Package: HDCI (via r-universe)

September 2, 2024

Type Package

Title High Dimensional Confidence Interval Based on Lasso and Bootstrap

Version 1.0-2

Date 2017-06-06

Author Hanzhong Liu, Xin Xu, Jingyi Jessica Li

Maintainer Xin Xu <xin.xu@yale.edu>

Imports glmnet, slam, parallel, foreach, iterators, doParallel, lattice, Matrix, mvtnorm

Description Fits regression models on high dimensional data to estimate coefficients and use bootstrap method to obtain confidence intervals. Choices for regression models are Lasso, Lasso+OLS, Lasso partial ridge, Lasso+OLS partial ridge.

License GNU General Public License version 2

NeedsCompilation no

RoxygenNote 6.0.1

Repository https://xinxuyale.r-universe.dev

RemoteUrl https://github.com/xinxuyale/hdci

RemoteRef HEAD

RemoteSha 8b93bf98f106a61400f08b52ad4806016da0adf3

Contents

`bootLasso` *Bootstrap Lasso*

Description

Does residual (or paired) bootstrap Lasso and produces confidence intervals for regression coefficients.

Usage

```
bootLasso(x, y, B = 500, type.boot = "residual", alpha = 0.05,
          cv.method = "cv", nfolds = 10, foldid, cv.OLS = FALSE, tau = 0,
          parallel = FALSE, standardize = TRUE, intercept = TRUE,
          parallel.boot = FALSE, ncores.boot = 1, ...)
```

Arguments

`x`	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
`y`	Response variable.
`B`	Number of replications in the bootstrap – default is 500.
`type.boot`	Bootstrap method which can take one of the following two values: "residual" or "paired". The default is residual.
`alpha`	Significance level – default is 0.05.
`cv.method`	The method used to select lambda in the Lasso – can be cv, cv1se, and escv; the default is cv.
`nfolds`, `foldid`, `cv.OLS`, `tau`, `parallel`	Arguments that can be passed to escv.glmnet.
`standardize`	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
`intercept`	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
`parallel.boot`	If TRUE, use parallel foreach to run the bootstrap replication. Must register parallel before hand, such as doParallel or others. See the example below.
`ncores.boot`	Number of cores used in the bootstrap replication.
`...`	Other arguments that can be passed to glmnet.

Details

The function runs residual (type.boot="residual") or paired (type.boot="paired") bootstrap Lasso procedure, and produces confidence interval for each individual regression coefficient. Note that there are two arguments related to parallel, "parallel" and "parallel.boot": "parallel" is used for parallel foreach in the escv.glmnet; while, "paralle.boot" is used for the parallel foreach in the bootstrap replication precodure.

Value

A list consisting of the following elements is returned.

`lambda.opt`	The optimal value of lambda selected by cv/cv1se/escv.
`Beta`	An estimate of the regression coefficients.
`interval`	A 2 by p matrix containing the confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## residual bootstrap Lasso
set.seed(0)
obj <- bootLasso(x = x, y = y, B = 10)
# confidence interval
obj$interval
sum((obj$interval[1,]<=beta) & (obj$interval[2,]>=beta))

## using parallel in the bootstrap replication
#library("doParallel")
#registerDoParallel(2)
#set.seed(0)
#system.time(obj <- bootLasso(x = x, y = y))
#system.time(obj <- bootLasso(x = x, y = y, parallel.boot = TRUE, ncores.boot = 2))
```

`bootLassoOLS` *Bootstrap Lasso OLS*

Description

Does residual (or paired) bootstrap Lasso+OLS and produces confidence intervals for regression coefficients.

Usage

```
bootLassoOLS(x, y, B = 500, type.boot = "residual", alpha = 0.05, OLS = TRUE,
             cv.method = "cv", nfolds = 10, foldid, cv.OLS = TRUE, tau = 0,
             parallel = FALSE, standardize = TRUE, intercept = TRUE,
             parallel.boot = FALSE, ncores.boot = 1, ...)
```

Arguments

`x`	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
`y`	Response variable.
`B`	Number of replications in the bootstrap – default is 500.
`type.boot`	Bootstrap method which can take one of the following two values: "residual" or "paired". The default is "residual".
`alpha`	Significance level – default is 0.05.
`OLS`	If TRUE, this function runs residual (or paired) bootstrap Lasso+OLS; if FALSE, it runs residual (or paired) bootstrap Lasso. The default value is TRUE.
`cv.method`	The method used to select lambda in the Lasso+OLS – can be cv, cv1se, and escv; the default is cv.
`nfolds`, `foldid`, `cv.OLS`, `tau`, `parallel`	Arguments that can be passed to escv.glmnet.
`standardize`	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
`intercept`	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
`parallel.boot`	If TRUE, use parallel foreach to run the bootstrap replication. Must register parallel before hand, such as doParallel or others. See the example below.
`ncores.boot`	Number of cores used in the bootstrap replication.
`...`	Other arguments that can be passed to glmnet.

Details

The function runs residual (type.boot="residual") or paired (type.boot="paired") bootstrap Lasso+OLS (if OLS=TRUE) procedure, and produces confidence interval for each individual regression coefficient. When the argument OLS=FALSE, it is the same as the function bootLasso, which runs residual (type.boot="residual") or paired (type.boot="paired") bootstrap Lasso procedure. Note that there are two arguments related to parallel, "parallel" and "parallel.boot": "parallel" is used for parallel foreach in the escv.glmnet; while, "paralle.boot" is used for the parallel foreach in the bootstrap replication precodure.

Value

A list consisting of the following elements is returned.

`lambda.opt`	The optimal value of lambda selected by cv/cv1se/escv.
`Beta.Lasso`	Lasso estimate of the regression coefficients.
`Beta.LassoOLS`	Lasso+OLS estimate of the regression coefficients. It gives back to Lasso estimate if the argument OLS=FALSE.
`interval.Lasso`	A 2 by p matrix containing the bootstrap Lasso confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.
`interval.LassoOLS`	A 2 by p matrix containing the bootstrap Lasso+OLS confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals. It equals interval.Lasso if the argument OLS=FALSE.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## residual bootstrap Lasso+OLS
set.seed(0)
obj <- bootLassoOLS(x = x, y = y, B = 10)
# confidence interval
obj$interval
sum((obj$interval[1,]<=beta) & (obj$interval[2,]>=beta))
```

```
## using parallel in the bootstrap replication
#library("doParallel")
#registerDoParallel(2)
#set.seed(0)
#system.time(obj <- bootLassoOLS(x = x, y = y))
#system.time(obj <- bootLassoOLS(x = x, y = y, parallel.boot = TRUE, ncores.boot = 2))
```

---

bootLOPR *Bootstrap Lasso OLS Partial Ridge*

Description

Combines residual (or paired) bootstrap Lasso+Partial Ridge and residual (or paired) bootstrap Lasso+OLS, and produces confidence intervals for regression coefficients.

Usage

```
bootLOPR(x, y, lambda2, B = 500, type.boot = "residual", thres = 0.5, alpha = 0.05,
        OLS = TRUE, cv.method = "cv", nfolds = 10, foldid, cv.OLS = TRUE, tau = 0,
         parallel = FALSE, standardize = TRUE, intercept = TRUE, parallel.boot =
         FALSE, ncores.boot = 1, ...)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
y	Response variable.
lambda2	Tuning parameter in the Partial Ridge. If missing, lambda2 will be set to 1/nobs, where nobs is the number of observations.
B	Number of replications in the bootstrap – default is 500.
type.boot	Bootstrap method which can take one of the following two values: "residual" or "paired". The default is residual.
thres	A threshold parameter. For the variables/predictors with selection probability (obtained by bootstrap) larger than thres, this function uses bootstrap Lasso+OLS (if OLS=TRUE) or bootstrap Lasso (if OLS=FALSE) to produce confidence intervals; while, for the variables/predictors with selection probability (obtained by bootstrap) smaller than thres, this function uses bootstrap Lasso+Partial Ridge to produce confidence intervals.
alpha	Significance level – default is 0.05.
OLS	If TRUE, this function uses Lasso+OLS estimator to compute the residuals for residual bootstrap Lasso+Partial Ridge; otherwise, it uses Lasso estimator to compute the residuals for residual bootstrap Lasso+Partial Ridge. The default value is TRUE. This argument can be ignored for paired bootstrap Lasso+Partial Ridge.

`cv.method` The method used to select lambda in the Lasso – can be cv, cv1se, and escv; the default is cv.

`nfolds`, `foldid`, `cv.OLS`, `tau`, `parallel`
Arguments that can be passed to escv.glmnet.

`standardize` Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.

`intercept` Should intercept be fitted (default is TRUE) or set to zero (FALSE).

`parallel.boot` If TRUE, use parallel foreach to run the bootstrap replication. Must register parallel before hand, such as doParallel or others. See the example below.

`ncores.boot` Number of cores used in the bootstrap replication.

`...` Other arguments that can be passed to glmnet.

Details

The function combines the performance of bootstrap Lasso+Partial Ridge and bootstrap Lasso+OLS (if OLS=TRUE). For "large" regression coefficient in the sense that their selection probability (obtained by bootstrap) is larger than a threshold value (thres), it uses bootstrap Lasso+OLS to produce confidence intervals which can decrease the interval length ; while, for "small" regression coefficients meaning that their selection probability (obtained by bootstrap) is smaller than a threshold value (thres), it uses bootstrap Lasso+Partial Ridge to produce confidence intervals which can guarantee coverage. Note that there are two arguments related to parallel, "parallel" and "parallel.boot": "parallel" is used for parallel foreach in the escv.glmnet; while, "paralle.boot" is used for the parallel foreach in the bootstrap replication precodure.

Value

A list consisting of the following elements is returned.

`lambda.opt` The optimal value of lambda selected by cv/cv1se/escv.

`Beta` Lasso+OLS (if OLS=TRUE) or Lasso (if OLS=FALSE) estimate of the regression coefficients.

`Beta.LPR` Lasso+Partial Ridge estimate of the regression coefficients.

`interval` A 2 by p matrix containing the bootstrap Lasso+OLS (if OLS=TRUE) or bootstrap Lasso (if OLS=FALSE) confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.

`interval.LPR` A 2 by p matrix containing the bootstrap Lasso+Partial Ridge confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.

`interval.LOPR` A 2 by p matrix containing the combining confidence intervals of bootstrap Lasso+Partial Ridge and bootstrap Lasso+OLS (or bootstrap Lasso if OLS=FALSE) – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## residual bootstrap Lasso OLS + Partial Ridge
set.seed(0)
obj <- bootLOPR(x = x, y = y, B = 10)
# confidence interval
obj$interval
sum((obj$interval[1,]<=beta) & (obj$interval[2,]>=beta))

## using parallel in the bootstrap replication
#library("doParallel")
#registerDoParallel(2)
#set.seed(0)
#system.time(obj <- bootLOPR(x = x, y = y))
#system.time(obj <- bootLOPR(x = x, y = y, parallel.boot = TRUE, ncores.boot = 2))
```

---

bootLPR *Bootstrap Lasso Partial Ridge*

Description

Does residual (or paired) bootstrap Lasso+Partial Ridge and produces confidence intervals for regression coefficients.

Usage

```
bootLPR(x, y, lambda2, B = 500, type.boot = "paired", alpha = 0.05, OLS = TRUE,
        cv.method = "cv", nfolds = 10, foldid, cv.OLS = TRUE, tau = 0,
        parallel = FALSE, standardize = TRUE, intercept = TRUE,
        parallel.boot = FALSE, ncores.boot = 1, ...)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
y	Response variable.
lambda2	Tuning parameter in the Partial Ridge. If missing, lambda2 will be set to 1/nobs, where nobs is the number of observations.
B	Number of replications in the bootstrap – default is 500.
type.boot	Bootstrap method which can take one of the following two values: "residual" or "paired". The default is residual.
alpha	Significance level – default is 0.05.
OLS	If TRUE, this function uses Lasso+OLS estimator to compute the residuals for residual bootstrap Lasso+Partial Ridge; otherwise, it uses Lasso estimator to compute the residuals for residual bootstrap Lasso+Partial Ridge. The default value is TRUE. This argument can be ignored for paired bootstrap Lasso+Partial Ridge.
cv.method	The method used to select lambda in the Lasso – can be cv, cv1se, and escv; the default is cv.
nfolds, foldid, cv.OLS, tau, parallel	Arguments that can be passed to escv.glmnet.
standardize	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
parallel.boot	If TRUE, use parallel foreach to run the bootstrap replication. Must register parallel before hand, such as doParallel or others. See the example below.
ncores.boot	Number of cores used in the bootstrap replication.
...	Other arguments that can be passed to glmnet.

Details

The function runs residual (type.boot="residual") or paired (type.boot="paired") bootstrap Lasso+Partial Ridge procedure, and produces confidence interval for each individual regression coefficient. If the argument OLS=TRUE, it uses Lasso+OLS estimator to compute the residuals for residual bootstrap Lasso+Partial Ridge; otherwise, it uses Lasso estimator to compute the residuals. Note that there are two arguments related to parallel, "parallel" and "parallel.boot": "parallel" is used for parallel foreach in the escv.glmnet; while, "paralle.boot" is used for the parallel foreach in the bootstrap replication precodure.

Value

A list consisting of the following elements is returned.

lambda.opt	The optimal value of lambda selected by cv/cv1se/escv.
Beta	Lasso+OLS (if OLS=TRUE) or Lasso (if OLS=FALSE) estimate of the regression coefficients.

Beta.LPR	Lasso+Partial Ridge estimate of the regression coefficients.
interval	A 2 by p matrix containing the bootstrap Lasso+OLS (if OLS=TRUE) or bootstrap Lasso (if OLS=FALSE) confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.
interval.LPR	A 2 by p matrix containing the bootstrap Lasso+Partial Ridge confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## residual bootstrap Lasso+Partial Ridge
set.seed(0)
obj <- bootLPR(x = x, y = y, type.boot = "residual", B = 10)
# confidence interval
obj$interval
sum((obj$interval[1,]<=beta) & (obj$interval[2,]>=beta))

## using parallel in the bootstrap replication
#library("doParallel")
#registerDoParallel(2)
#set.seed(0)
#system.time(obj <- bootLPR(x = x, y = y))
#system.time(obj <- bootLPR(x = x, y = y, parallel.boot = TRUE, ncores.boot = 2))
```

---

ci *Confidence Interval*

Description

Gets bootstrap confidence intervals for each of the coefficients of variables/predictors.

Usage

```
ci(Beta, Beta_bootstrap, alpha = 0.05, type = c("basic", "quantile", "bca", "basic2"),
   a, Beta2)
```

Arguments

Beta	An estimate of the coefficients.
Beta_bootstrap	Bootstrap estimates of the coefficients – a B by p matrix, where B is the number of replications in the bootstrap and p is number of variables/predictors.
alpha	Significance level – default is 0.05.
type	Different type of confidence interval: basic, quantile, bca (adjusted bootstrap confidence intervals) and basic2 (a modified basic confidence interval).
a	Parameter in the bca confidence interval.
Beta2	An estimator of the coefficients used in the basic2 confidence interval.

Value

interval	A 2 by p matrix containing the confidence intervals – the first row is the lower bounds of the confidence intervals for each of the coefficients and the second row is the upper bounds of the confidence intervals.

---

escv.glmnet *escv glmnet*

Description

Does k-fold estimation stability with cross-validation (escv) for glmnet and returns optimal values for lambda.

Usage

```
escv.glmnet(x, y, lambda = NULL, nfolds = 10, foldid, cv.OLS = FALSE, tau = 0, parallel
            = FALSE, standardize = TRUE, intercept = TRUE, ...)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector. Can be in sparse matrix format (inherit from class "sparseMatrix" as in package Matrix).
y	Response variable.
lambda	Optional user-supplied lambda sequence for the Lasso; default is NULL, and glmnet chooses its own sequence.
nfolds	Number of folds - default is 10.

foldid	An optional vector of values between 1 and nfold identifying what fold each observation is in. If supplied, nfolds can be missing.
cv.OLS	If TRUE, uses two-stage estimator Lasso+OLS in the fits (using Lasso to select variables/predictors and then using OLS to refit the coefficients for the selected variables/predictors. The default value is FALSE.
tau	Tuning parameter in modified Least Squares (mls). Default value is 0, which corresponds to OLS.
parallel	If TRUE, use parallel foreach to fit each fold. Must register parallel before hand, such as doParallel or others. See the example below.
standardize	Logical flag for x variable standardization, prior to fitting the model sequence. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
...	Other arguments that can be passed to glmnet.

Details

The function is similar to cv.glmnet, and returns the values of lambda selected by cross-validation (cv), by cross-validation within 1 standard error (cv1se) and by estimation stability with cross-validation (escv). The function runs glmnet nfolds+1 times; the first to get the lambda sequence, and then the remainder to compute the first stage fit (i.e., Lasso) with each of the folds omitted. The error (cv and also es) is accumulated, and the average error and standard deviation over the folds is computed. Note that, similar to cv.glmnet, the results of escv.glmnet are random, since the folds are selected at random. Users can reduce this randomness by running escv.glmnet many times, and averaging the error curves.

Value

A list consisting of the following elements is returned.

lambda	The values of lambda used in the fits.
glmnet.fit	A fitted glmnet object for the full data.
cv	The mean cross-validated error - a vector of length length(lambda).
cv.error	Estimate of standard error of cv.
es	The mean estimation stability (es) value - a vector of length length(lambda).
es.error	Estimate of standard error of es.
lambda.cv	Value of lambda that gives minimum cv.
lambda.cv1se	Largest value of lambda such that cross-validated error is within 1 standard error of the minimum.
lambda.escv	Value of lambda selected by escv – giving the minimum es within the range of lambdas which are no less than lambda.cv.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## escv without parallel
# using Lasso+OLS in the cv fit.
set.seed(0)
obj <- escv.glmnet(x, y, cv.OLS = TRUE)

# using Lasso in the cv fit.
set.seed(0)
obj <- escv.glmnet(x, y)

## escv with parallel
#library("doParallel")
#library("doRNG")
#registerDoParallel(2)

# using Lasso+OLS in the cv fit.
#registerDoRNG(seed = 0)
#obj <- escv.glmnet(x, y, cv.OLS = TRUE, nfolds = 4, parallel = TRUE)

# using Lasso in the cv fit.
#registerDoRNG(seed = 0)
#obj <- escv.glmnet(x, y, parallel = TRUE)
```

Lasso *Lasso*

Description

Gets Lasso estimator for a given value of lambda or for the value of lambda choosing by cross-validation (or escv).

Usage

```
Lasso(x, y, lambda = NULL, fix.lambda = TRUE, cv.method = "cv", nfolds = 10, foldid,
```

```
cv.OLS = FALSE, tau = 0, parallel = FALSE, standardize = TRUE, intercept = TRUE
  , ...)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
y	Response variable.
lambda	A value of lambda - default is NULL. lambda should be given a value when fix.lambda=TRUE.
fix.lambda	If TRUE, computes Lasso+OLS (or Lasso) for a fix value of lambda given by the argument "lambda"; otherwise, computes Lasso+OLS (or Lasso) for the value of lambda choosing by cv/cv1se/escv.
cv.method	The method used to select lambda – can be cv, cv1se, and escv; the default is cv. cv.method is useful only when fix.lambda=FALSE.
nfolds, foldid, cv.OLS, tau, parallel	Arguments that can be passed to escv.glmnet (useful only when fix.lambda=FALSE).
standardize	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
...	Other arguments that can be passed to glmnet.

Details

The function computes the Lasso estimator for a give value of lambda (if fix.lambda=TRUE) or for the value of lambda choosing by cv/cv1se/escv (if fix.lambda=FALSE).

Value

A list consisting of the following elements is returned.

beta	The Lasso estimate for the coefficients of variables/predictors.
beta0	A value of intercept term.
lambda	The value/values of lambda.
meanx	The mean vector of variables/predictors if intercept=TRUE, otherwise is a vector of 0's.
mu	The mean of the response if intercept=TRUE, otherwise is 0.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
```

```
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## Lasso estimator
# for a given value of lambda
set.seed(0)
obj.escv <- escv.glmnet(x, y)
obj <- Lasso(x, y, lambda = obj.escv$lambda.cv)
# Lasso estimate of the regression coefficients
obj$beta
# intercept term
obj$beta0
# prediction
mypredict(obj, newx = matrix(rnorm(10*p), 10, p))

# for lambda choosing by cross-validation (cv) which uses Lasso in the cv fit
set.seed(0)
obj <- Lasso(x, y, fix.lambda = FALSE)

# for lambda choosing by cross-validation (cv) which uses Lasso+OLS in the cv fit
set.seed(0)
obj <- Lasso(x, y, fix.lambda = FALSE, cv.OLS = TRUE)
```

LassoOLS *Lasso OLS*

Description

Computes the two-stage estimator Lasso+OLS (default) or the Lasso estimator (if OLS=FALSE).

Usage

```
LassoOLS(x, y, OLS = TRUE, lambda = NULL, fix.lambda = TRUE, cv.method = "cv", nfolds
       = 10, foldid, cv.OLS = TRUE, tau = 0, parallel = FALSE, standardize = TRUE,
         intercept = TRUE, ...)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
y	Response variable.

OLS	If TRUE, computes Lasso+OLS; otherwise, computes Lasso estimator. The default is TRUE.
lambda	A value of lambda - default is NULL. lambda should be given a value when fix.lambda=TRUE.
fix.lambda	If TRUE, computes Lasso+OLS (or Lasso) for a fix value of lambda given by the argument "lambda"; otherwise, computes Lasso+OLS (or Lasso) for the value of lambda choosing by cv/cv1se/escv.
cv.method	The method used to select lambda – can be cv, cv1se, and escv; the default is cv. cv.method is useful only when fix.lambda=FALSE.
nfolds, foldid, cv.OLS, tau, parallel	Arguments that can be passed to escv.glmnet (useful only when fix.lambda=FALSE). Note that, the default value of cv.OLS is TRUE, which means using Lasso+OLS in the cv fits.
standardize	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
...	Other arguments that can be passed to glmnet.

Details

If OLS=TRUE (default), this function computes the Lasso+OLS estimator for a give value of lambda (if fix.lambda=TRUE) or for the value of lambda choosing by cv/cv1se/escv (if fix.lambda=FALSE). If OLS=FALSE, this function computes the Lasso estimator in the same way as the function "Lasso". Note that, we use the easy-to-understand notation "Lasso+OLS" denoting the "Lasso+mLS" estimator defined in the paper: Liu H, Yu B. Asymptotic Properties of Lasso+mLS and Lasso+Ridge in Sparse High-dimensional Linear Regression. Electronic Journal of Statistics, 2013, 7.

Value

A list consisting of the following elements is returned.

beta	The Lasso+OLS (or Lasso when OLS=FALSE) estimate for the coefficients of variables/predictors.
beta0	A value of intercept term.
lambda	The value/values of lambda.
meanx	The mean vector of variables/predictors if intercept=TRUE, otherwise is a vector of 0's.
mu	The mean of the response if intercept=TRUE, otherwise is 0.
tau	Tuning parameter in modified Least Squares (mls).

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
```

```
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## Lasso+OLS estimator
# for a given value of lambda
set.seed(0)
obj.escv <- escv.glmnet(x, y)
obj <- LassoOLS(x, y, lambda = obj.escv$lambda.cv)
# Lasso+OLS estimate of the regression coefficients
obj$beta
# intercept term
obj$beta0
# prediction
mypredict(obj, newx = matrix(rnorm(10*p), 10, p))

# for lambda choosing by cross-validation (cv) which uses Lasso+OLS in the cv fit
set.seed(0)
obj <- LassoOLS(x, y, fix.lambda = FALSE)

# for lambda choosing by cross-validation (cv) which uses Lasso in the cv fit
set.seed(0)
obj <- LassoOLS(x, y, fix.lambda = FALSE, cv.OLS = FALSE)
```

LPR *Lasso Partial Ridge*

Description

Computes the two-stage estimator Lasso+Partial Ridge.

Usage

```
LPR(x, y, lambda = NULL, fix.lambda = TRUE, lambda2, cv.method = "cv", nfolds = 10,
   foldid, cv.OLS = TRUE, tau = 0, parallel = FALSE, standardize = TRUE, intercept =
    TRUE, ...)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.

y	Response variable.
lambda	lambda: A value of lambda - default is NULL. lambda should be given a value when fix.lambda=TRUE.
fix.lambda	If TRUE, computes Lasso+Partial Ridge estimator for a fix value of lambda given by the argument "lambda"; otherwise, computes Lasso+Partial Ridge estimator for the value of lambda choosing by cv/cv1se/escv.
lambda2	Tuning parameter in the Partial Ridge. If missing, lambda2 will be set to 1/nobs, where nobs is the number of observations.
cv.method	The method used to select lambda – can be cv, cv1se, and escv; the default is cv. cv.method is useful only when fix.lambda=FALSE.
nfolds, foldid, cv.OLS, tau, parallel	Arguments that can be passed to escv.glmnet (useful only when fix.lambda=FALSE). Note that, the default value of cv.OLS is TRUE, which means using Lasso+OLS in the cv fits.
standardize	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).
...	Other arguments that can be passed to glmnet.

Details

This function computes the Lasso+Partial Ridge estimator for a give value of lambda (if fix.lambda=TRUE) or for the value of lambda choosing by cv/cv1se/escv (if fix.lambda=FALSE).

Value

A list consisting of the following elements is returned.

beta	The Lasso+Partial Ridge estimator for the coefficients of variables/predictors.
beta0	A value of intercept term.
lambda	The value/values of lambda.
lambda2	The value of lambda2.
meanx	The mean vector of variables/predictors if intercept=TRUE, otherwise is a vector of 0's.
mu	The mean of the response if intercept=TRUE, otherwise is 0.
normx	The vector of standard error of variables/predictors if standardize=TRUE, otherwise is a vector of 1's.
tau	Tuning parameter in modified Least Squares (mls).

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
```

```
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## Lasso+Partial Ridge estimator
# for a given value of lambda
set.seed(0)
obj.escv <- escv.glmnet(x, y)
obj <- LPR(x, y, lambda = obj.escv$lambda.cv)
# Lasso+OLS estimate of the regression coefficients
obj$beta
# intercept term
obj$beta0
# prediction
mypredict(obj, newx = matrix(rnorm(10*p), 10, p))

# for lambda choosing by cross-validation (cv) which uses Lasso+OLS in the cv fit
set.seed(0)
obj <- LPR(x, y, fix.lambda = FALSE)

# for lambda choosing by cross-validation (cv) which uses Lasso in the cv fit
set.seed(0)
obj <- LPR(x, y, fix.lambda = FALSE, cv.OLS = FALSE)
```

`mls` *Modified Least Squares*

Description

Computes modified Least Squares estimate.

Usage

```
mls(x, y, tau = 0, standardize = TRUE, intercept = TRUE)
```

Arguments

`x`	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
`y`	Response variable.

tau	Tuning parameter in modified Least Squares (mls). Default value is 0, which corresponds to Ordinary Least Squares (OLS).
standardize	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).

Details

The function is used to compute the modified Least Squares (mLS) estimator defined in the paper: Liu H, Yu B. Asymptotic Properties of Lasso+mLS and Lasso+Ridge in Sparse High-dimensional Linear Regression. Electronic Journal of Statistics, 2013, 7.

Value

A list consisting of the following elements is returned.

beta	The mLS coefficient of variables/predictors.
beta0	A value of intercept term.
meanx	The mean vector of variables/predictors if intercept=TRUE, otherwise is a vector of 0's.
mu	The mean of the response if intercept=TRUE, otherwise is 0.
normx	The vector of standard error of variables/predictors if standardize=TRUE, otherwise is a vector of 1's.
tau	The tuning parameter in mLS.

Examples

```
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## modified Least Squares
set.seed(0)
obj <- mls(x = x[, 1:20], y = y)
# the OLS estimate of the regression coefficients
obj$beta
# intercept term
obj$beta0
# prediction
```

```
mypredict(obj, newx = matrix(rnorm(10*20), 10, 20))
```

`mypredict` *My Predict*

Description

Returns the predicted values.

Usage

```
mypredict(object, newx)
```

Arguments

`object`	An object from mls, Lasso, LassoOLS or PartialRidge.
`newx`	Matrix of the values of variables/predictors for doing prediction; each row is an observation vector.

Value

The predicted values for a give newx matrix is returned.

Examples

```
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## modified Least Squares
set.seed(0)
obj <- mls(x = x[, 1:20], y = y)
# the OLS estimate of the regression coefficients
obj$beta
# intercept term
obj$beta0
# prediction
mypredict(obj, newx = matrix(rnorm(10*20), 10, 20))
```

PartRidge *Partial Ridge*

Description

Computes the Partial Ridge estimator.

Usage

```
PartRidge(x, y, lambda2 = 0, varset, standardize = TRUE, intercept = TRUE)
```

Arguments

x	Input matrix as in glmnet, of dimension nobs x nvars; each row is an observation vector.
y	Response variable.
lambda2	Tuning parameter for the Partial Ridge. The default value is 0, which gives back to the OLS estimator.
varset	A set indicating which variable/predictors are not penalized. Partial Ridge puts l2 penalty only on the coefficients of the variables/predictors not included in the varset.
standardize	Logical flag for x variable standardization, prior to fitting the model. Default is standardize=TRUE.
intercept	Should intercept be fitted (default is TRUE) or set to zero (FALSE).

Details

This function computes the Partial Ridge estimator, which adds l2 penalty only on the coefficients of variabels/predictors not included in the set varset, to the loss function (residual sum of squares).

Value

A list consisting of the following elements is returned.

beta	The Lasso+Partial Ridge estimator for the coefficients of variables/predictors.
beta0	A value of intercept term.
meanx	The mean vector of variables/predictors if intercept=TRUE, otherwise is a vector of 0's.
mu	The mean of the response if intercept=TRUE, otherwise is 0.
normx	The vector of standard error of variables/predictors if standardize=TRUE, otherwise is a vector of 1's.
lambda2	The value of lambda2.

Examples

```
library("glmnet")
library("mvtnorm")

## generate the data
set.seed(2015)
n <- 200      # number of obs
p <- 500
s <- 10
beta <- rep(0, p)
beta[1:s] <- runif(s, 1/3, 1)
x <- rmvnorm(n = n, mean = rep(0, p), method = "svd")
signal <- sqrt(mean((x %*% beta)^2))
sigma <- as.numeric(signal / sqrt(10))  # SNR=10
y <- x %*% beta + rnorm(n)

## Partial Ridge Regression
# Lasso
set.seed(0)
obj.escv <- escv.glmnet(x, y)
obj <- Lasso(x, y, lambda = obj.escv$lambda.cv)
# variable set
betalasso <- obj$beta
selectvar <- betalasso != 0
# partial ridge
PR.obj <- PartRidge(x = x, y = y, lambda2 = 1/n, varset = selectvar)
# parial ridge estimate of the regression coefficients
beta <- PR.obj$beta
# intercept term
beta0 <- PR.obj$beta0
# prediction
mypredict(PR.obj, newx = matrix(rnorm(10*p), 10, p))
```

Index