Package ‘picasso’

February 20, 2015

Type Package
Title Pathwise Calibrated Sparse Shooting Algorithm
Version 0.3.0
Date 2014-10-18
Author Xingguo Li, Tuo Zhao and Han Liu
Maintainer Xingguo Li <xingguo.leo@gmail.com>
Depends R (>= 2.15.0), lattice, igraph, MASS, Matrix
Description Implement a new family of efficient algorithms, called PathwIse Cali-

brAted Sparse Shooting AlgOrithm, for a variety of sparse learning problems, including Sparse Linear Regression, Sparse Logistic Regression, Sparse Column Inverse Operator and Sparse Multivariate Regression. Different types of active set identifica-
tion schemes are implemented, such as cyclic search, greedy search, stochastic search and proximal gradient search. Besides, the package provides the choices between con-

vex (L1 norm) and non-convex (MCP and SCAD) regularizations. Moreover, group regularization, such as group Lasso, group MCP and group SCAD, are also implemented for Sparse Linear Regression, Sparse Logistic Regression and Sparse Multivariate Regression.

License GPL-2
Repository CRAN
NeedsCompilation yes
Date/Publication 2014-10-19 01:37:15

R topics documented:

picasso-package .............................................................. 2
coe.l.asso ................................................................. 3
coe.logit ................................................................. 3
eyedata ................................................................. 4
picasso ................................................................. 5
plot.l.asso .............................................................. 10
plot.logit .............................................................. 11
plot.scio ............................................................... 11
plot.sim ............................................................... 12
Description

Provides the implementation of a new family of efficient algorithms, called Pathwise Calibrated Sparse Shooting Algorithm (picasso), for a variety of sparse learning problems, including Sparse Linear Regression, Sparse Logistic Regression, Sparse Column Inverse Operator and Sparse Multivariate Regression. Different types of active set identification schemes are implemented, such as cyclic search, greedy search, stochastic search and proximal gradient search. Besides, the package provides the choices between convex (L1 norm) and non-convex (MCP and SCAD) regularizations. Moreover, group regularization, such as group Lasso, group MCP and group SCAD, are also implemented for Sparse Linear Regression, Sparse Logistic Regression and Sparse Multivariate Regression.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>picasso</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>0.3.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2014-10-18</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
</tbody>
</table>

Author(s)

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

See Also

picasso and scio.generator.
Extract Model Coefficients for an object with S3 class "lasso"

Description
Extract estimated regression coefficient vectors from the solution path.

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

Arguments
- object: An object with S3 class "lasso"
- 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.

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

See Also
picasso and picasso-package.

Extract Model Coefficients for an object with S3 class "logit"

Description
Extract estimated regression coefficient vectors from the solution path.

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

Arguments
- object: An object with S3 class "logit"
- 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.
Arguments

object  An object with S3 class "logit"
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.

Author(s)

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

See Also

picasso and picasso-package.

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 and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>
References


See Also

`picasso-package`.

Examples

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

---

**picasso**

*Pathwise Calibrated Sparse Shooting Algorithm*

**Description**

The function "picasso" implements the user interface.

**Usage**

```r
picasso(X, Y, lambda = NULL, nlambda = NULL, lambda.min.ratio = NULL,
family = "gaussian", method="l1", alg = "cyclic", gamma = 3,
sym = "or", standardize = FALSE, perturb = TRUE, design.sd = TRUE,
res.sd = FALSE, gr = NULL, gr.d = NULL, gr.size = NULL, max.act.in = 3,
truncation = 0.01, prec = 1e-4, max.ite = 1e4, verbose = TRUE)
```

**Arguments**

- `X` For sparse linear regression and sparse logistic regression, `X` is an `n` by `d` design matrix. For sparse multivariate regression, `X` is an `n` by `d` design matrix. For sparse inverse column operation, there are 2 options when `family = "nnp"`:
  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).

- `Y` For sparse linear regression and sparse logistic regression, `Y` is an `n` dimensional response vector. For sparse multivariate regression, `Y` is an `n` by `m` response matrix. For sparse inverse column operation, no input for `Y` is needed.

- `lambda` A sequence of decresing positive value 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 from `lambda.max` to `lambda.min.ratio*lambda.max`. For Lq regression, the default value of `lambda.max` is $\pi \sqrt{\log(d)/n}$. 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.ratio

The smallest value for lambda, as a fraction of the upper bound (MAX) of the
regularization parameter. The program can automatically generate lambda as a
sequence of length = nlambda starting from MAX to lambda.min.ratio*MAX in
log scale. The default value is 0.25.

family

Options for model. Sparse linear regression and sparse multivariate regres-
sion is applied if family = "gaussian", sparse logistic regression is ap-
plied if family = "logit" and sparse column inverse operation is applied
if family = "nnpn". The default value is "gaussian".

method

Options for regularization. Lasso is applied if method = "l1", MCP is applied
if method = "mcp" and SCAD Lasso is applied if method = "scad". For
sparse linear regression and sparse logistic regression, group lasso is applied if
method = "group", group MCP is applied if method = "group.mcp" and
group SCAD is applied if method = "group.scad". The default value is "l1".

alg

Options for active set identification. Cyclic search is applied if alg = "cyclic",
greedy search is applied if alg = "greedy", proximal gradient based search is
applied if alg = "prox" and stochastic search is applied if alg = "stoc". The
default value is "cyclic".

gamma

Parameter for MCP and SCAD. The default value is 3.

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".

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.

design.sd

Flag of whether the design matrix is standardized. The default value is TRUE.

res.sd

Flag of whether the response variables are standardized. The default value is
FALSE.

gr

A list which defines the grouping of the variables. The default setting is grouping
every two variables if method = "glasso".

gr.d

The number of variables in each group. The default value is gr.d = "2".

gr.size

A vector of the number of variables in each group. The default value is a vector
of 2 of size d/2.

max.act.in

The maximum number of active variables to add into the active set when alg = "cyclic".
The default value is 3.

truncation

The value of (1 + truncation) * lambda for active set update when alg = "cyclic".
The default value is 0.

prec

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

max.ite

The iteration limit. The default value is 1e4.

verbose

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

For sparse linear regression,

$$\min_{\beta} \frac{1}{2n} \|Y - X\beta\|^2_2 + \lambda R(\beta),$$

where $R(\beta)$ can be $\ell_1$ norm, MCP, SCAD, group $\ell_1$, group MCP or group SCAD.

For sparse multivariate regression,

$$\min_{\beta} \frac{1}{2nm} \|Y - X\beta\|^2_p + \lambda R(\beta),$$

where $R(\beta)$ is $\ell_{12}$ norm of matrix, group MCP or group SCAD.

For sparse logistic regression,

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} \left( \log(1 + e^{x_i^T \beta}) - y_i x_i^T \beta \right) + \lambda R(\beta),$$

where $R(\beta)$ can be $\ell_1$ norm, MCP, SCAD, group $\ell_1$, group MCP or group SCAD.

For sparse column inverse operation,

$$\min_{\beta} \frac{1}{2} \beta^T S \beta - e^T \beta + \lambda R(\beta),$$

where $R(\beta)$ can be $\ell_1$ norm, MCP or SCAD.

Value

An object with S3 classes "lasso", "logit", "mvr" and "scio" corresponding to sparse linear regression, sparse logistic regression, sparse multivariate regression and sparse column inverse operator respectively is returned:

- **beta**: A matrix of regression estimates whose columns correspond to regularization parameters for sparse linear regression and sparse logistic regression. A list of matrices of regression estimation corresponding to regularization parameters for sparse multivariate regression and sparse column inverse operator.
- **intercept**: The value of intercepts corresponding to regularization parameters for sparse linear regression, sparse multivariate regression and sparse logistic regression.
- **Y**: The value of Y used in the program.
- **X**: The value of X used in the program.
- **lambda**: The sequence of regularization parameters used in the program.
- **nlambda**: The number of values used in lambda.
- **family**: The family from the input.
- **method**: The method from the input.
alg  The alg from the input.
sym  The sym 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 for sparse inverse column operator.
standardize  The standardize from the input.
perturb  The perturb from the input.
df  The degree of freedom (number of nonzero coefficients) along the solution path for sparse linear regression, sparse multivariate regression and sparse logistic regression.
ite  A list of vectors where the \( i \)-th entries of ite[[1]] and ite[[2]] correspond to the outer iteration and inner iteration of \( i \)-th regularization parameter respectively.
verbose  The verbose from the input.

Author(s)

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

References


See Also

picasso-package.

Examples

# Sparse linear regression
## Generate the design matrix and regression coefficient vector
n = 100
d = 400
X = matrix(rnorm(n*d), n, d)
beta = c(3,2,0,1.5,rep(0,d-4))

## Generate response using Gaussian noise, and fit sparse linear models
noise = rnorm(n)
```
Y = XX*%beta + noise
out.l1.cyclic = picasso(X, Y, nlambda=10)
out.l1.greedy = picasso(X, Y, nlambda=10, alg="greedy")
out.mcp.cyclic = picasso(X, Y, nlambda=10, method="mcp")
out.gr.prox = picasso(X, Y, nlambda=10, method="group", alg="prox")

## Visualize the solution path
plot(out.l1.cyclic)
plot(out.l1.greedy)
plot(out.mcp.cyclic)
plot(out.gr.prox)

################################################################
## sparse logistic regression
## generate the design matrix and regression coefficient vector
n = 100
d = 400
X = matrix(rnorm(n*d), n, d)
beta = c(3, 2, 0.1.5, rep(0, d-4))

## generate response and fit sparse logistic models
p = exp(XX*%beta)/(1+exp(XX*%beta))
Y = rbinom(n, rep(1, n), p)
out.l1.cyclic = picasso(X, Y, nlambda=10, family="logit")
out.l1.greedy = picasso(X, Y, nlambda=10, family="logit", alg="greedy")
out.mcp.cyclic = picasso(X, Y, nlambda=10, family="logit", method="mcp")
out.gr.prox = picasso(X, Y, nlambda=10, family="logit", method="group", alg="prox")

## Visualize the solution path
plot(out.l1.cyclic)
plot(out.l1.greedy)
plot(out.mcp.cyclic)
plot(out.gr.prox)

## estimate of bernoulli parameters
p.l1 = out.l1.cyclic$p

################################################################
## sparse multivariate regression
## generate the design matrix and regression coefficient vector
n = 100
d = 400
p = 10
X = matrix(rnorm(n*d), n, d)
beta = matrix(0, d, p)
beta[c(1, 2, 4),] = c(3, 2, 1.5)

## generate response using Gaussian noise, and fit sparse linear models
noise = matrix(rnorm(n*p), nrow=n)
Y = XX*%beta + noise
out.cyclic = picasso(X, Y, nlambda=10)
```
plot.lasso

Plot Function for "lasso"

Description

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

Usage

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

Arguments

x An object with S3 class "lasso".
... Arguments to be passed to methods.

Author(s)

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

See Also

picasso and picasso-package.
**plot.logit**

*Plot Function for "logit"*

### Description

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

### Usage

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

### Arguments

- **x**: An object with S3 class "logit".
- **...**: Arguments to be passed to methods.

### Author(s)

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

### See Also

[picasso](https://cran.r-project.org/package=picasso) and [picasso-package](https://cran.r-project.org/package=picasso).

---

**plot.scio**

*Plot Function for "scio"*

### Description

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

### Usage

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

### Arguments

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

Author(s)

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

See Also

picasso and picasso-package

---

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

```r
## 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 and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

scio.generator, picasso and picasso-package
Description

Predicting responses of the given design data.

Usage

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

Arguments

- `object`: An object with S3 class "lasso"
- `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.lasso` 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 and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

`picasso` and `picasso-package`.
Examples

```r
## 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 "l1".
out=picasso(x=X.fit,Y=Y.fit,lambda=lambda,family="gaussian")

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

predict.logit  

*Prediction for an object with S3 class "logit"*

Description

Predicting responses of the given design data.

Usage

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

Arguments

- **object**: An object with S3 class "logit"
- **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).
- **p.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

predictNlogit produces predicted values of the responses of the newdata from the estimated beta values in the object, i.e.

\[ \hat{p} = \frac{e^{\hat{\beta}_0 + X_{new}\hat{\beta}}}{1 + e^{\hat{\beta}_0 + X_{new}\hat{\beta}}} \]

Value

pNpred The predicted response vectors based on the estimated models.

Author(s)

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

See Also

picasso and picasso-package.

Examples

```r
## 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),]
beta = c(rep(sqrt(1/3),3),rep(0,d-3))
Y.fit = exp(X.fit%*%beta)/(1+exp(X.fit%*%beta))

## Regression with "l1".
out=picasso(X=X.fit,Y=Y.fit,lambda=lambda,family="logit")

## Display results
p=predict(out,X.pred)
```
Description

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

Usage

## S3 method for class 'lasso'
print(x, ...)

Arguments

x       An object with S3 class "lasso".
...
...   Arguments to be passed to methods.

Details

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

Author(s)

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

See Also

picasso and picasso-package.

Description

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

Usage

## S3 method for class 'logit'
print(x, ...)

Arguments

x       An object with S3 class "logit".
...
...   Arguments to be passed to methods.
Details
This call simply outlines the options used for computing a logit object.

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

See Also
picasso and picasso-package.

print.mvr
Print Function for an object with S3 class "mvr"

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

Usage
## S3 method for class 'mvr'
print(x, ...) 

Arguments

x  An object with S3 class "mvr".

...  Arguments to be passed to methods.

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

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

See Also
picasso and picasso-package.
**print.scio**  
*Print Function for an object with S3 class "scio"*

**Description**  
Print a summary of the information about an object with S3 class "s1im".

**Usage**  
```r  
## S3 method for class 'scio'  
print(x, ...)  
```

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

**Details**  
This call simply outlines the options used for computing a scio object.

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

**See Also**  
`picasso` and `picasso-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 and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

`picasso` and `scio.generator`

---

**scio.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

```r
scio.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".
- `v`: 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.
- `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.
seed
Set seed for data generation. The default value is 1.

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.

verbose
If verbose = 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!=j with probability prob, and 0 otherwise. It results in about d*(d-1)*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!=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!=j with the probability prob if both i and j belong to the same group, and 0 otherwise. It results in about g*(d/g)*((d/g-1)*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 otherwise. 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 cov2cor(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 and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

picasso and picasso-package

Examples

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

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

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

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

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

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

scio.plot

Graph visualization for an object with S3 class "scio"

Description

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

Usage

```r
scio.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 and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

See Also

- picasso
- picasso-package

Examples

```r
## visualize the hub graph
L = scio.generator(graph = "hub")
scio.plot(L$theta)

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

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

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

dcoef.lasso, 3
dcoef.logit, 3
deyedata, 4
dpicasso, 2–4, 5, 10–13, 15–19, 21, 22
dpicasso-package, 2
dplot.lasso, 10
dplot.logit, 11
dplot.scio, 11
dplot.sim, 12
dpredict.lasso, 13
dpredict.logit, 14
dprint.lasso, 16
dprint.logit, 16
dprint.mvr, 17
dprint.scio, 18
dprint.sim, 18

dscio.generator, 2, 12, 19, 19
dscio.plot, 21