Package ‘MGGM’

March 22, 2016

Type Package

Title Structural Pursuit Over Multiple Undirected Graphs

Version 1.0

Suggests MASS

Date 2016-03-13

Author Yunzhang Zhu, Xiaotong Shen, Wei Pan, Yiwen Sun

Maintainer Yiwen Sun <sunxx847@umn.edu>

Description
Implement algorithms to recover multiple networks by pursuit of both sparseness and cluster.

License GPL-2

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-03-22 00:22:19

R topics documented:

`MGGM-package` *Structural Pursuit Over Multiple Undirected Graphs*

Description

Implement algorithms to recover multiple networks by pursuit of both sparseness and cluster.

Details

Package:	MGGM
Type:	Package
Title:	Structural Pursuit Over Multiple Undirected Graphs
Version:	1.0
Suggests:	MASS
Date:	2016-03-13
Author:	Yunzhang Zhu, Xiaotong Shen, Wei Pan, Yiwen Sun
Maintainer:	Yiwen Sun <sunxx847@umn.edu>
Description:	Implement algorithms to recover multiple networks by pursuit of both sparseness and cluster.
License:	GPL-2

Index of help topics:

```
MGGM-package            Structural Pursuit Over Multiple Undirected
                        Graphs
MGGM.path               Solution path of multiple Gaussion graph model
```

Author(s)

Yunzhang Zhu, Xiaotong Shen, Wei Pan, Yiwen Sun Maintainer: Yiwen Sun <sunxx847@umn.edu>

References

Yunzhang Zhu, Xiaotong Shen, Wei Pan. Structural pursuit over multiple undirected graphs

---

`MGGM.path` *Solution path of multiple Gaussion graph model*

Description

MGGM.path gives the solution paths for both convex and non-convex version of multiple Gaussion graph model(MGGM). See [1] for detail of MGGM.

Usage

```
MGGM.path(S_bar, nn, Lambda1.vec, Lambda2.vec, graph, tau = 0.01,
MAX_iter = 200, eps_mat = 1e-04)
```

Arguments

`S_bar`	A matrix with dimension $(p, p * L)$. It contains all sample covariance matrices over all L observed stages.
`nn`	A L-length vector, indicating the sample size on each observed stage.

`Lambda1.vec`	A vector containing tuning parameter λ_1, who controls the sparseness of the estimated precision matrices.
`Lambda2.vec`	A vector containing tuning parameter λ_2, who controls the degree of clustering of the estimated precision matrices.
`graph`	A matrix with dimension $(2, E)$, where E is the number of edges. Each column of $graph$ indicates an edge by the indices of connected graphs.
`tau`	The tuning parameter used in truncated L_1 penalty(TLP).
`MAX_iter`	The maximum iteration for DC programming.
`eps_mat`	The convergence criterion of swiping columns.

Value

`sol_nonconvex`	An array with dimension $(p, p{*}L, length(Lambda2.vec), length(Lambda2.vec))$, is the solution path of the non-convex problem, containing all estimated precision matrices over all time and all pairs of tuning parameters.
`sol_convex`	An array with dimension $(p, p{*}L, length(Lambda2.vec), length(Lambda2.vec))$, is the solution path of the convex problem, containing all estimated precision matrices over all time and all pairs of tuning parameters.

aa

Author(s)

Yunzhang Zhu, Xiaotong Shen, Wei Pan, Yiwen Sun

References

[1] Yunzhang Zhu, Xiaotong Shen, Wei Pan. Structural pursuit over multiple undirected graphs

Examples

```
library(MASS)

## generating L true sparse precision and covariance matrices
L <- num_of_matrix <- 4
## two different underlying matrices
L0 <- 2
p <- dim_of_matrix <-  20
n <- 120 #number of observations for each l
nn <- rep(n,L)
MAX_iter <- 200 #max number of iterations

Gene_cov<-function(p){
    sigma <- runif(p-1,0.5,1)
    covmat0 <- diag(1,p)
    for (i in 1:(p-1)){
        for (j in (i+1):p){
            temp <- exp(-sum(sigma[i:(j-1)]/2))
            covmat0[i,j] <- temp
            covmat0[j,i] <- temp
```

```
        }
    }
    return(covmat0)
}
covmat1 <- Gene_cov(p)
covmat_inverse1 <- solve(covmat1)
covmat2 <- Gene_cov(p)
covmat_inverse2 <- solve(covmat2)

## set first L/2 and last L/2 matrices to be the same
covmat0 <- cbind(matrix(rep(covmat1,L/2),p,p*L/2), matrix(rep(covmat2,L/2),p,p*L/2))
covmat_inverse0 <- cbind(matrix(rep(covmat_inverse1,L/2),p,p*L/2),
matrix(rep(covmat_inverse2,L/2),p,p*L/2))


## generating sample covariance matrices S_bar = [S_1, ... S_L]
S_bar <- matrix(0,p,L*p)
for (l in 1:L){
    temp <- mvrnorm(n = nn[l], rep(0,p), covmat0[,((l-1)*p+1):(l*p)])
    S_bar[,((l-1)*p+1):(l*p)] <- crossprod(temp) / nn[l]
}


## matrices generation ends
Lambda1.vec <- log(p)*c(.8, .5, .4, .3, 0.2) #lasso penlaty
Lambda2.vec <- log(p)*c(.1, .08, .06, .05, .04, .03, .0) #grouping penalty
tau <- c(0.01) #thresholding parameter

## generating graphs
graph_complete = matrix(0,2,L*(L-1)/2)
for (l1 in 1:(L-1)){
  graph_complete[,(L*(l1-1)-(l1-1)*l1/2+1):(L*l1-l1*(l1+1)/2)] = rbind(rep(l1,L-l1),(l1+1):L)
}
graph <- graph_complete - 1

sol_path <- MGGM.path(S_bar, nn, Lambda1.vec, Lambda2.vec, graph, tau)
```

Index