Package ‘ScreenClean’

February 19, 2015

Title Screen and clean variable selection procedures
Version 1.0.1
Date 2012-10-30
Author Pengsheng Ji, Jiashun Jin, Qi Zhang
Maintainer Qi Zhang <karlmzhang@gmail.com>
Description Routines for a collection of screen-and-clean type variable selection procedures, including UPS and GS.
Imports MASS, Matrix, quadprog
License GPL (>= 2)
Repository CRAN
Date/Publication 2012-10-30 17:34:49
NeedsCompilation no

R topics documented:

ScreenClean-package ........................................ 2
CleaningStep .................................................. 2
FindAllCG ..................................................... 3
FindCG ........................................................ 4
IterGS ......................................................... 5
PMLE ........................................................... 6
ScreeningStep ............................................... 6
ThresholdGram ............................................... 7
VectorizeBase ............................................... 8

Index 9
ScreenClean-package

*Screen and clean variable selection procedures, including UPS and GS.*

### Description

Routines for a collection of screen-and-clean type variable selection procedures.

### Details

- **Package:** ScreenClean
- **Type:** Package
- **Version:** 1.0.1
- **Date:** 2012-10-30
- **License:** GPL (>= 2)

### Note

In order to use ScreenClean, the data need to be normalized, to make the standard deviation of the noise to be 1, and the l_2 norm of each length n predictor vector to be 1.

### Author(s)

Pengsheng Ji, Jiashun Jin, Qi Zhang

Maintainer: Qi Zhang<qiz19@pitt.edu>

### References


---

CleaningStep

*GC-step of the graphlet screening*

### Description

CleaningStep performs the cleaning step of the graphlet screening
FindAllCG

Usage

CleanStep(survivor, y.tilde, gram, lambda, uu)

Arguments

- **survivor**: the result of the screening step, a logical vector.
- **y.tilde**: $X'y$, where $X$ and $y$ are the predictor matrix and the response vector.
- **gram**: the thresholded sparse gram matrix
- **lambda**: the tuning parameters of the cleaning step, whose optimal choice is tied to the sparse level.
- **uu**: the tuning parameter of the cleaning step; its optimal choice has the intuition of the detected minimal signal strength.

Value

- **beta.gs**: the estimated regression coefficient of the graphlet screening, a numeric vector

See Also

IterGS.ScreeningStep

Examples

```r
# See the demoGs.r
```

<table>
<thead>
<tr>
<th>FindAllCG</th>
<th>Find all the connected subgraphs whose size &lt;= lc</th>
</tr>
</thead>
</table>

Description

FindAllCG uses FindCG iteratively, and lists all the connected subgraphs with no more than lc nodes

Usage

FindAllCG(adjacency.matrix, lc)

Arguments

- **adjacency.matrix**: p by p adjacency matrix of an undirected graph; it must be symmetric.
- **lc**: the maximal size of the connected subgraphs to be listed

Value

- **cg.all**: A list, whose kth component is a matrix with k columns that lists all the connected subgraphs with k nodes.
See Also

FindCG

Examples

```r
require(MASS)
require(Matrix)
p <- 10
Omega <- sparseMatrix(c(1:(p-1),2:p),c(2:p,1:(p-1)),x=1)
cg.all <- FindAllCG(Omega,3)
```

FindCG

*Find the connected subgraphs with a certain number of nodes*

Description

FindCG is used to find all the connected subgraphs with a certain number of nodes.

Usage

```r
FindCG(adjacency.matrix, cg.initial)
```

Arguments

- **adjacency.matrix**
  - p by p adjacency matrix of an undirected graph. It must be symmetric.

- **cg.initial**
  - It could be 1:p or a matrix, whose elements are positive integers from 1 to p. If it is a length p vector, FindCG converts it into a matrix with one column. For a matrix with k columns, FindCG reads its rows as the indices of a collection of connected subgraphs with k nodes.

Value

- **cg.new**
  - If the input is a matrix with k columns and stores the indices of all the size k connected subgraphs, the output is a matrix with k+1 columns storing the indices of all the connected subgraphs with k+1 nodes.

See Also

FindAllCG
IterGS

Examples

```r
require(MASS)
require(Matrix)
p <- 10
Omega <- sparseMatrix(c(1:(p-1),2:p),c(2:p,1:(p-1)),x=1)
cg.2 <- FindCG(Omega,c(1:p))
cg.3 <- FindCG(Omega,cg.2)
```

---

IterGS  
*Iterative graphlet screening procedure*

Description

The iterative graphlet screening procedure, main function of the package.

Usage

```r
IterGS(y.tilde, gram, gram.bias, cg.all, sp, tau, nm, q0=0.1, scale = 1, max.iter = 3, std.thresh = 1.05, beta.initial = NULL)
```

Arguments

- `y.tilde` \( X'y \) where \( X \) and \( y \) are the predictor matrix and the response vector, respectively.
- `gram` the thresholded gram matrix
- `gram.bias` the bias of the thresholded gram matrix
- `cg.all` all the connected cg.all of gram with size no more than nm.
- `sp` the expected sparse level
- `tau` the minimal signal strength to be detected
- `nm` the maximal size of the connected subgraphs considered in the screening step.
- `q0` the minimal screening parameter
- `scale` optional numerical parameter of the screening step. The default is 1
- `max.iter` the maximal number of iterations. The default is 3.
- `std.thresh` the threshold of the std change that stop the loop. The default is 1.05.
- `beta.initial` the initial estimate of beta in reducing the bias. The default is \( uu*\text{sign}(y.tilde)/(\text{abs}(y.tilde)>uu) \).

Value

IterGS returns a list with two elements

- `estimate` The iterative GS estimate of beta
- `n.iter` The number of iterations it takes
**Examples**

```
##See demoIterGs.r
```

**PMLE**

*Penalized MLE procedure used in the cleaning step*

**Description**

Penalized MLE procedure used in the cleaning step, an inner function.

**Usage**

```
PMLE(gram, y, lambda, uu)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gram</td>
<td>the sub gram matrix of the small scale quadratic problem.</td>
</tr>
<tr>
<td>y</td>
<td>the sub-vector of y.tilde</td>
</tr>
<tr>
<td>lambda</td>
<td>the tuning parameter of the cleaning step, tied to the sparse level.</td>
</tr>
<tr>
<td>uu</td>
<td>the tuning parameters of the cleaning step. It has the intuitive interpretation of the minimal signal strength to be detected.</td>
</tr>
</tbody>
</table>

**Value**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>the estimate of the subvector of beta</td>
</tr>
</tbody>
</table>

**See Also**

*CleaningStep*

**ScreeningStep**

*GS-step of the graphlet screening*

**Description**

ScreeningStep performs the cleaning step of the graphlet screening

**Usage**

```
ScreeningStep(y.tilde, gram, cg.all, nm, v, r, q0 = 0.1, scale = 1)
```
ThresholdGram

Arguments

- **y.tilde** $X' y$, where $X$ and $y$ are the predictor matrix and the response vector.
- **gram** the regularized gram matrix
- **cg.all** a list whose kth element is a matrix of k columns. Its rows contain all the connected subgraph with k nodes.
- **nm** the maximal subgraph investigated in the screening step
- **v** an essential tuning parameter of graphlet screening, tied to the sparse level
- **r** an essential tuning parameter of graphlet screening, tied to the signal strength
- **q0** the minimal screening parameter
- **scale** $q(D, F) = q^{max}(D, F) * scale$, default is scale=1

Value

- **survivor** A logical vector, where TRUE means retained as a potential signal.

Note

When nm=1, it is just univariate thresholding, and thus the screening step of UPS.

See Also

- CleaningStep, IterGS

Examples

```
## See the demoGS.r
```

---

**ThresholdGram**

*Thresholds the gram matrix*

Description

Thresholds the gram matrix

Usage

```
ThresholdGram(gram.full, delta = 1/log(dim(gram.full)[1]))
```

Arguments

- **gram.full** the gram matrix before the elementwise thresholding, a p by p symmetric matrix
- **delta** the threshold, the default is 1/log(p)
Value

A list with two elements

gram.sd the thresolded gram matrix, a sparse matrix
gram.bias the difference of the orginal matrix and the threholded matrix

Examples

```r
p <- 10
off.diag <- matrix(runif(p^2), p, p)
omega <- (off.diag + t(off.diag))*0.3
diag(omega) <- 1
omega.omega <- ThresholdGram(omega, 0.3)
omega.omega$gram
omega.omega$gram.bias
```

Description

expresses the number i on the base as a vector, an inner function.

Usage

```r
VectorizeBase(i, base, length)
```

Arguments

- `i` the non-negative number to be converted
- `base` the base to be converted on
- `length` the length of the converted vector

Value

`vector` A vector with the given length, whose elements can be read as the number i with the given base.
Index

*Topic **connected subgraph**
  * FindAllCG, 3
  * FindCG, 4

*Topic **graph**
  * FindAllCG, 3
  * FindCG, 4

CleaningStep, 2, 6, 7

FindAllCG, 3, 4
FindCG, 4, 4

IterGS, 3, 5, 7

PMLE, 6

ScreenClean (ScreenClean-package), 2
ScreenClean-package, 2
ScreeningStep, 3, 6

ThresholdGram, 7

VectorizeBase, 8