Package ‘glmgraph’

July 19, 2015

Type Package

Title Graph-Constrained Regularization for Sparse Generalized Linear Models

Version 1.0.3

Date 2015-07-19

Author Li Chen, Jun Chen

Maintainer Li Chen <li.chen@emory.edu>

Description We propose to use sparse regression model to achieve variable selection while accounting for graph-constraints among coefficients. Different linear combination of a sparsity penalty(L1) and a smoothness(MCP) penalty has been used, which induces both sparsity of the solution and certain smoothness on the linear coefficients.

License GPL-2

Depends Rcpp (>= 0.11.0)

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-07-19 09:52:47

R topics documented:

glmgraph-package *Fit a GLM with a combination of sparse and smooth regularization*

Description

Fit a generalized linear model at grids of tuning parameter via penalized maximum likelihood. The regularization path is computed for a combination of sparse and smooth penalty at two grids of values for the regularization parameter lambda1(Lasso or MCP penalty) and lambda2(Laplacian penalty). Fits linear, logistic regression models.

Details

Package:	glmgraph
Type:	Package
Version:	1.0-0
Date:	2015-03-11
License:	GPL-2

The algorithm accepts a design matrix `X`, a vector of responses `Y` and a Laplacian matrix `L`. Produces the regularization path over the grid of tuning parameter `lambda1` and `lambda2`. It consists of the following main functions
`glmgraph`
`cv.glmgraph`
`plot.glmgraph`
`coef.glmgraph`
`predict.glmgraph`

Author(s)

Li Chen <li.chen@emory.edu>, Jun Chen <jun.chen2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen(2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
```

```
magnitude <- 1
## Construct Adjacency and Laplacian matrices
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
Y <- eta+rnorm(n)
obj <- glmgraph(X,Y,L,family="gaussian")
plot(obj)
betas <- coef(obj)
betas <- coef(obj,lambda1=c(0.1,0.2))
yhat <- predict(obj,X,type="response")
cv.obj <- cv.glmgraph(X,Y,L)
plot(cv.obj)
beta.min <- coef(cv.obj)
yhat.min <- predict(cv.obj,X)
```

`coef.cv.glmgraph` *Retrieve coefficients from a fitted "cv.glmgraph" object.*

Description

Retrieve coefficients from a fitted "cv.glmgraph" object based on the chosen regularization parameters from cross validation.

Usage

```
## S3 method for class 'cv.glmgraph'
coef(object,s,...)
```

Arguments

`object`	Fitted "`cv.glmgraph`" model object.
`s`	Either "`lambda1.min`" or "`lambda1.1se`". If "`lambda1.min`" is used, coefficients of best cross validation criteria(minimum "mse" or "mae" if `family` is "gaussian"; maximum "auc" or minimum "deviance" if `family` is "binomial") are returned. Otherwise, coefficients based on one-standard error rule are returned. The default value is "`lambda1.min`".
`...`	Other parameters to `coef`

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

`predict.cv.glmgraph`,`cv.glmgraph`

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
## construct laplacian matrix from adjacency matrix
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
### gaussian
Y <- eta+rnorm(n)
cv.obj <- cv.glmgraph(X,Y,L)
beta.min <- coef(cv.obj)
```

`coef.glmgraph` *Retrieve coefficients from a fitted "glmgraph" object.*

Description

Retrieve coefficients from a fitted "glmgraph" object, depending on the user-specified regularization parameters.

Usage

```
## S3 method for class 'glmgraph'
coef(object,lambda1,lambda2,...)
```

Arguments

`object`	Fitted `"glmgraph"` model object.
`lambda1`	Values of the regularization parameter `lambda1` at which retrieval of coefficients are requested. For values of `lambda1` not in the sequence of fitted models, linear interpolation is used. However, `lambda1` should be within the range of `lambda1` used to fit `glmgraph` object.
`lambda2`	The user-specified regularization `lambda2` should be exactly subset of the `lambda2` used to fit `glmgraph` object. Linear interpolation is not used.
`...`	Other parameters to `coef`

Details

If `lambda1` and `lambda2` are missing, all coefficients of fitted `glmgraph` object will be returned. If only `lambda1` is missing, then coefficients of specified `lambda2` will be returned.

Value

The object returned depends on type.

Author(s)

Li Chen <li.chen@emory.edu>, Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

`predict.glmgraph`,`glmgraph`

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
## construct laplacian matrix from adjacency matrix
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
```

```
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
obj <- glmgraph(X,Y,L)
coefs <- coef(obj)
coefs <- coef(obj,lambda2=0.01)
coefs <- coef(obj,lambda1=c(0.11,0.12))
coefs <- coef(obj,lambda1=c(0.11,0.12),lambda2=0.01)
```

`cv.glmgraph` *Cross-validation for glmgraph*

Description

Performs k-fold cross validation for glmgraph

Usage

```
cv.glmgraph(X,Y,L,...,type.measure=c("mse","mae","deviance","auc"),nfolds=5,trace=TRUE)
```

Arguments

`X`	`X` matrix as in `glmgraph`.
`Y`	Response `Y` as in `glmgraph`.
`L`	User-specified Laplacian matrix `L` as in `glmgraph`.
`...`	Additional arguments as in `glmgraph`.
`type.measure`	if `family` is "gaussian", the `type.measure` option is "mse"(mean squared error) or "mae"(mean absolute error); if `family` is "binomial", the `type.measure` option is "deviance" or "auc"(area under the curve). The default is "mse".
`nfolds`	The number of cross-validation folds. Default is 5.
`trace`	Print out the cross validation steps if `trace` is specified TRUE.

Details

The function runs `glmgraph` `nfolds+1` times; the first to get the `lambda1` and `lambda2` sequence, and then the remainder to compute the fit with each of the folds omitted. The error is accumulated, and the average error and standard deviation over the folds is computed. Note also that the results of `cv.glmgraph` are random, since the folds are selected at random. Users can reduce this randomness by running `cv.glmgraph` many times, and averaging the error curves.

Value

An object "cv.glmgraph" containing:

- `obj` The fitted glmgraph object for the whole data.
- `cvmat` A data frame summarized cross validation results, which could be obtained by print function. It has lambda2,lambda1.min,cvmin,semin,lambda1.1se as columns. Each row represents that for this lambda2, lambda1 with best type.measure cvmin is chosen and reported as lambda1.min. If one standard error rule is applied, lambda1.1se and its corresponding best type.measure value semin is reported.
- `cvm` The mean cross-validated type.measure value. A list of vector contains type.measure. Each element of the list is a vector that is type.measure value for one lambda2 across all lambda1 sequence averaged across K-fold.
- `cvsd` The estimate of standard error of cvm.
- `cvmin` Best cross-validation type.measure value across all combination of lambda1 and lambda2. It is minimum "mse" or "mae" if family is "gaussian"; it is the maximum "auc" or minimum "deviance" if family is "binomial".
- `cv.1se` Simliar to cvmin except one standard error rule is applied.
- `lambda1.min` Coupled with lambda2.min is the optimal regularization parameter selection.
- `lambda2.min` Coupled with lambda1.min is the optimal regularization parameter selection.
- `lambda1.1se` Coupled with lambda2.1se is the optimal regularization parameter selection if one standard error rule is applied.
- `lambda2.1se` Coupled with lambda1.1se is the optimal regularization parameter selection if one standard error rule is applied.
- `beta.min` Estimated beta with best type.measure value with the regularization parameter of lambda1.min and lambda2.min.
- `beta.1se` Estimated beta with best type.measure value with the regularization parameter of lambda1.1se and lambda2.1se.

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen(2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

glmgraph,coef.cv.glmgraph,predict.cv.glmgraph

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
## construct laplacian matrix from adjacency matrix
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
### gaussian
Y <- eta+rnorm(n)
cv.obj <- cv.glmgraph(X,Y,L,penalty="lasso",lambda2=c(0,1.28))
beta.min <- coef(cv.obj)
print(cv.obj)
### binomial
Y <- rbinom(n,1,prob=1/(1+exp(-eta)))
cv.obj <- cv.glmgraph(X,Y,L,family="binomial",lambda2=c(0,1.28),penalty="lasso",type.measure="auc")
beta.min <- coef(cv.obj)
print(cv.obj)
```

`glmgraph` *Fit a GLM with a combination of sparse and smooth regularization*

Description

Fit a generalized linear model at grids of tuning parameter via penalized maximum likelihood. The regularization path is computed for a combination of sparse and smooth penalty at two grids of values for the regularization parameter lambda1(Lasso or MCP penalty) and lambda2(Laplacian penalty). Fits linear, logistic regression models.

Usage

```
glmgraph(X, Y, L, family=c("gaussian","binomial"), penalty=c("MCP","lasso") ,
mcpapproach=c("mmcd", "adaptive", "original"),gamma=8,
lambda1,nlambda1=100,lambda2=c(0, 0.01 * 2^(0:7)),eps=1e-3,max.iter=2000,
dfmax=round(ncol(X)/2),penalty.factor=rep(1,ncol(X)),standardize=TRUE,warn=FALSE,...)
```

Arguments

`X`	Input matrix; each row is an observation vector.
`Y`	Response vector. Quantitative for `family="gaussian"` or binary(0/1) for `family="binomial"`.
`family`	Either "gaussian", "binomial", depending on the response.
`L`	User-specified Laplacian matrix.
`penalty`	The sparse penalty to be applied to the model. Either "MCP" (the default), or "lasso".
`mcpapproach`	For `family="binomial"`, three optional algorithms are provided when `penalty` is set to MCP: "mmcd"(Majorization minimization by coordinate descent); "adaptive"(Adaptive rescaling) and "original"(without any adjustment). For `family="gaussian"`, the option could only be "original".
`gamma`	The tuning parameter of the MCP penalty.The default value is 8.
`nlambda1`	The number of `lambda1` values. Default is 100.
`lambda1`	A user-specified sequence of `lambda1` values. Typical usage is to have the program compute its own `lambda1` sequence based on `nlambda1` and `lambda1.min.ratio`. Supplying a value of `lambda1` overrides this. By default, a sequence of values of length `nlambda1` is computed, equally spaced on the log scale.
`lambda2`	A user-specified sequence of `lambda2` values. The default value are 0 and 0.01*2^(0:7). The selection of `lambda2` depends on the data and should be adapted in some cases. A good suggestion is to try a few `lambda2` and plot the results.
`eps`	Convergence threshold for coordinate descent. Each inner coordinate-descent loop continues until the relative change in the objective function is less than `eps1`. Default is `1e-3`.
`max.iter`	Maximum number of passes over the data for all `lambda1` values. Default is 2000.
`dfmax`	Limit the maximum number of variables in the model. Useful for very large `p`. Default value equals to half of `p`.
`penalty.factor`	A multiplicative factor for the penalty applied to each coefficient. If supplied, `penalty.factor` must be a numeric vector of length equal to the number of columns of `X`. The purpose of `penalty.factor` is to apply differential penalization if some coefficients are thought to be more likely than others to be in the model. In particular, `penalty.factor` can be 0, in which case the coefficient is always in the model without shrinkage.
`standardize`	Logical flag for x variable standardization, prior to fitting the model sequence. The coefficients are always returned on the original scale. Default is `standardize=TRUE`. If variables are in the same units already, you might not wish to standardize.
`warn`	Return warning messages for failures to converge and model selection issues. Default is FALSE.
`...`	Other parameters to `glmgraph`

Value

An object "glmgraph" containing:

betas	A list of fitted coefficients. The number of rows for each matrix is equal to the number of coefficients, and the number of columns is smaller or equal to nlambda1.
lambda1s	A list of vector. Each vector is a sequence of used lambda1 for each used lambda2.
lambda2	A sequence of lambda2 actually used.
loglik	A list of log likelihood for each value of lambda1 and lambda2.
df	A list of the number of nonzero values for each value of lambda1 and lambda2.

Author(s)

Li Chen <li.chen@emory.edu>, Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

plot.glmgraph, cv.glmgraph

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
### construct laplacian matrix from adjacency matrix
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
obj <- glmgraph(X,Y,L,family="gaussian")
```

```
plot(obj)
### binomial
Y <- rbinom(n,1,prob=1/(1+exp(-eta)))
obj <- glmgraph(X,Y,L,family="binomial")
plot(obj)
```

`plot.cv.glmgraph` *Plot the cross-validation curve produced by cv.glmgraph*

Description

Plots the cross-validation curve for the "cv.glmgraph" object, along with standard error bars.

Usage

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

Arguments

`x`	A "`cv.glmgraph`" object.
`...`	Other graphical parameters to `plot`

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

`glmgraph`, `cv.glmgraph`

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
```

```
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
### construct laplacian matrix from adjacency matrix
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
cv.obj <- cv.glmgraph(X,Y,L)
plot(cv.obj)
```

---

plot.glmgraph *Plot coefficients from a "glmgraph" object*

Description

Plot solution path for a fitted "glmgraph" object.

Usage

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

Arguments

x	Fitted "glmgraph" model.
...	Other graphical parameters to plot

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

glmgraph

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
### construct laplacian matrix from adjacency matrix
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
obj <- glmgraph(X,Y,L)
plot(obj)
### binomial
Y <- rbinom(n,1,prob=1/(1+exp(-eta)))
obj <- glmgraph(X,Y,L,family="binomial")
plot(obj)
```

`predict.cv.glmgraph` *make prediction from a fitted "cv.glmgraph" object.*

Description

This function makes predictions from a cross-validated glmgraph model, using the stored "`cv.glmgraph`" object, and the optimal value chosen for `lambda1` and `lambda2`.

Usage

```
## S3 method for class 'cv.glmgraph'
predict(object,X,s=c("lambda1.min","lambda1.1se"),
type=c("response", "coefficients","class", "nzeros","link"),...)
```

Arguments

`object`	Fitted "`cv.glmgraph`" model object.
`X`	Matrix at which predictions are to be made.

s	Either "lambda1.min" or "lambda1.1se".If "lambda1.min" is used, prediction based on coefficient of best cross validation criteria(minimum "mse" or "mae" if family is "gaussian"; maximum "auc" or minimum "deviance" if family is "binomial") are returned. Otherwise, predictficients based on one-standard error rule are returned. The default value is "lambda1.min".
type	Type of prediction: "link" returns the linear predictors; "response" gives the fitted values; "class" returns the binomial outcome with the highest probability; "coefficients" returns the coefficients; "nzeros" returns a list containing the indices and names of the nonzero variables at each combination of lambda1 and lambda2.
...	Other parameters to predict

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@emory.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

cv.glmgraph,coef.cv.glmgraph

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
### construct laplacian matrix from adjacency matrix
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
cv.obj <- cv.glmgraph(X,Y,L)
beta.min <- predict(cv.obj,X,type="coefficients")
```

predict.glmgraph *Model predictions based on a fitted "glmgraph" object.*

Description

Similar to other predict methods, this function returns predictions from a fitted "glmgraph" object.

Usage

```
## S3 method for class 'glmgraph'
predict(object, X, type=c("response", "coefficients",
"class", "nzeros","link"), lambda1, lambda2,...)
```

Arguments

object	Fitted "glmgraph" model object.
X	Matrix of values at which predictions are to be made.
lambda1	Values of the regularization parameter lambda1 at which predictions are requested. For values of lambda1 not in the sequence of fitted models, linear interpolation is used.
lambda2	Values of the regularization parameter lambda1 at which predictions are requested. Specified lambda2 should be the subet of lambda2 used to fit glmgraph object.
type	Type of prediction: "link" returns the linear predictors; "response" gives the fitted values; "class" returns the binomial outcome with the highest probability; "coefficients" returns the coefficients; "nzeros" returns a list containing the indices and names of the nonzero variables at each combination of lambda1 and lambda2.
...	Other parameters to predict

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

glmgraph

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
## construct laplacian matrix from adjacency matrix
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
obj <- glmgraph(X,Y,L)
res <- predict(obj, X, type="link", lambda1=0.05,lambda2=0.01)
res <- predict(obj, X, type="response", lambda1=0.05,lambda2=0.01)
res <- predict(obj,X,type="nzeros",lambda1=0.05,lambda2=0.01)
### binomial
Y <- rbinom(n,1,prob=1/(1+exp(-eta)))
obj <- glmgraph(X,Y,L,family="binomial")
res <- predict(obj,X,type="class",lambda1=c(0.05,0.06),lambda2=c(0.02,0.16,0.32))
```

---

print.cv.glmgraph *print a glmgraph object*

Description

Print a summary of the cv.glmgraph solution path information during cross validation

Usage

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

Arguments

x	fitted `cv.glmgraph` object
...	Other parameters to `print`

Details

The call prints the `cvmat` object from a fitted `cv.glmgraph` object. The call also prints the chosen regularization parameters lambda1 and lambda2 along with best `cv.type`(minimum "mse" or "mae" if `family` is "gaussian"; maximum "auc" or minimum "deviance" if `family` is "binomial") after cross validation.

Author(s)

Li Chen <li.chen@emory.edu> , Jun Chen <chen.jun2@mayo.edu>

References

Li Chen. Han Liu. Hongzhe Li. Jun Chen. (2015) glmgraph: Graph-constrained Regularization for Sparse Generalized Linear Models.(Working paper)

See Also

`cv.glmgraph`

Examples

```
set.seed(1234)
library(glmgraph)
n <- 100
p1 <- 10
p2 <- 90
p <- p1+p2
X <- matrix(rnorm(n*p), n,p)
magnitude <- 1
A <- matrix(rep(0,p*p),p,p)
A[1:p1,1:p1] <- 1
A[(p1+1):p,(p1+1):p] <- 1
diag(A) <- 0
btrue <- c(rep(magnitude,p1),rep(0,p2))
intercept <- 0
eta <- intercept+X%*%btrue
### construct laplacian matrix from adjacency matrix
diagL <- apply(A,1,sum)
L <- -A
diag(L) <- diagL
### gaussian
Y <- eta+rnorm(n)
cv.obj <- cv.glmgraph(X,Y,L)
print(cv.obj)
```

Index