnoal), e.g., output of `covres`.
`Omega.list`	list of precision matrices of tensor, i.e., `Omega.list[[k]]` is the precision matrix for the kth tensor mode, $k \in \{1, \ldots, K\}$. For example, output of `link{Tlasso.fit}`.
`k`	index of interested mode, default is 1.

Details

This function computes bias-corrected sample covariance of residuals (excludes diagnoal, diagnoal is zero vector). Note that output matrix excludes diagnoal while sample covariance of residuals includes diagnoal, see Lyu et al. (2019) for details. Elements in `Omega.list` are true precision matrices or estimation of the true ones, the latter can be output of `Tlasso.fit`.

Value

A matrix whose (i,j) entry (excludes diagnoal; diagnoal is zero vector) is bias-corrected sample covariance of the ith and jth residuals in the kth mode. See Lyu et al. (2019) for details.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

`varcor`, `covres`

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
k=1 # index of interested mode
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
# obersavations from tensor normal distribution
out.tlasso = Tlasso.fit(DATA,T=1,lambda.vec = lambda.thm)
# output is a list of estimation of precision matrices

rho=covres(DATA, out.tlasso, k = k)
# sample covariance of residuals, including diagnoal
bias_rho=biascor(rho,out.tlasso,k=k)
bias_rho # bias-corrected sample covariance of residuals
# diagnoal is zero vector
```

`ChainOmega` *Precision Matrix of Triangle Graph*

Description

Generate precision matrix of triangle graph (chain like network) following the set-up in Fan et al. (2009).

Usage

```
ChainOmega(p, sd = 1, norm.type = 2)
```

Arguments

`p`	dimension of generated precision matrix.
`sd`	seed for random number generation, default is 1.
`norm.type`	normalization methods of generated precision matrix, i.e., $\Omega_{11} = 1$ if norm.type = 1 and $\|\Omega\|_F = 1$ if norm.type = 2. Default value is 2.

Details

This function first construct a covariance matrix Σ that its (i,j) entry is $\exp(-|h_i - h_j|/2)$ with $h_1 < h_2 < \ldots < h_p$. The difference $h_i - h_{i+1}$ is generated i.i.d. from Unif(0.5,1). See Fan et al. (2009) for more details.

Value

A precision matrix generated from triangle graph.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

NeighborOmega

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size

Omega.true.list = list()

for ( k in 1:length(m.vec)){
 Omega.true.list[[k]] = ChainOmega(m.vec[k],sd=k*100,norm.type=2)
}
Omega.true.list  # a list of length 3 contains precision matrices from triangle graph
```

---

covres *Sample Covariance Matrix of Residuals*

Description

Generate sample covariance matrix of residuals (includes diagnoal) described in Lyu et al. (2019).

Usage

```
covres(data, Omega.list, k = 1)
```

Arguments

data	tensor object stored in a m1 * m2 * ... * mK * n array, where n is sample size and mk is dimension of the kth tensor mode.
Omega.list	list of precision matrices of tensor, i.e., Omega.list[[k]] is precision matrix for the kth tensor mode, $k \in \{1, \ldots, K\}$.
k	index of interested mode, default is 1.

Details

This function computes sample covariance of residuals and is the basis for support recovery procedure in Lyu et al. (2019). Note that output matrix includes diagnoal while bias corrected matrix (output of biascor) for inference is off-diagnoal, see Lyu et al. (2019) for details. Elements in Omega.list are true precision matrices or estimation of the true ones, the latter can be output of Tlasso.fit.

Value

A matrix whose (i,j) entry (includes diagnoal) is sample covariance of the ith and jth residuals in the kth mode. See Lyu et al. (2019) for details.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

varcor, biascor

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
k=1 # index of interested mode
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
# obersavations from tensor normal distribution
out.tlasso = Tlasso.fit(DATA,T=1,lambda.vec = lambda.thm)
# output is a list of estimation of precision matrices
rho=covres(DATA, out.tlasso, k = k) # sample covariance of residuals, including diagnoal
rho
```

---

est.analysis *Estimation Errors and TPR/TNR*

Description

Compute estimation errors and TPR/TNR of optimization for sparse tensor graphical models

Usage

```
est.analysis(Omega.hat.list, Omega.true.list, offdiag = TRUE)
```

Arguments

Omega.hat.list	list of estimation of precision matrices of tensor, i.e., Omega.hat.list[[k]] is estimation of precision matrix for the kth tensor mode, $k \in \{1, \ldots, K\}$. For example, output of Tlasso.fit.
Omega.true.list	list of true precision matrices of tensor, i.e., Omega.true.list[[k]] is true precision matrix for the kth tensor mode, $k \in \{1, \ldots, K\}$.

`offdiag` logical; indicate if excludes diagnoal when computing performance measures. If `offdiag = TRUE`, diagnoal in each matrix is ingored when comparing two matrices. Default is `TRUE`.

Details

This function computes performance measures of optimazation for sparse tensor graphical models. Errors are measured in Frobenius norm and Max norm. Model selection measures are TPR and TNR. All these measures are computed in each mode, average across all modes, and kronecker production of precision matrices.

Value

A list, named `Out`, of following performance measures:

`Out$error.kro`	error in Frobenius norm of kronecker product
`Out$tpr.kro`	TPR of kronecker product
`Out$tnr.kro`	TNR of kronecker product
`Out$av.error.f`	averaged Frobenius norm error across all modes
`Out$av.error.max`	averaged Max norm error across all modes
`Out$av.tpr`	averaged TPR across all modes
`Out$av.tnr`	averaged TNR across all modes
`Out$error.f`	vector; error in Frobenius norm of each mode
`Out$error.max`	vector; error in Max norm of each mode
`Out$tpr`	vector; TPR of each mode
`Out$tnr`	vector; TNR of each mode

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

`Tlasso.fit`, `NeighborOmega`, `ChainOmega`

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
k=1 # index of interested mode
Omega.true.list = list()
Omega.true.list[[1]] = ChainOmega(m.vec[1], sd = 1)
Omega.true.list[[2]] = ChainOmega(m.vec[2], sd = 2)
Omega.true.list[[3]] = ChainOmega(m.vec[3], sd = 3)
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
# obersavations from tensor normal distribution
```

```
out.tlasso = Tlasso.fit(DATA,T=1,lambda.vec = lambda.thm)
# output is a list of estimation of precision matrices
est.analysis(out.tlasso, Omega.true.list, offdiag=TRUE)
# generate a list of performance measures
```

`graph.pattern` *Graph Pattern Visualization*

Description

Draw an undirected graph based on presicion matrix to present connection among variables.

Usage

```
graph.pattern(
  mat,
  main = NULL,
  edge.color = "gray50",
  vertex.color = "red",
  vertex.size = 3,
  vertex.label = NA,
  thres = 1e-05
)
```

Arguments

`mat`	precision matrix that encodes information of graph struture.
`main`	main title of graph. Default is `NULL`.
`edge.color`	color of edge. Default is `"gray50"`.
`vertex.color`	color of vertex. Default is `"red"`.
`vertex.size`	size of vertex. Default is 3.
`vertex.label`	label of vertex. Default is `NA`.
`thres`	thresholding level of substituting entry with zero, set entry to zero if its absolute value equals or is less than `thres`. If `thres` is negative or zero, no entry will be substituted with zero.

Details

This function generates an udirected graph based on precision matrix. If an entry is zero, then no edge connects corresponding pair of nodes.

Value

A plot of undirected graph.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

infer.analysis, est.analysis

Examples

```
graph.pattern(ChainOmega(5, sd = 13))
# a triangle graph
```

---

infer.analysis *Inference Performance Measures*

Description

False positive, false negative, discoveries, and non-discoveries of inference for sparse tensor graphical models.

Usage

```
infer.analysis(mat.list, critical, Omega.true.list, offdiag = TRUE)
```

Arguments

mat.list	list of matrices. (i,j) entry in its kth element is test statistic value for (i,j) entry of kth true precision matrix.
critical	critical level of rejecting null hypothesis. If critical is not positive, all null hypothesis will not be rejected.
Omega.true.list	list of true precision matrices of tensor, i.e., Omega.true.list[[k]] is true precision matrix for the kth tensor mode, $k \in \{1, \ldots, K\}$.
offdiag	logical; indicate if excludes diagnoal when computing performance measures. If offdiag = TRUE, diagnoal in each matrix is ingored when comparing two matrices. Default is TRUE.

Details

This function computes performance measures of inference for sparse tensor graphical models. False positive, false negative, discovery (number of rejected null hypothesis), non-discovery (number of non-rejected null hypothesis), and total non-zero entries of each true precision matrix is listed in output.

Value

A list, named `Out`, of following performance measures:

Out$fp	vector; number of false positive of each mode
Out$fn	vector; number of false negative of each mode
Out$d	vector; number of all discovery of each mode
Out$nd	vector; number of all non-discovery of each mode
Out$t	vector; number of all true non-zero entries in true precision matrix of each mode

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

Tlasso.fit, est.analysis, ChainOmega

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
Omega.true.list = list()
Omega.true.list[[1]] = ChainOmega(m.vec[1], sd = 1)
Omega.true.list[[2]] = ChainOmega(m.vec[2], sd = 2)
Omega.true.list[[3]] = ChainOmega(m.vec[3], sd = 3)
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
# obersavations from tensor normal distribution
out.tlasso = Tlasso.fit(DATA,T=1,lambda.vec = lambda.thm)
# output is a list of estimation of precision matrices
mat.list=list()
for ( k in 1:3) {
  rho=covres(DATA, out.tlasso, k = k)
  # sample covariance of residuals, including diagnoal
  varpi2=varcor(DATA, out.tlasso, k = k)
  # variance correction term for kth mode's sample covariance of residuals
  bias_rho=biascor(rho,out.tlasso,k=k)
  # bias corrected

  tautest=matrix(0,m.vec[k],m.vec[k])
  for( i in 1:(m.vec[k]-1)) {
    for ( j in (i+1):m.vec[k]){
      tautest[j,i]=tautest[i,j]=sqrt((n-1)*prod(m.vec[-k]))*
        bias_rho[i,j]/sqrt(varpi2*rho[i,i]*rho[j,j])
    }
  }
  # list of matrices of test statistic values (off-diagnoal). See Sun et al. 2016
  mat.list[[k]]=tautest
}

infer.analysis(mat.list, qnorm(0.975), Omega.true.list, offdiag=TRUE)
```

```
# inference measures (off-diagnoal)
```

NeighborOmega *Precision Matrix of Nearest-Neighbor Graph*

Description

Generate precision matrix of nearest-neighbor network following the set-up in Li and Gui (2006) and Lee and Liu (2006).

Usage

```
NeighborOmega(p, sd = 1, knn = 4, norm.type = 2)
```

Arguments

p	dimension of generated precision matrix.
sd	seed for random number generation. Default is 1.
knn	sparsity of precision matrix, i.e., matrix is generated from a knn nearest-neighbor graph. knn should be less than p. Default is 4.
norm.type	normalization methods of generated precision matrix, i.e., $\Omega_{11} = 1$ if norm.type = 1 and $\|\Omega\|_F = 1$ if norm.type = 2. Default value is 2.

Details

For a knn nearest-neighbor graph, this function first randomly picks p points from a unit square and computes all pairwise distances among the points. Then it searches for the knn nearest-neighbors of each point and a pair of symmetric entries in the precision matrix that has a random chosen value from $[-1, -0.5] \cup [0.5, 1]$. Finally, to ensure positive definite property, it normalizes the matrix as $\Omega < -\Omega + (\lambda(\Omega) + 0.2)1_p$ where $\lambda(\cdot)$ refers to the samllest eigenvalue.

Value

A precision matrix generated from the knn nearest-neighor graph.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

ChainOmega

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
knn=4 # sparsity

Omega.true.list = list()

for ( k in 1:length(m.vec)){
  Omega.true.list[[k]] = NeighborOmega(m.vec[k],knn=4, sd=k*100,norm.type=2)
}
Omega.true.list  # a list of length 3 contains precision matrices from 4-nearnest neighbor graph
```

`signal` *Regression Parameter of Conditional Linear Model*

Description

Compute regression parameter of conditional linear model of separable tensor normal distribution described in Lyu et al. (2019).

Usage

```
signal(Omega.list, i = 1, k = 1)
```

Arguments

`Omega.list`	list of precision matrices of tensor, i.e., `Omega.list[[k]]` is the kth precision matrix. Omega.list can be either true precision matrices or output of `Tlasso.fit`. for the kth tensor mode, $k \in \{1, \ldots, K\}$.
`i`	index of interested regression parameter, default is 1. See details in Lyu et al. (2019).
`k`	index of interested mode, default is 1.

Details

This function computes regression parameter and is fundamental for sample covariance of residuals and bias correction. See details in Lyu et al. (2019).

Value

A vector of regression paramter.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

covres, biascor

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
k=1 # index of interested mode
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
# obersavations from tensor normal distribution
out.tlasso = Tlasso.fit(DATA,T=1,lambda.vec = lambda.thm)
# output is a list of estimation of precision matrices
signal(out.tlasso, i=2 , k=k )
# the regression parameter for conditional linear model of 2rd row in 1st mode
```

Tlasso *Non-Convex Optimization and Statistical Inference for Sparse Tensor Graphical Models*

Description

An optimal alternating optimization algorithm for estimation of precision matrices of sparse tensor graphical models, and an efficient inference procedure for support recovery of the precision matrices.

Details

Package:	Tlasso
Type:	Package
Date	2016-09-17
License:	GPL (>= 2)

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.
Maintainer: Xiang Lyu <xianglyu@berkeley.edu>

References

Fan J, Feng Y, Wu Y. *Network exploration via the adaptive LASSO and SCAD penalties.* The annals of applied statistics, 2009,
Friedman J, Hastie T, Tibshirani R. *Sparse inverse covariance estimation with the graphical lasso.* Biostatistics, 2008: 9.3: 43
Lee W, Liu Y. *Joint estimation of multiple precision matrices with common structures.* Journal of Machine Learning Research,
Li H, Gui J. *Gradient directed regularization for sparse Gaussian concentration graphs, with applications to inference of gene*
Lyu X, Sun W, Wang Z, Liu H, Yang J, Cheng G. *Tensor Graphical Model: Non-convex Optimization and Statistical Inference*

`Tlasso.fit` *Non-Convex Optimization for Sparse Tensor Graphical Models*

Description

An alternating optimization algorithm for estimation of precision matrices of sparse tensor graphical models. See Lyu et al. (2019) for details.

Usage

```
Tlasso.fit(data, T = 1, lambda.vec = NULL, norm.type = 2, thres = 1e-05)
```

Arguments

`data`	tensor object stored in a m1 * m2 * ... * mK * n array, where n is sample size and mk is dimension of the kth tensor mode.
`T`	number of maximal iteration, default is 1. Each iteration involves update on all modes. If output change less than `thres` after certain iteration, in terms of summation on Frobenius norm, this function will be terminated (before Tth iteration).
`lambda.vec`	vector of tuning parameters (λ_1,...,λ_K). Defalut is NULL, s.t. it is tuned via `HUGE` package directly.
`norm.type`	normalization method of precision matrix, i.e., $\Omega_{11} = 1$ if norm.type = 1 and $\|\Omega\|_F = 1$ if norm.type = 2. Default value is 2.
`thres`	thresholding value that terminates algorithm before Tth iteration if output change less than `thres` after certain iteration, in terms of summation over Frobenius norm. If `thres` is negative or zero, this algorithm will iterate T times.

Details

This function conducts an alternating optimization algorithm to sparse tensor graphical model. The output is optimal consistent even when `T=1`, see Lyu et al. (2019) for details. There are two termination criteria, `T` and `thres`. Algorithm will be terminated if output in certain iteration change less than `thres`. Otherwise, T iterations will be fully operated.

Value

A length-K list of estimation of precision matrices.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

varcor, biascor, huge

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
# obersavations from tensor normal distribution
out.tlasso = Tlasso.fit(DATA,T=10,lambda.vec = lambda.thm,thres=10)
# terminate by thres
out.tlasso = Tlasso.fit(DATA,T=3,lambda.vec = lambda.thm,thres=0)
# thres=0, iterate 10 times
```

Trnorm *Separable Tensor Normal Distribution*

Description

Generate observations from separable tensor normal distribution.

Usage

```
Trnorm(
  n,
  m.vec,
  mu = array(0, m.vec),
  Sigma.list = NULL,
  type = "Chain",
  sd = 1,
  knn = 4,
  norm.type = 2
)
```

Arguments

n	number of generated observations.
m.vec	vector of tensor mode dimensions, e.g., m.vec=c(m1, m2, m3) for a 3-mode tensor normal distribution.
mu	array of mean for tensor normal distribution with dimension m.vec. Default is zero mean.
Sigma.list	list of covariance matrices in mode sequence. Default is NULL.
type	type of precision matrix, default is 'Chain'. Optional values are 'Chain' for triangle graph and 'Neighbor' for nearest-neighbor graph. Useless if Sigma.list is not NULL.
sd	seed of random number generation, default is 1.
knn	sparsity of precision matrix, i.e., matrix is generated from a knn nearest-neighbor graph. Default is 4. Useless if type='Chain' or Sigma.list is not NULL.
norm.type	normalization method of precision matrix, i.e., $\Omega_{11} = 1$ if norm.type = 1 and $\|\Omega\|_F = 1$ if norm.type = 2. Default value is 2.

Details

This function generates obeservations from separable tensor normal distribution and returns a m1 * ... * mK * n array. If Sigma.list is not given, default distribution is from either triangle graph or nearest-neighbor graph (depends on type).

Value

An array with dimension m_1 * ... * m_K * n.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

ChainOmega, NeighborOmega

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
DATA=Trnorm(n,m.vec,type='Chain')
# a 5*5*5*10 array of oberservation from 5*5*5 separable tensor
#     normal distribtuion with mean zero and
#         precision matrices from triangle graph
```

varcor *Variance Correction of Sample Covariance of Residuals*

Description

Generate variance correction term of sample covariance of residuals described in Lyu et al. (2019).

Usage

```
varcor(data, Omega.list, k = 1)
```

Arguments

`data`	tensor object stored in a m1 * m2 * ... * mK * n array, where n is sample size and mk is dimension of the kth tensor mode.
`Omega.list`	list of precision matrices of tensor, i.e., `Omega.list[[k]]` is precision matrix for the kth tensor mode, $k \in \{1, \ldots, K\}$. Elements in `Omega.list` are true precision matrices or estimation of the true ones, the latter can be output of `Tlasso.fit`.
`k`	index of interested mode, default is 1.

Details

This function computes variance correction term of sample covariance of residuals and is utilized to normalize test statistic into standord normal, see Lyu et al. (2019).

Value

A scalar of variance correction for the kth mode.

Author(s)

Xiang Lyu, Will Wei Sun, Zhaoran Wang, Han Liu, Jian Yang, Guang Cheng.

See Also

`varcor`, `biascor`, `covres`

Examples

```
m.vec = c(5,5,5)  # dimensionality of a tensor
n = 5   # sample size
k=1 # index of interested mode
lambda.thm = 20*c( sqrt(log(m.vec[1])/(n*prod(m.vec))),
                   sqrt(log(m.vec[2])/(n*prod(m.vec))),
                   sqrt(log(m.vec[3])/(n*prod(m.vec))))
DATA=Trnorm(n,m.vec,type='Chain')
```

```
# obersavations from tensor normal distribution
out.tlasso = Tlasso.fit(DATA,T=1,lambda.vec = lambda.thm)
# output is a list of estimation of precision matrices

rho=covres(DATA, out.tlasso, k = k)
# sample covariance of residuals, including diagnoal
varpi2=varcor(DATA, out.tlasso, k = k)
# variance correction term for kth mode's sample covariance of residuals
```

Index