Package ‘flare’
August 2, 2019

Type Package
Title Family of Lasso Regression
Version 1.6.0.2
Date 2018-03-21
Author Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan, and Han Liu
Maintainer ORPHANED
Depends R (>= 2.15.0), lattice, MASS, Matrix, igraph
Imports methods
Description Provide the implementation of a family of Lasso variants including Dantzig Selector, LAD Lasso, SQRT Lasso, Lq Lasso for estimating high dimensional sparse linear model. We adopt the alternating direction method of multipliers and convert the original optimization problem into a sequential L1 penalized least square minimization problem, which can be efficiently solved by linearization algorithm. A multi-stage screening approach is adopted for further acceleration. Besides the sparse linear model estimation, we also provide the extension of these Lasso variants to sparse Gaussian graphical model estimation including TIGER and CLIME using either L1 or adaptive penalty. Missing values can be tolerated for Dantzig selector and CLIME. The computation is memory-optimized using the sparse matrix output.
License GPL-2
Repository CRAN
NeedsCompilation yes
Date/Publication 2019-08-02 12:59:57 UTC
X-CRAN-Original-Maintainer Xingguo Li <xingguo.leo@gmail.com>
X-CRAN-Comment Orphaned and corrected on 2019-07-02 as misuse of package= for PACKAGE= was not corrected despite reminder. More corrections on 2019-08-02.

R topics documented:

flare-package ......................................................... 2
coeff.slim ............................................................ 3
The package "flare" provides the implementation of a family of novel regression methods (Lasso, Dantzig Selector, LAD Lasso, SQRT Lasso, Lq Lasso) and their extensions to sparse precision matrix estimation (TIGER and CLIME using L1) in high dimensions. We adopt the alternating direction method of multipliers and convert the original optimization problem into a sequence of L1-penalized least square minimization problems with the linearization method and multi-stage screening of variables. Missing values can be tolerated for Dantzig selector in the design matrix and response vector, and CLIME in the data matrix. The computation is memory-optimized using the sparse matrix output. In addition, we also provide several convenient regularization parameter selection and visualization tools.

### Details

<table>
<thead>
<tr>
<th>Package</th>
<th>flare</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.6.0</td>
</tr>
<tr>
<td>Date</td>
<td>2018-01-03</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
</tbody>
</table>
Author(s)
Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

References

See Also
sugm and slim.

describe(coef.slim)

**coef.slim**

*Extract Model Coefficients for an object with S3 class "slim"*

Description
Extract estimated regression coefficient vectors from the solution path.

Usage
```r
## S3 method for class 'slim'
coef(object, lambda.idx = c(1:3), beta.idx = c(1:3), ...)
```

Arguments
- **object**: An object with S3 class "slim"
- **lambda.idx**: The indices of the regularization parameters in the solution path to be displayed. The default values are c(1:3).
- **beta.idx**: The indices of the estimate regression coefficient vectors in the solution path to be displayed. The default values are c(1:3).
- **...**: Arguments to be passed to methods.
The Bardet-Biedl syndrome Gene expression data from Scheetz et al. (2006)

Description
Gene expression data (20 genes for 120 samples) from the microarray experiments of mammalian-eye tissue samples of Scheetz et al. (2006).

Usage
data(eyedata)

Format
The format is a list containing conatins a matrix and a vector. 1. x - an 120 by 200 matrix, which represents the data of 120 rats with 200 gene probes. 2. y - a 120-dimensional vector of, which represents the expression level of TRIM32 gene.

Details
This data set contains 120 samples with 200 predictors

Author(s)
Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

References

See Also
flare-package.
Examples

```r
data(eyedata)
image(x)
```

---

**Description**

Internal flare functions

**Usage**

```r
sugm.likelihood(Sigma, Omega)
sugm.tracel2(Sigma, Omega)
sugm.cv(obj, loss=c("likelihood", "tracel2"), fold=5)
part.cv(n, fold)
sugm.clime.ladm.scr(Sigma, lambda, nlambda, n, d, maxdf, rho, shrink, prec,
                   max.ite, verbose)
sugm.tiger.ladm.scr(data, n, d, maxdf, rho, lambda, shrink, prec,
                    max.ite, verbose)
slim.lad.ladm.scr.btr(Y, X, lambda, nlambda, n, d, maxdf, rho, max.ite, prec,
                     intercept, verbose)
slim.sqrt.ladm.scr(Y, X, lambda, nlambda, n, d, maxdf, rho, max.ite, prec,
                   intercept, verbose)
slim.dantzig.ladm.scr(Y, X, lambda, nlambda, n, d, maxdf, rho, max.ite, prec,
                     intercept, verbose)
slim.lq.ladm.scr.btr(Y, X, q, lambda, nlambda, n, d, maxdf, rho, max.ite, prec,
                     intercept, verbose)
slim.lasso.ladm.scr(Y, X, lambda, nlambda, n, d, maxdf, max.ite, prec,
                    intercept, verbose)
```

**Arguments**

- **Sigma**: Covariance matrix.
- **Omega**: Inverse covariance matrix.
- **obj**: An object with S3 class returned from "sugm".
- **loss**: Type of loss function for cross validation.
- **fold**: The number of fold for cross validation.
- **n**: The number of observations (sample size).
- **d**: Dimension of data.
- **maxdf**: Maximal degree of freedom.
- **lambda**: Grid of non-negative values for the regularization parameter lambda.
- **nlambda**: The number of the regularization parameter lambda.
### plot.roc

Plot Function for "roc"

#### Description

Plot the ROC curve for an object with S3 class "roc"

#### Usage

```r
## S3 method for class 'roc'
plot(x, ...)
```

#### Arguments

- `x` : An object with S3 class "roc"
- `...` : System reserved (No specific usage)

#### Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>
See Also

sugm.roc, sugm and flare-package.

plot.select  
Plot Function for "select"

Description
Plot the optimal graph by model selection.

Usage
## S3 method for class 'select'
plot(x, ...)

Arguments
x  An object with S3 class "select"
... System reserved (No specific usage)

Author(s)
Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also
sugm and sugm.select

plot.sim  
Plot Function for "sim"

Description
Visualize the covariance matrix, the empirical covariance matrix, the adjacency matrix and the graph pattern of the true graph structure.

Usage
## S3 method for class 'sim'
plot(x, ...)

Arguments
x  An object with S3 class "sim"
... Arguments to be passed to methods.
Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

sugm.generator, sugm and flare-package

Description

Visualize the solution path of regression estimate corresponding to regularization paramters.

Usage

## S3 method for class "slim"
plot(x, ...)

Arguments

x An object with S3 class "slim".

Arguments to be passed to methods.

Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

slim and flare-package.
**plot.sugm**

Plot Function for "sugm"

### Description
Plot sparsity level information and 3 typical sparse graphs from the graph path.

### Usage
```r
## S3 method for class 'sugm'
plot(x, align = FALSE, ...)
```

### Arguments
- **x**: An object with S3 class "sugm"
- **align**: If `align = FALSE`, 3 plotted graphs are aligned
- **...**: Arguments to be passed to methods.

### Author(s)
Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

### See Also
- `sugm` and `flare-package`

---

**predict.slim**

Prediction for an object with S3 class "slim"

### Description
Predicting responses of the given design data.

### Usage
```r
## S3 method for class 'slim'
predict(object, newdata, lambda.idx = c(1:3), Y.pred.idx = c(1:5), ...)
```
predict.slim

Arguments

object          An object with S3 class "slim"
newdata        An optional data frame in which to look for variables with which to predict. If
                omitted, the training data of the are used.
lambda.idx     The indices of the regularization parameters in the solution path to be displayed.
                The default values are c(1:3).
Y.pred.idx     The indices of the predicted response vectors in the solution path to be displayed.
                The default values are c(1:5).
...            Arguments to be passed to methods.

Details

predict.slim produces predicted values of the responses of the newdata from the estimated beta
values in the object, i.e.
\[ \hat{Y} = \hat{\beta}_0 + X_{new}\hat{\beta}. \]

Value

Y.pred        The predicted response vectors based on the estimated models.

Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

slim and flare-package.

Examples

```r
## load library
library(flare)
## generate data
set.seed(123)
 n = 100
d = 200
d1 = 10
rho0 = 0.3
lambda = c(3:1)*sqrt(log(d)/n)
Sigma = matrix(0,nrow=d,ncol=d)
Sigma[1:d1,1:d1] = rho0
diag(Sigma) = 1
mu = rep(0,d)
X = mvrnorm(n=2*n,mu=mu,Sigma=Sigma)
X.fit = X[1:n,
X.pred = X[(n+1):(2*n)],
```
eps = rt(n=n,df=n-1)
beta = c(rep(sqrt(1/3),3),rep(0,d-3))
Y.fit = X.fit%*%beta+eps

## Regression with "dantzig".
out=slim(X=X.fit,Y=Y.fit,lambda=lambda,method = "lq",q=1)

## Display results
Y=predict(out,X.pred)

---

**print.roc**

*Print Function for an object with S3 class "roc"*

### Description

Print the information about true positive rates, false positive rates, the area under curve and maximum F1 score

### Usage

```r
## S3 method for class 'roc'
print(x, ...)
```

### Arguments

- `x` An object with S3 class "roc"
- `...` Arguments to be passed to methods.

### Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu

Maintainer: Xingguo Li <xingguo.leo@gmail.com>

### See Also

- `sugm.roc`
- `sugm`
- `flare-package`
print.select  
*Print Function for an object with S3 class "select"*

**Description**

Print the information about the model usage, graph dimension, model selection criterion, sparsity level of the optimal graph.

**Usage**

```r
## S3 method for class 'select'
print(x, ...)
```

**Arguments**

- `x`  
  An object with S3 class "select"

- `...`  
  Arguments to be passed to methods.

**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**See Also**

- `sugm.select`, `sugm` and `flare-package`

---

print.sim  
*Print Function for an object with S3 class "sim"*

**Description**

Print the information about the sample size, the dimension, the pattern and sparsity of the true graph structure.

**Usage**

```r
## S3 method for class 'sim'
print(x, ...)
```

**Arguments**

- `x`  
  An object with S3 class "sim".

- `...`  
  Arguments to be passed to methods.


**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**See Also**

`sutm` and `sutm.generator`

---

**print.slim**

*Print Function for an object with S3 class "slim"*

**Description**

Print a summary of the information about an object with S3 class "slim".

**Usage**

```r
## S3 method for class 'slim'
print(x, ...)
```

**Arguments**

- `x` An object with S3 class "slim".
- `...` Arguments to be passed to methods.

**Details**

This call simply outlines the options used for computing a slim object.

**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**See Also**

`sutm` and `flare-package`. 
**print.sugm**

*Print Function for an object with S3 class "sugm"*

**Description**

Print a summary of the information about an object with S3 class "sugm".

**Usage**

```r
## S3 method for class 'sugm'
print(x, ...)
```

**Arguments**

- `x` An object with S3 class "sugm".
- `...` Arguments to be passed to methods.

**Details**

This call simply outlines the options used for computing a sugm object.

**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu

Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**See Also**

`sugm` and `flare-package`.

---

**slim**

*Sparse Linear Regression using Nonsmooth Loss Functions and L1 Regularization*

**Description**

The function "slim" implements a family of Lasso variants for estimating high dimensional sparse linear models including Dantzig Selector, LAD Lasso, SQRT Lasso, Lq Lasso for estimating high dimensional sparse linear model. We adopt the alternating direction method of multipliers (ADMM) and convert the original optimization problem into a sequential L1-penalized least square minimization problem, which can be efficiently solved by combining the linearization and multi-stage screening of variables. Missing values can be tolerated for Dantzig selector in the design matrix and response vector.
Usage

slim(X, Y, lambda = NULL, nlambda = NULL,
    lambda.min.value = NULL, lambda.min.ratio = NULL,
    rho = 1, method="lq", q = 2, res.sd = FALSE,
    prec = 1e-5, max.ite = 1e5, verbose = TRUE)

Arguments

Y    The $n$-dimensional response vector.
X    The $n$ by $d$ design matrix. $d \geq 2$ is required.
lambda    A sequence of decreasing positive numbers to control the regularization. Typical usage is to leave the input lambda = NULL and have the program compute its own lambda sequence based on nlambda and lambda.min.ratio. Users can also specify a sequence to override this. Default value is from lambda.max to lambda.min.ratio*lambda.max. For Lq regression, the default value of lambda.max is $\pi \sqrt{\log(d)/n}$. For Dantzig selector, the default value of lambda.max is the minimum regularization parameter, which yields an all-zero estimates.
nlambda   The number of values used in lambda. Default value is 5.
lambda.min.value   The smallest value for lambda, as a fraction of the upperbound (lambda.max) of the regularization parameter. The program can automatically generate lambda as a sequence of length = nlambda starting from lambda.max to lambda.min.ratio*lambda.max in log scale. The default value is $\log(d)/n$ for for Dantzig selector 0.3*lambda.max for Lq Lasso.
lambda.min.ratio   The smallest ratio of the value for lambda. The default value is 0.3 for Lq Lasso and 0.5 for Dantzig selector.
rho   The penalty parameter used in ADMM. The default value is $\sqrt{d}$.
method   Dantzig selector is applied if method = "dantzig" and $L_q$ Lasso is applied if method = "lq". Standard Lasso is provided if method = "lasso". The default value is "lq".
q   The loss function used in Lq Lasso. It is only applicable when method = "lq" and must be in [1,2]. The default value is 2.
res.sd   Flag of whether the response variables are standardized. The default value is FALSE.
prec   Stopping criterion. The default value is $1e-5$.
max.ite    The iteration limit. The default value is $1e5$.
verbose   Tracing information printing is disabled if verbose = FALSE. The default value is TRUE.

Details

Standard Lasso

$$\min \frac{1}{2n} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1$$
Dantzig selector solves the following optimization problem

\[
\min ||\beta||_1, \quad \text{s.t. } ||X'(Y - X\beta)||_\infty < \lambda
\]

\(L_q\) loss Lasso solves the following optimization problem

\[
\min n^{-\frac{q}{2}}||Y - X\beta||_q + \lambda||\beta||_1
\]

where \(1 \leq q \leq 2\). \(L_q\) Lasso is equivalent to LAD Lasso and SQR Lasso when \(q = 1\) and \(q = 2\) respectively.

Value

An object with S3 class "slim" is returned:

- **beta**
  - A matrix of regression estimates whose columns correspond to regularization parameters.
- **intercept**
  - The value of intercepts corresponding to regularization parameters.
- **Y**
  - The value of \(Y\) used in the program.
- **X**
  - The value of \(X\) used in the program.
- **lambda**
  - The sequence of regularization parameters \(\lambda\) used in the program.
- **nlambda**
  - The number of values used in \(\lambda\).
- **method**
  - The method from the input.
- **sparsity**
  - The sparsity levels of the solution path.
- **ite**
  - A list of vectors where \(\text{ite}[1]\) is the number of external iteration and \(\text{ite}[2]\) is the number of internal iteration with the i-th entry corresponding to the i-th regularization parameter.
- **verbose**
  - The verbose from the input.

Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

References

See Also

 flare-package, print.slim, plot.slim, coef.slim and predict.slim.

Examples

```r
## load library
library(flare)
## generate data
n = 50
d = 100
X = matrix(rnorm(n*d), n, d)
beta = c(3,2,0,1.5,rep(0,d-4))
eps = rnorm(n)
Y = X%*%beta + eps
nlamb = 5
ratio = 0.3

## Regression with "dantzig", general "lq" and "lasso" respectively
out1 = slim(X=X,Y=Y,nlamb=nlamb,lambda.min.ratio=ratio,method="dantzig")
out2 = slim(X=X,Y=Y,nlamb=nlamb,lambda.min.ratio=ratio,method="lq",q=1)
out3 = slim(X=X,Y=Y,nlamb=nlamb,lambda.min.ratio=ratio,method="lq",q=1.5)
out4 = slim(X=X,Y=Y,nlamb=nlamb,lambda.min.ratio=ratio,method="lq",q=2)
out5 = slim(X=X,Y=Y,nlamb=nlamb,lambda.min.ratio=ratio,method="lasso")

## Display results
print(out4)
plot(out4)
coef(out4)
```

sugm

**High-dimensional Sparse Undirected Graphical Models.**

Description

The function "sugm" estimates sparse undirected graphical models, i.e. Gaussian precision matrix, in high dimensions. We adopt two estimation procedures based on column by column regression scheme: (1) Tuning-Insensitive Graph Estimation and Regression based on square root Lasso (tiger); (2) The Constrained L1 Minimization for Sparse Precision Matrix Estimation using either L1 penalty (clime). The optimization algorithm for all three methods are implemented based on the alternating direction method of multipliers (ADMM) with the linearization method and multi-stage screening of variables. Missing values can be tolerated for CLIME in the data matrix. The computation is memory-optimized using the sparse matrix output.

Usage

```
sugm(data, lambda = NULL, nlamb = NULL, lambda.min.ratio = NULL, rho = NULL, method = "tiger", sym = "or", shrink=NULL, prec = 1e-4, max.ite = 1e4, standardize = FALSE, perturb = TRUE, verbose = TRUE)
```
Arguments

data
There are 2 options for "clime": (1) data is an n by d data matrix (2) a d by d sample covariance matrix. The program automatically identifies the input matrix by checking the symmetry. (n is the sample size and d is the dimension). For “tiger”, covariance input is not supported and d\geq3 is required. For “clime”, d\geq2 is required.

lambda
A sequence of decreasing positive numbers to control the regularization. Typical usage is to leave the input lambda = NULL and have the program compute its own lambda sequence based on nlambdas and lambda.min.ratio. Users can also specify a sequence to override this. Default value is from lambda.max to lambda.min.ratio*lambda.max. For "tiger", the default value of lambda.max is \( \pi \sqrt{\log(d)/n} \). For "clime", the default value of lambda.max is the minimum regularization parameter, which yields an all-zero off-diagonal estimates.

nlambdas
The number of values used in lambda. Default value is 5.

lambda.min.ratio
The smallest value for lambda, as a fraction of the upperbound (lambda.max) of the regularization parameter. The program can automatically generate lambda as a sequence of length = nlambdas starting from lambda.max to lambda.min.ratio*lambda.max in log scale. The default value is 0.25 for "tiger" and 0.5 for "clime".

rho
Penalty parameter used in the optimization algorithm for clime. The default value is \( \sqrt{d} \).

method
"tiger" is applied if method = "tiger" and "clime" is applied if method="clime". Default value is “tiger”.

sym
Symmetrization of output graphs. If sym = "and", the edge between node i and node j is selected ONLY when both node i and node j are selected as neighbors for each other. If sym = "or", the edge is selected when either node i or node j is selected as the neighbor for each other. The default value is "or".

shrink
Shrinkage of regularization parameter based on precision of estimation. The default value is 1.5 if method = "clime" and the default value is 0 if method="tiger".

prec
Stopping criterion. The default value is 1e-4.

max.ite
The iteration limit. The default value is 1e4.

standardize
Variables are standardized to have mean zero and unit standard deviation if standardize = TRUE. The default value is FALSE.

perturb
The diagonal of Sigma is added by a positive value to guarantee that Sigma is positive definite if perturb = TRUE. User can specify a numeric value for perturb. The default value is perturb = TRUE.

verbose
Tracing information printing is disabled if verbose = FALSE. The default value is TRUE.

Details

CLIME solves the following minimization problem

\[
\min ||\Omega||_1 \quad \text{s.t.} \quad ||S\Omega - I||_\infty \leq \lambda,
\]
where $|\cdot|_1$ and $|\cdot|_\infty$ are element-wise 1-norm and $\infty$-norm respectively.

"tiger" solves the following minimization problem

$$\min |X - XB|_{2,1} + \lambda |B|_1 \quad \text{s.t. } B_{jj} = 0,$$

where $|\cdot|_1$ and $|\cdot|_{2,1}$ are element-wise 1-norm and $L_{2,1}$-norm respectively.

**Value**

An object with S3 class "sugm" is returned:

- **data**: The $n$ by $d$ data matrix or $d$ by $d$ sample covariance matrix from the input.
- **cov.input**: An indicator of the sample covariance.
- **lambda**: The sequence of regularization parameters used in the program.
- **nlambda**: The number of values used in lambda.
- **icov**: A list of $d$ by $d$ precision matrices corresponding to regularization parameters.
- **sym**: The sym from the input.
- **method**: The method from the input.
- **path**: A list of $d$ by $d$ adjacency matrices of estimated graphs as a graph path corresponding to lambda.
- **sparsity**: The sparsity levels of the graph path.
- **ite**: If method = "clime", it is a list of two matrices where ite[[1]] is the number of external iterations and ite[[2]] is the number of internal iterations with the entry of (i,j) as the number of iteration of i-th column and j-th lambda. If method="tiger", it is a matrix of iteration with the entry of (i,j) as the number of iteration of i-th column and j-th lambda.
- **df**: It is a $d$ by nlambda matrix. Each row contains the number of nonzero coefficients along the lasso solution path.
- **standardize**: The standardize from the input.
- **perturb**: The perturb from the input.
- **verbose**: The verbose from the input.

**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu

Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**References**

sugm.generator

Data generator for sparse undirected graph estimation.

Description

Implements the data generation from multivariate normal distributions with different graph structures, including "random", "hub", "cluster", "band", and "scale-free".

Usage

sugm.generator(n = 200, d = 50, graph = "random", v = NULL, u = NULL, g = NULL, prob = NULL, seed = NULL, vis = FALSE, verbose = TRUE)

Arguments

n
The number of observations (sample size). The default value is 200.

d
The number of variables (dimension). For "hub" and "cluster", \(d \geq 4\) is required. For "random", "band" and "scale-free", \(d \geq 3\) is required. The default value is 50.

graph
The graph structure with 5 options: "random", "hub", "cluster", "band", and "scale-free".

Examples

```r
## load package required
library(flare)

## generating data
n = 50
d = 50
D = sugm.generator(n=n, d=d, graph="band", g=1)
plot(D)

## sparse precision matrix estimation with method "clime"
out1 = sugm(D$data, method = "clime")
plot(out1)
sugm.plot(out1$path[[4]])

## sparse precision matrix estimation with method "tiger"
out2 = sugm(D$data, method = "tiger")
plot(out2)
sugm.plot(out2$path[[5]])
```
The off-diagonal elements of the precision matrix, controlling the magnitude of partial correlations with \( u \). The default value is 0.3.

\( u \) A positive number being added to the diagonal elements of the precision matrix, to control the magnitude of partial correlations. The default value is 0.1.

\( g \) For "cluster" or "hub" graph, \( g \) is the number of hubs or clusters in the graph. The default value is about \( d/20 \) if \( d \geq 40 \) and 2 if \( d < 40 \). For "band" graph, \( g \) is the bandwidth and the default value is 1. NOT applicable to "random" graph.

\( \text{prob} \) For "random" graph, it is the probability that a pair of nodes has an edge. The default value is 3/\( d \). For "cluster" graph, it is the probability that a pair of nodes has an edge in each cluster. The default value is 6*\( g/d \) if \( d/g \leq 30 \) and 0.3 if \( d/g > 30 \). NOT applicable to "hub", "band", and "scale-free" graphs.

\( \text{seed} \) Set seed for data generation. The default value is 1.

\( \text{vis} \) Visualize the adjacency matrix of the true graph structure, the graph pattern, the covariance matrix and the empirical covariance matrix. The default value is FALSE.

\( \text{verbose} \) If \( \text{verbose} = \text{FALSE} \), tracing information printing is disabled. The default value is TRUE.

**Details**

Given the adjacency matrix \( \theta \), the graph patterns are generated as below:

(I) "random": Each pair of off-diagonal elements are randomly set \( \theta[i,j]=\theta[j,i]=1 \) for \( i\neq j \) with probability \( \text{prob} \), and 0 other wise. It results in about \( d*(d-1)*\text{prob}/2 \) edges in the graph.

(II) "hub": The row/columns are evenly partitioned into \( g \) disjoint groups. Each group is associated with a "center" row \( i \) in that group. Each pair of off-diagonal elements are set \( \theta[i,j]=\theta[j,i]=1 \) for \( i\neq j \) if \( j \) also belongs to the same group as \( i \) and 0 otherwise. It results in \( d-g \) edges in the graph.

(III) "cluster": The row/columns are evenly partitioned into \( g \) disjoint groups. Each pair of off-diagonal elements are set \( \theta[i,j]=\theta[j,i]=1 \) for \( i\neq j \) with the probability \( \text{prob} \) if both \( i \) and \( j \) belong to the same group, and 0 other wise. It results in about \( g*(d/g)*(d/g-1)*\text{prob}/2 \) edges in the graph.

(IV) "band": The off-diagonal elements are set to be \( \theta[i,j]=1 \) if \( 1<=|i-j|<=g \) and 0 other wise. It results in \( (2d-1-g)*g/2 \) edges in the graph.

(V) "scale-free": The graph is generated using B-A algorithm. The initial graph has two connected nodes and each new node is connected to only one node in the existing graph with the probability proportional to the degree of the each node in the existing graph. It results in \( d \) edges in the graph.

The adjacency matrix \( \theta \) has all diagonal elements equal to 0. To obtain a positive definite covariance matrix, the smallest eigenvalue of \( \theta*v \) (denoted by \( e \)) is computed. Then we set the
covariance matrix equal to \( \text{cov2cor}(\text{solve}(\theta v + (|e|+0.1+u)*I)) \) to generate multivariate normal data.

**Value**

An object with S3 class "sim" is returned:

- `data` The \( n \) by \( d \) matrix for the generated data
- `sigma` The covariance matrix for the generated data
- `omega` The precision matrix for the generated data
- `sigmahat` The empirical covariance matrix for the generated data
- `theta` The adjacency matrix of true graph structure (in sparse matrix representation) for the generated data

**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**See Also**

`flare` and `flare-package`

**Examples**

```r
## load package required
library(flare)

## band graph with bandwidth 3
L = sugm.generator(graph = "band", g = 3)
plot(L)

## random sparse graph
L = sugm.generator(vis = TRUE)

## random dense graph
L = sugm.generator(prob = 0.5, vis = TRUE)

## hub graph with 6 hubs
L = sugm.generator(graph = "hub", g = 6, vis = TRUE)

## cluster graph with 8 clusters
L = sugm.generator(graph = "cluster", g = 8, vis = TRUE)

## scale-free graphs
L = sugm.generator(graph="scale-free", vis = TRUE)
```
**sugm.plot**

*Graph visualization for an object with S3 class "sugm"

**Description**

Implements the graph visualization using adjacency matrix. It can automatic organize 2D embedding layout.

**Usage**

```r
sugm.plot(G, epsflag = FALSE, graph.name = "default", cur.num = 1, location)
```

**Arguments**

- **G**
  The adjacency matrix corresponding to the graph.
- **epsflag**
  If `epsflag = TRUE`, save the plot as an eps file in the target directory. The default value is `FALSE`.
- **graph.name**
  The name of the output eps files. The default value is "default".
- **cur.num**
  The number of plots saved as eps files. Only applicable when `epsflag = TRUE`. The default value is 1.
- **location**
  Target directory. The default value is the current working directory.

**Details**

The user can change `cur.num` to plot several figures and select the best one. The implementation is based on the popular package "igraph".

**Author(s)**

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

**See Also**

`flare` and `flare-package`

**Examples**

```r
## load package required
library(flare)

## visualize the hub graph
L = sugm.generator(graph = "hub")
sugm.plot(L$theta)

## visualize the band graph
```
L = sugm.generator(graph = "band", g=5)
sugm.plot(L$theta)

## visualize the cluster graph
L = sugm.generator(graph = "cluster")
sugm.plot(L$theta)

## Not run:
# show working directory
getwd()

# plot 5 graphs and save the plots as eps files in the working directory
sugm.plot(L$theta, epsflag = TRUE, cur.num = 5)

## End(Not run)

---

**sugm.roc**

*Draw ROC Curve for an object with S3 class "sugm"*

**Description**

Draws ROC curve for a graph path according to the true graph structure.

**Usage**

```
sugm.roc(path, theta, verbose = TRUE)
```

**Arguments**

- `path`: A graph path.
- `theta`: The true graph structure.
- `verbose`: If `verbose = FALSE`, tracing information printing is disabled. The default value is `TRUE`.

**Details**

To avoid the horizontal oscillation, false positive rates is automatically sorted in the ascent order and true positive rates also follow the same order.

**Value**

An object with S3 class "roc" is returned:

- `F1`: The F1 scores along the graph path.
- `tp`: The true positive rates along the graph path.
- `fp`: The false positive rates along the graph paths.
- `AUC`: Area under the ROC curve.
Note

For a lasso regression, the number of nonzero coefficients is at most n-1. If d>>n, even when regularization parameter is very small, the estimated graph may still be sparse. In this case, the AUC may not be a good choice to evaluate the performance.

Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

sugm and flare-package

Examples

```r
## load package required
library(flare)

#generate data
L = sugm.generator(d = 30, graph = "random", prob = 0.1)
out1 = sugm(L$data, lambda=10*(seq(log10(.4), log10(0.03), length.out=20)))

#draw ROC curve
Z1 = sugm.roc(out1$path,L$theta)

#Maximum F1 score
max(Z1$F1)
```

---

**sugm.select**

*Model selection for high-dimensional undirected graphical models*

**Description**

Implements the regularization parameter selection for high dimensional undirected graphical models. The optional approaches are stability approach to regularization selection (stars) and cross validation selection (cv).

**Usage**

```r
sugm.select(est, criterion = "stars", stars.subsample.ratio = NULL,
             stars.thresh = 0.1, rep.num = 20, fold = 5,
             loss="likelihood", verbose = TRUE)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>est</td>
<td>An object with S3 class &quot;sugm&quot;</td>
</tr>
<tr>
<td>criterion</td>
<td>Model selection criterion. &quot;stars&quot; and &quot;cv&quot; are available for both graph estimation methods. The default value is &quot;stars&quot;.</td>
</tr>
<tr>
<td>stars.subsample.ratio</td>
<td>The subsampling ratio. The default value is $10*\sqrt{n}/n$ when $n&gt;144$ and 0.8 when $n\leq144$, where $n$ is the sample size. Only applicable when criterion = &quot;stars&quot;.</td>
</tr>
<tr>
<td>stars.thresh</td>
<td>The variability threshold in stars. The default value is 0.1. Only applicable when criterion = &quot;stars&quot;.</td>
</tr>
<tr>
<td>rep.num</td>
<td>The number of subsamplings. The default value is 20.</td>
</tr>
<tr>
<td>fold</td>
<td>The number of folds used in cross validation. The default value is 5. Only applicable when criterion = &quot;cv&quot;.</td>
</tr>
<tr>
<td>loss</td>
<td>Loss to be used in cross validation. Two losses are available: &quot;likelihood&quot; and &quot;traceL2&quot;. Default &quot;likelihood&quot;. Only applicable when criterion = &quot;cv&quot;.</td>
</tr>
<tr>
<td>verbose</td>
<td>If verbose = FALSE, tracing information printing is disabled. The default value is TRUE.</td>
</tr>
</tbody>
</table>

Details

Stability approach to regularization selection (stars) is a natural way to select optimal regularization parameter for all three estimation methods. It selects the optimal graph by variability of subsamplings and tends to over-select edges in Gaussian graphical models. Besides selecting the regularization parameters, stars can also provide an additional estimated graph by merging the corresponding subsampled graphs using the frequency counts. The K-fold cross validation is also provided for selecting the parameter lambda, and two loss functions are adopted as follow:

$$likelihood : Tr(\Sigma \Omega) - \log |\Omega|$$

$$traceL2 : Tr(diag(\Sigma \Omega - I)^2).$$

Value

An object with S3 class "select" is returned:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>refit</td>
<td>The optimal graph selected from the graph path</td>
</tr>
<tr>
<td>opt.icov</td>
<td>The optimal precision matrix selected.</td>
</tr>
<tr>
<td>merge</td>
<td>The graph path estimated by merging the subsampling paths. Only applicable when the input criterion = &quot;stars&quot;.</td>
</tr>
<tr>
<td>variability</td>
<td>The variability along the subsampling paths. Only applicable when the input criterion = &quot;stars&quot;.</td>
</tr>
<tr>
<td>opt.index</td>
<td>The index of the selected regularization parameter.</td>
</tr>
<tr>
<td>opt.lambda</td>
<td>The selected regularization/thresholding parameter.</td>
</tr>
<tr>
<td>opt.sparsity</td>
<td>The sparsity level of &quot;refit&quot;.</td>
</tr>
</tbody>
</table>

and anything else included in the input est
Note

The model selection is NOT available when the data input is the sample covariance matrix.

Author(s)

Xingguo Li, Tuo Zhao, Lie Wang, Xiaoming Yuan and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

References


See Also

`sugm` and `flare-package`.

Examples

```r
## load package required
library(flare)

#generate data
L = sugm.generator(d = 10, graph = "hub")
out1 = sugm(L$data)

#model selection using stars
#out1.select1 = sugm.select(out1, criterion = "stars", stars.thresh = 0.1)
#plot(out1.select1)

#model selection using cross validation
out1.select2 = sugm.select(out1, criterion = "cv")
plot(out1.select2)
```
Index

coeff.slim, 3, 17

eyedata, 4

flare, 22, 23
flare (flare-package), 2
flare-internal, 5
flare-package, 2

dpart.cv (flare-internal), 5
plot.roc, 6, 20
plot.select, 7, 20
plot.sim, 7, 20
plot.slim, 8, 17
plot.sugm, 9, 20
predict.slim, 9, 17
print.roc, 11, 20
print.select, 12, 20
print.sim, 12, 20
print.slim, 13, 17
print.sugm, 14, 20

slim, 3, 4, 6, 8, 10, 13, 14
slim.dantzig.ladm.scr (flare-internal), 5
slim.lad.ladm.scr (flare-internal), 5
slim.lasso.ladm.scr (flare-internal), 5
slim.lq.ladm.scr (flare-internal), 5
slim.sqrt.ladm.scr (flare-internal), 5

sugm, 3, 6–9, 11–14, 17, 25, 27
sugm.clime.ladm.scr (flare-internal), 5
sugm.cv (flare-internal), 5
sugm.generation, 8, 13, 20, 20
sugm.likelihood (flare-internal), 5
sugm.plot, 20, 23
sugm.roc, 7, 11, 20, 24
sugm.select, 7, 12, 20, 25
sugm.tiger.ladm.scr (flare-internal), 5
sugm.trace12 (flare-internal), 5

x (eyedata), 4

y (eyedata), 4