Package 'fastclime'

April 29, 2016

Type Package

Title A Fast Solver for Parameterized LP Problems, Constrained L1 Minimization Approach to Sparse Precision Matrix Estimation and Dantzig Selector

Version 1.4.1

Date 2016-04-22

Author Haotian Pang, Di Qi, Han Liu and Robert Vanderbei

Maintainer Haotian Pang <hpang@princeton.edu>

Depends R (>= 2.15.0), lattice, igraph, MASS, Matrix

Description Provides a method of recovering the precision matrix efficiently and solving for the dantzig selector by applying the parametric simplex method. The computation is based on a linear optimization solver. It also contains a generic LP solver and a parameterized LP solver using parametric simplex method.

License GPL-2

Repository CRAN

NeedsCompilation yes

Date/Publication 2016-04-29 23:59:28

R topics documented:

`fastclime-package` *Fast Parametric Simplex Solver for CLIME and Linear Programming*

Description

A package for generic linear programming, parameterized linear programming and constrainted l1 minimization approach to sparse precision matrix estimation

Details

Package:	fastclime
Type:	Package
Version:	1.2.4
Date:	2014-04-25
License:	GPL-2
LazyLoad:	yes

The package "fastclime" provides 5 main functions:
(1) the data generator creates random samples from multivariate normal distributions with different graph structures. Please refer to `fastclime.generator`.
(2) The parametric simplex solver for constrainted l1 minimization approach to sparse precision matrix estimation. Please refer to `fastclime`.
(3) The path selector function gives the path and precision matrix for a given parameter in CLIME. Please refer to `fastclime.selector`.
(4) A generic linear programming solver and a parameterized linear programming solver. Please refer to `fastlp` and `paralp`.
(5) An implementation of the Primal Dual (i.e. Self Dual) Simplex Method on the Dantzig selector. Please refer to `dantzig`, `dantzig.selector` and `dantzig.generator`.

Author(s)

Haotian Pang, Di Qi, Han Liu and Robert Vanderbei
Maintainer: Haotian Pang<hpang@princeton.edu>;

See Also

`fastclime.generator`, `fastclime`, `fastclime.plot`, `fastclime.selector`, `fastlp`, `paralp`, `dantzig`, `dantzig.selector`, and `dantzig.generator`

dantzig *A solver for the Dantzig selector estimator*

Description

Implementation of the Primal Dual (i.e. Self Dual) Simplex Method on Dantzig selector

Usage

```
dantzig(X, y, lambda = 0.01, nlambda = 50)
```

Arguments

X	x is an n by d data matrix
y	y is a length n response vector
lambda	The parametric simplex method will stop when the calculated parameter is smaller than lambda. The default value is 0.01.
nlambda	This is the number of the maximum path length one would like to achieve. The default length is 50.

Details

This program applies the parametric simplex linear programming method to the Dantzig selector to solve for the regression coefficient vector. The solution path of the problem corresponds to the parameter in the parametric simplex method.

Value

An object with S3 class "dantzig" is returned:

X	X is the n by d data matrix.
y	y is a length n response vector.
BETA0	BETA0 is a d by validn matrix where each column has an estimated regression coefficient vector given a lambda interval.
n0	n0 is the number of rows in the n by d data matrix.
d0	d0 is the number of columns in the n by d data matrix.
validn	validn is the number of solutions along the solution path. The maximum is nlambda.
lambdalist	lambdalist is the decrementing path of the lambda solution values.

Note

The program will stop when either the maximum number of iterations for each column nlambda is achieved or when the required lambda is achieved for each column. Note if d is large and nlambda is also large, it is possible that the program will fail to allocate memory for the path.

Author(s)

Haotian Pang, Han Liu, Robert Vanderbei and Di Qi
Maintainer: Haotian Pang<hpang@princeton.edu>

See Also

dantzig.selector

Examples

```
#generate data
a = dantzig.generator(n = 200, d = 100, sparsity = 0.1)

#regression coefficient estimation
b = dantzig(a$X0, a$y, lambda = 0.1, nlambda = 100)
```

---

dantzig.generator *Dantzig data generator*

Description

Generates sparse linear regression model for testing dantzig function

Usage

```
dantzig.generator(n = 50, d = 100, sparsity = 0.1, sigma0=1)
```

Arguments

n	The number of observations (sample size). The default value is 50.
d	The number of variables (dimension). The default value is 100.
sparsity	d is either the number of nonzero entries out of d or the proportion of nonzero entries in BETA
sigma0	sigma0 is the standard deviation of the noise vector

Details

Generates sparse linear regression model for testing dantzig function

Value

An object with S3 class "dantzig.generator" is returned:

X0	X0 is the n by d matrix for the generated data
y	A y is a n response vector for the generated data
BETA	BETA is a length d regression coefficient vector
s	s is the number of nonzero entries out of d
pos	A vector containing the indices of the nonzero entries (may contain repeats)

Author(s)

Haotian Pang, Han Liu, Robert Vanderbei and Di Qi
Maintainer: Di Qi <dqi@princeton.edu>

See Also

dantzig

Examples

```
##
L = dantzig.generator(n = 50, d = 100, sparsity = 0.1)
```

---

dantzig.selector *Dantzig selector*

Description

Function used to select the solution path for a given lambda

Usage

```
dantzig.selector(lambdalist, BETA0, lambda)
```

Arguments

lambdalist	lambdalist is the length validn decrementing path of the lambda solution values. It is obtained from the dantzig function.
BETA0	BETA0 is a d by validn matrix where each column has an estimated regression coefficient vector given a given lambda interval. It is obtained from the dantzig function.
lambda	lambda is the lambda solution value the user wishes to estimate a regression coefficient vector with.

Details

Finds the estimated regression coefficient vector associated with a given `lambda`

Value

`beta0`	`beta0` is the estimated regression coefficient vector for the given `lambda`.

Author(s)

Di Qi, Haotian Pang
Maintainer: Di Qi <dqi@princeton.edu>

See Also

`dantzig`

Examples

```
# generate data
a = dantzig.generator(n = 200, d = 100, sparsity = 0.1)

# regression coefficient estimation
b = dantzig(a$X0, a$y, lambda = 0.1, nlambda = 100)

# estimated regression coefficient vector
c = dantzig.selector(b$lambdalist, b$BETA0, 15)
```

`fastclime` *The main solver for fastclime package*

Description

A fast parametric simplex solver for constrainted l1 minimization approach to sparse precision matrix estimation

Usage

```
fastclime(x, lambda.min = 0.1, nlambda = 50)
```

Arguments

`x`	There are 2 options: (1) `x` 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)

`lambda.min` This is the smallest value of lambda you would like the solver to explorer. The default value is `0.1`. If `nlambda` is large enough, the precision matrix selector function `fastclime.selector` will be able to find all precision matrix corresponding to all lambda values ranging from `1` to `lambda.min`.

`nlambda` It is the number of the path length one would like to achieve. The default length is 50. Note if d is large and nlambda is also large, it is possible that the program will fail to allocate memory for the path.

Details

This program uses parametric simplex linear programming method to solve CLIME (Constrained l1 Minimization Sparse Precision Matrix Estimation) problem. The solution path of the problem corresponds to the parameter in the parametric simplex method.

Value

An object with S3 class `"fastclime"` 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.

`sigmahat` The empirical covariance of the data. If cov.inpu is TRUE, sigmahat = data

`maxnlambda` The length of the path. If the program finds `lambda.min` in less than `nlambda` iterations for all columns, then the acutal maximum lenth for all columns will be returned. Otherwise it equals `nlambda`.

`lambdamtx` The sequence of regularization parameters for each column, it is a `nlambda` by `d` matrix. It will be filled with 0 when the program finds the required `lambda.min` value for that column. This parameter is required for `fastclime.selector`.

`icovlist` A `nlambda` list of `d` by `d` precision matrices as an alternative graph path (numerical path) corresponding to `lambdamtx`. This parameter is also required for `fastclime.selector`.

Note

The program will stop when either the maximum number of iteration for each column `nlambda` is achieved or when the required `lambda.min` is achieved for each column. When the dimension is huge, make sure `nlambda` is small so that there are enough memory to allocate the solution path. `lambdamtx` and `icovlist` will be used in `fastclime.selector`.

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

`fastclime.generator`, `fastclime.plot`, `fastclime.selector` and `fastclime-package`.

Examples

```
#generate data
L = fastclime.generator(n = 100, d = 20)

#graph path estimation
out1 = fastclime(L$data,0.1)
out2 = fastclime.selector(out1$lambdamtx, out1$icovlist,0.2)
fastclime.plot(out2$adaj)

#graph path estimation using the sample covariance matrix as the input.
out1 = fastclime(cor(L$data),0.1)
out2 = fastclime.selector(out1$lambdamtx, out1$icovlist,0.2)
fastclime.plot(out2$adaj)
```

`fastclime.generator` *Data generator*

Description

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

Usage

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

Arguments

`n`	The number of observations (sample size). The default value is `200`.
`d`	The number of variables (dimension). The default value is `50`.
`graph`	The graph structure with 4 options: `"random"`, `"hub"`, `"cluster"` and `"band"`.
`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 >= 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 <= 30` and `0.3` if `d/g > 30`. NOT applicable to `"hub"` or `"band"` graphs.
`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` other wise. 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 `probif` both `i` and `j` belong to the same group, and `0` other wise. 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` other wise. It results in `(2d-1-g)*g/2` edges in the graph.

The adjacency matrix `theta` has all diagonal elements equal to `0`. To obtain a positive definite precision matrix, the smallest eigenvalue of `theta*v` (denoted by `e`) is computed. Then we set the precision matrix equal to `theta*v+(|e|+0.1+u)I`. The covariance matrix is then computed 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)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

`fastclime` and `fastclime-package`

Examples

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

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

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

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

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

---

fastclime.plot *Graph visualization*

Description

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

Usage

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

Arguments

G	The adjaceny 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 applicale 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)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime and fastclime-package

Examples

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

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

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

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

---

fastclime.selector *A precision matrix and path selector function for fastclime*

Description

Select the precision matrix and solution path for a given parameter lambda

Usage

```
fastclime.selector(lambdamtx, icovlist, lambda)
```

Arguments

lambdamtx	The sequence of regularization parameters for each column, it is a nlambda by d matrix.
icovlist	A nlambda list of d by d precision matrices as an alternative graph path (numerical path) corresponding to lambdamtx.
lambda	The user specified parameter lambda. The function will return the solution path corresponding to this value. Note lambda has to be larger than or equal to lambda.min input in fastclime.

Details

The output from fastclime stores a list of precision matrice and a matrix of parameters. This program will select the required solution path and precision matrix for a given parameter lambda.

Value

An object with S3 class "fastclime.selector" is returned:

icov	The estimated precision matrix corresponding to lambda.
adaj	The estimated graph path corresponding to lambda.
sparsity	The sparsity level of this estimated graph for this value of lambda.

Note

The function is able to estimate the precision matrices corresponding to all lambda values ranging from 1 to lambda.min, provided a large enough nlambda is used in fastclime. The function will give a message if the program could not find the solution path corresponding to the required lamba. The user may want to incease nlambda in fastclime in order to find the required solution path.

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime and fastclime-package

Examples

```
#generate data
L = fastclime.generator(n = 100, d = 20)

#graph path estimation
out1 = fastclime(L$data,0.1)
out2 = fastclime.selector(out1$lambdamtx, out1$icovlist,0.2)
fastclime.plot(out2$adaj)
```

---

fastlp *A generic LP solver*

Description

A generic linear programming solver using parametric simplex method

Usage

```
fastlp(obj,mat,rhs,lambda=0)
```

Arguments

obj	The objective vector of the coefficient with length n.
mat	The constraint matrix of the linear programming with dimension m*n. Note this argument must be in matrix form even it is a vector.
rhs	The right hand side vector of the constraint with length m.
lambda	The parametric simplex method will stop when the calculated paramter is smaller than lambda. The default value is zero and it corresponds to the optimal value.

Details

This function is used to solve a general linear programming in standard inequality form: "maximize obj*x, subject to: mat*x<=rhs, x>=0"

Value

The optimal value will be returned if it exists. Otherwise the function will indicate the problem is infeasible or unbounded.

Note

The linear programming should be in the form "maximize obj*x, subject to: mat*x<=rhs, x>=0". If the original problem is not in this form. The user has to convert it into this form. For example, the equality constrants can be separated into two inequality constraints.

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime and fastclime-package

Examples

```
#generate an LP problem and solve it
A=matrix(c(-1,-1,0,1,-2,1),nrow=3)
b=c(-1,-2,1)
c=c(-2,3)
fastlp(c,A,b)
```

paralp *A solver for parameterized LP problems*

Description

A parameterized linear programming solver using parametric simplex method

Usage

```
paralp(obj,mat,rhs,obj_bar,rhs_bar,lambda=0)
```

Arguments

obj	The objective vector of the coefficient with length n.
mat	The constraint matrix of the linear programming with dimension m*n. Note this argument must be in matrix form even it is a vector.
rhs	The right hand side vector of the constraint with length m.
obj_bar	The vector used to time the parameter and added to the objective vector, with length n. This pertubation vector must be nonnegative.
rhs_bar	The vector used to time the parameter and added to the right hand side vector, with length m. This pertubation vector must be nonnegative.
lambda	The parametric simplex method will stop when the calculated paramter is smaller than lambda. The default value is zero and it corresponds to the optimal value.

Details

This function is used to solve a general linear programming in standard inequality form: "maximize obj*x+obj_bar*lambda, subject to: mat*x<=rhs+rhs_bar*lambda, x>=0"

Value

The optimal value will be returned if it exists with a proper value of chosen lambda. Otherwise the function will indicate the problem is infeasible or unbounded.

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime and fastclime-package

Examples

```
#generate an LP problem and solve it
A=matrix(c(-1,-1,0,1,-2,1),nrow=3)
b=c(-1,-2,1)
c=c(-2,3)
b_bar=c(1,1,1)
c_bar=c(1,1)
paralp(c,A,b,c_bar,b_bar)
```

---

plot.fastclime *Plot function for S3 class "fastclime"*

Description

Plot sparsity level information (the first column) from the graph path

Usage

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

Arguments

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

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime

---

plot.sim *Plot function for S3 class "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"
...	System reserved (No specific usage)

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime.generator and fastclime

---

print.fastclime *Print function for S3 class "fastclime"*

Description

Print the information about the model usage, the graph path length, graph dimension, sparsity level

Usage

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

Arguments

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

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime and fastclime

print.sim *Print function for S3 class "sim"*

Description

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

Usage

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

Arguments

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

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

See Also

fastclime.generator and fastclime.generator

stockdata *Stock price of S&P 500 companies from 2003 to 2008*

Description

This data set consists of stock price and company information.

Usage

```
data(stockdata)
```

Format

The format is a list containing conatins two matrices. 1. data - 1258x452, represents the 452 stocks' close prices for 1258 trading days. 2. info - 452x3: The 1st column: the query symbol for each company. The 2nd column: the categoriy for each company. The 3rd column: the full name of each company.

Details

This data set can be used to perform high-dimensional graph estimation to analyze the relationships between S&P 500 companies.

Author(s)

Haotian Pang, Han Liu and Robert Vanderbei
Maintainer: Haotan Pang<hpang@princeton.edu>

Source

It is publicly available at http://ichart.finance.yahoo.com

Examples

```
data(stockdata)
image(stockdata$data)
stockdata$info
```

Index