Package: sparseMatEst (via r-universe)

August 27, 2024

Type Package

Title Sparse Matrix Estimation and Inference

Version 1.0.0

Author Adam B Kashlak [aut, cre], Xinyu Zhang [ctb]

Maintainer Adam B Kashlak <kashlak@ualberta.ca>

Description The 'sparseMatEst' package provides functions for estimating sparse covariance and precision matrices with error control. A false positive rate is fixed corresponding to the probability of falsely including a matrix entry in the support of the estimator. It uses the binary search method outlined in Kashlak and Kong (2019) <arXiv:1705.02679> and in Kashlak (2019) <arXiv:1903.10988>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports stats, glasso

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-17 08:20:05 UTC

Contents

sparseMatEst-package *Sparse Matrix Estimation and Inference*

Description

The sparseMatEst library provides functions for estimating sparse covariance and precision matrices with error control.

Details

Given a data matrix, this package contains two main functions used to estimate the covariance matrix and the precision matrix of the data under the assumption of sparsity, that most off-diagonal entries are zero. This is achieved by selecting a false positive rate corresponding to the probability that a true zero entry is falsely chosen to be non-zero by the estimator.

The false positive rate can be treated as an interpretable penalization parameter. Setting this to zero will return a diagonal matrix. Choosing a false positive rate away from zero will allow for the estimator to contain some non-zero off-diagonal entries.

Future updates coming Fall 2019 include inferential tools based on these sparse matrix estimators. These include a variant of linear and quadratic discriminant analysis, fitting a Gaussian mixture assuming sparsity, network clustering algorithm, and a method to fit a random design linear regression model.

Author(s)

Adam B Kashlak <kashlak@ualberta.ca>

References

Kashlak, Adam B., and Linglong Kong. "A concentration inequality based methodology for sparse covariance estimation." arXiv preprint arXiv:1705.02679 (2017).

Kashlak, Adam B. "Non-asymptotic error controlled sparse high dimensional precision matrix estimation." arXiv preprint arXiv:1903.10988 (2019).

genSparseMatrix *Sparse matrix generator*

Description

A way to generate sparse matrices for simulation and testing.

Usage

```
genSparseMatrix(k, rho, type)
```

Arguments

k	dimension parameter
rho	additional parameter
type	type of matrix to generate

Details

genSparseMatrix constructs a sparse matrix to be used for testing and simulation. The type argument determines what type of matrix is produced while k effects the dimension and rho is an additional parameter to effect the output.

For type = 'tri', a (k x k) tridiagonal matrix is returned with off-diagonal entries equal to rho.

For type = 'arm', a (k x k) autoregressive matrix is returned with off-diagonal entries equal to rho^|i-j|.

For type = 'band', a (k x k) banded matrix is returned with rho bands.

For type = 'rand', a (k x k) matrix is returned with rho in (0,1) randomly selected off-diagonal entries set to be non-zero.

For type = 'tree', the adjacency matrix for a k-deep binary tree is returned with off-diagonal entries set to rho. The dimension of this matrix is k(k+1)/2.

For type = 'multi', a (k x k) matrix is returned with rho off-diagonals set to ones.

For type = 'block', a block diagonal matrix is returned with k blocks of size rho.

Value

a matrix corresponding to the arguments chosen.

Author(s)

Adam B Kashlak <kashlak@ualberta.ca>

Examples

```
out = list();
out[[1]]= genSparseMatrix( 20,0.5,"tri" );
out[[2]]= genSparseMatrix( 20,0.5,"arm" );
out[[3]]= genSparseMatrix( 20,5,"band" );
out[[4]]= genSparseMatrix( 20,0.5,"rand" );
out[[5]]= genSparseMatrix( 7,0.5,"tree" );
out[[6]]= genSparseMatrix( 20,5,"multi" );
out[[7]]= genSparseMatrix( 5,4,"block" );

  par(mfrow=c(2,3));
  lab = c("tri","arm","band","rand","tree","multi","block");
  for( i in 2:7  )
    image(out[[i]],main=lab[i]);
```

sparseCov *Sparse covariance matrix estimator with error control*

Description

Given a data matrix, sparseCov estimates the covariance matrix for the data under the assumption that the true covariance matrix is sparse, i.e. that most of the off-diagonal entries are equal to zero.

Usage

```
sparseCov(dat, alf = 0.5, iter = 10, pnrm = Inf, THRSH = "hard")
```

Arguments

dat	nxk data matrix, n observations, k dimensions
alf	false positive rate in [0,1], Default is 0.5
iter	number of iterates, Default is 10
pnrm	norm to use, Default = Inf
THRSH	Type of thresholding used; Takes values: hard, soft, adpt, scad.

Details

The algorithm begins with the empirical covariance estimator as computed by cov. It then iteratively computes covariance estimators with false positive rates of alf, alf^2, and so on until iter estimators have been constructed.

The norm chosen determines the topology on the space of matrices. pnorm defaults to Inf being the operator norm or maximal eigenvalue. This is theoretically justified to work in the references. Other norms could be considered, but their performance is not a strong.

Four thresholding methods are implemented. THRSH defaults to hard thresholding where small matrix entries are set to zero while large entries are not affected. The soft and adpt thresholds shrink all entries towards zero. The scad threshold interpolates between hard and soft thresholding. More details can be found in the references.

Value

a list of arrays containing iter+1 sparse covariance matrices corresponding to false positive rates of 1, alf, alf^2,..., alf^iter. Each list corresponds to one type of thresholding chosen by THRSH.

Author(s)

Adam B Kashlak <kashlak@ualberta.ca>

References

Kashlak, Adam B., and Linglong Kong. "A concentration inequality based methodology for sparse covariance estimation." arXiv preprint arXiv:1705.02679 (2017).

Examples

```
# Generate four sparse covariance matrix estimators
# with false positive rates of 0.5, 0.25, 0.125,
# and 0.0625
n = 30
k = 50
dat = matrix(rnorm(n*k),n,k)
out = sparseCov( dat, alf=0.5, iter=4, THRSH=c("hard","soft") )
  par(mfcol=c(2,2))
  lab = c(1,0.5,0.5^2,0.5^3,0.5^4);
  for( i in 2:5 )
    image( out$hard[,,i]!=0, main=lab[i] )
  for( i in 2:5 )
    image( log(abs(out$hard[,,i] - out$soft[,,i])), main=lab[i] )
```

`sparsePrec` *Sparse precision matrix estimator with error control*

Description

Given a data matrix, `sparsePrec` estimates the precision matrix for the data under the assumption that the true precision matrix is sparse, i.e. that most of the off-diagonal entries are equal to zero.

Usage

```
sparsePrec(dat = 0, prec = 0, alf = 0.5, iter = 10, pnrm = Inf,
  THRSH = "hard", rho = 1, regMeth = "glasso")
```

Arguments

`dat`	nxk data matrix, n observations, k dimensions
`prec`	start with a precision estimator instead of `dat`
`alf`	false positive rate in [0,1], Default is 0.5
`iter`	number of iterates, Default is 10
`pnrm`	norm to use, Default = Inf
`THRSH`	Type of thresholding used; Takes values: hard, soft, adpt, scad.
`rho`	Penalization parameter for the initial estimator, Default is 1
`regMeth`	Type of initial estimator, Default is 'glasso'

Details

The algorithm begins with an initial precision matrix estimator being either `regMeth` = 'glasso' for debiased graphical lasso or `regMeth` = 'ridge' for debiased ridge estimator. It then iteratively computes covariance estimators with false positive rates of `alf`, `alf^2`, and so on until `iter` estimators have been constructed.

If dat is not included, but prec is, then the code runs as before but using the argument prec as the initial precision matrix estimator. In this case, the regMeth is not considered.

The norm chosen determines the topology on the space of matrices. pnorm defaults to Inf being the operator norm or maximal eigenvalue. This is theoretically justified to work in the references. Other norms could be considered, but their performance is not a strong.

Four thresholding methods are implemented. THRSH defaults to hard thresholding where small matrix entries are set to zero while large entries are not affected. The soft and adpt thresholds shrink all entries towards zero. The scad threshold interpolates between hard and soft thresholding. More details can be found in the references.

Value

a list of arrays containing iter+1 sparse precision matrices corresponding to false positive rates of 1, alf, alf^2,..., alf^iter. Each list corresponds to one type of thresholding chosen by THRSH.

Author(s)

Adam B Kashlak <kashlak@ualberta.ca>

References

Kashlak, Adam B. "Non-asymptotic error controlled sparse high dimensional precision matrix estimation." arXiv preprint arXiv:1903.10988 (2019).

Examples

```
# Generate four sparse covariance matrix estimators
# with false positive rates of 0.5, 0.25, 0.125,
# and 0.0625
n = 30
k = 50
dat = matrix(rnorm(n*k),n,k)
out = sparsePrec( dat, alf=0.5, iter=4, THRSH=c("hard","soft") )
  par(mfcol=c(2,2))
  lab = c(1,0.5,0.5^2,0.5^3,0.5^4);
  for( i in 2:5 )
    image( out$hard[,,i]!=0, main=lab[i] )
  for( i in 2:5 )
    image( log(abs(out$hard[,,i] - out$soft[,,i])), main=lab[i] )
```

Index