

Package ‘LINselect’

January 10, 2020

Title Selection of Linear Estimators

Version 1.1.3

Date 2020-01-07

Author Yannick Baraud, Christophe Giraud, Sylvie Huet

Maintainer Benjamin Auder <benjamin.auder@universite-paris-saclay.fr>

Description

Estimate the mean of a Gaussian vector, by choosing among a large collection of estimators, following the method developed by Y. Baraud, C. Giraud and S. Huet (2014) <doi:10.1214/13-AIHP539>.

In particular it solves the problem of variable selection by choosing the best predictor among predictors emanating from different methods as lasso, elastic-net, adaptive lasso, pls, randomForest. Moreover, it can be applied for choosing the tuning parameter in a Gauss-lasso procedure.

Imports mvtnorm, elasticnet, MASS, randomForest, pls, gtools, stats

Depends R (>= 3.5.0)

License GPL (>= 3)

Encoding latin1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-01-10 05:30:40 UTC

R topics documented:

LINselect-package	2
penalty	2
simulData	3
tuneLasso	4
VARselect	6

Index	10
--------------	-----------

LINselect-package *Selection of linear estimators*

Description

LINselect allows to estimate the mean of a Gaussian vector, by choosing among a large collection of estimators. In particular it solves the problem of variable selection by choosing the best predictor among predictors emanating from different methods as lasso, elastic-net, adaptive lasso, pls, randomForest. Moreover, it can be applied for choosing the tuning parameter in a Gauss-lasso procedure.

Details

Package: LINselect
Title: Selection of linear estimators
Version: 0.0-0
Date: 2013-03-01
Author: Yannick Baraud, Christophe Giraud, Sylvie Huet
Maintainer: Annie Bouvier <Annie.Bouvier@jouy.inra.fr>
Suggests: mvtnorm, elasticnet, MASS, randomForest, pls, gtools
License: GPL (>= 3)
URL:
Encoding: latin1

Author(s)

Yannick Baraud, Christophe Giraud, Sylvie Huet

penalty *penalty*

Description

Calculate the penalty function for estimators selection.

Usage

penalty(Delta, n, p, K)

Arguments

Delta	vector with Dmax+1 components : weights in the penalty function.
n	integer : number of observatons.
p	integer : number of variables.
K	scalar : constant in the penalty function.

Value

A vector with the same length as Delta: for each $d=0, \dots, D_{\max}$, let $N=n-d$, $D=d+1$ and $\text{pen}(d) = x K N/(N-1)$ where x satisfies

$\phi(x) = \exp(-\text{Delta}(d))$, when $\text{Delta}(d) < 50$,
 where $\phi(x) = \text{pf}(q=x/(D+2), \text{df1}=D+2, \text{df2}=N-1, \text{lower.tail}=F) - (x/D) \text{pf}(q=(N+1)x/D(N-1), \text{df1}=D, \text{df2}=N+1, \text{lower.tail}=F)$

$\psi(x) = \text{Delta}(d)$, when $\text{Delta}(d) \geq 50$,
 where $\psi(x) = \text{lbeta}(1+D/2, (N-1)/2) - \log(2(2x+(N-1)D)/((N-1)(N+2)x)) - (N-1)/2 \log((N-1)/(N-1+x)) - (D/2) \log(x)$

Note

The values of the penalty function greater than $1e+08$ are set to $1e+08$.

If for some $\text{Delta}(d)$ the equation $\phi(x) = \exp(-\text{Delta}(d)/(d+1))$ has no solution, then the execution is stopped.

Author(s)

Yannick Baraud, Christophe Giraud, Sylvie Huet

simulData

simulData

Description

Function to simulate data $Y = X\beta + \sigma N(0, 1)$

Usage

```
simulData(p = 100, n = 100, beta = NULL, C = NULL, r = 0.95,
          rSN = 10)
```

Arguments

p	integer : number of variates. Should be >15 if beta=NULL
n	integer : number of observations
beta	vector with p components. See details.
C	matrix p x p. Covariance matrix of X. See details.
r	scalar for calculating the covariance of X when C=NULL.
rSN	scalar : ratio signal/noise

Details

When beta is NULL, then p should be greater than 15 and beta=c(rep(2.5, 5), rep(1.5, 5), rep(0.5, 5), rep(0, p-15))

When C is NULL, then C is block diagonal with

$C[a, b] = r^{**abs(a-b)}$ for $1 \leq a, b \leq 15$

$C[a, b] = r^{**abs(a-b)}$ for $16 \leq a, b \leq p$

The lines of X are n i.i.d. gaussian variables with mean 0 and covariance matrix C.

The variance σ^{**2} equals the squared euclidean norm of $X\beta$ divided by $rSN*n$.

Value

A list with components :

Y vector n : $Y = X\beta + \sigma N(0, 1)$

X matrix n x p : values of the covariates. See details.

C matrix p x p. See details

sigma scalar. See details.

beta vector with p components. See details.

Note

Library mvtnorm is loaded.

Author(s)

Yannick Baraud, Christophe Giraud, Sylvie Huet

tuneLasso

tuneLasso

Description

tune the lasso parameter in the regression model : $Y = X\beta + \sigma N(0, 1)$ using the lasso or the gauss-lasso method

Usage

```
tuneLasso(Y, X, normalize = TRUE, method = c("lasso", "Glasso"),
          dmax = NULL, vfold = TRUE, V = 10, LINselect = TRUE, a = 0.5,
          K = 1.1, verbose = TRUE, max.steps = NULL)
```

Arguments

Y	vector with n components : response variable.
X	matrix with n rows and p columns : covariates.
normalize	logical : corresponds to the input <code>normalize</code> of the functions <code>enet</code> and <code>cv.enet</code> . If TRUE the variates X are normalized.
method	vector of characters whose components are subset of ("lasso", "Glasso")
dmax	integer : maximum number of variables in the lasso estimator. $d_{\max} \leq D$ where $D = \min(3 \cdot p/4, n-5)$ if $p \geq n$ $D = \min(p, n-5)$ if $p < n$. Default : $d_{\max} = D$.
Vfold	logical : if TRUE the tuning is done by Vfold-CV
V	integer. Gives the value of V in the Vfold-CV procedure
LINselect	logical : if TRUE the tuning is done by LINselect
a	scalar : value of the parameter α in the LINselect criteria
K	scalar : value of the parameter K in the LINselect criteria
verbose	logical : if TRUE a trace of the current process is displayed in real time.
max.steps	integer : maximum number of steps in the lasso procedure. Corresponds to the input <code>max.steps</code> of the function <code>enet</code> . Default : $\text{max.steps} = 2 \cdot \min(p, n)$

Value

A list with one or two components according to `method`.

`lasso` if `method` contains "lasso" is a list with one or two components according to `Vfold` and `LINselect`.

- `Ls` if `LINselect=TRUE`. A list with components
 - `support`: vector of integers. Estimated support of the parameter vector β .
 - `coef`: vector whose first component is the estimated intercept.
The other components are the estimated non zero coefficients.
 - `fitted`: vector with length n. Fitted value of the response.
 - `crit`: vector containing the values of the criteria for each value of λ .
 - `lambda`: vector containing the values of the tuning parameter of the lasso algorithm.
- `CV` if `Vfold=TRUE`. A list with components
 - `support`: vector of integers. Estimated support of the parameter vector β .
 - `coef`: vector whose first component is the estimated intercept.
The other components are the estimated non zero coefficients.
 - `fitted`: vector with length n. Fitted value of the response.
 - `crit`: vector containing the values of the criteria for each value of λ .
 - `crit.err`: vector containing the estimated standard-error of the criteria.
 - `lambda`: vector containing the values of the tuning parameter of the lasso algorithm.

`Glasso` if `method` contains "Glasso". The same as `lasso`.

Note

library elasticnet is loaded.

Author(s)

Yannick Baraud, Christophe Giraud, Sylvie Huet

References

See Baraud et al. 2010 <http://hal.archives-ouvertes.fr/hal-00502156/fr/>
 Giraud et al., 2013, <http://projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.ss/1356098553>

Examples

```
#source("charge.R")
library("LINselect")

# simulate data with
## Not run: ex <- simulData(p=100,n=100,r=0.8,rSN=5)

## Not run: ex1.tuneLasso <- tuneLasso(ex$Y,ex$X)

## Not run: data(diabetes)
## Not run: attach(diabetes)
## Not run: ex.diab <- tuneLasso(y,x2)
## Not run: detach(diabetes)
```

 VARselect

VARselect

Description

Estimation in the regression model : $Y = X\beta + \sigma N(0, 1)$

Variable selection by choosing the best predictor among predictors emanating from different methods as lasso, elastic-net, adaptive lasso, pls, randomForest.

Usage

```
VARselect(Y, X, dmax = NULL, normalize = TRUE, method = c("lasso",
  "ridge", "pls", "en", "ALridge", "ALpls", "rF", "exhaustive"),
  pen.crit = NULL, lasso.dmax = NULL, ridge.dmax = NULL, pls.dmax = NULL,
  en.dmax = NULL, ALridge.dmax = NULL, ALpls.dmax = NULL, rF.dmax = NULL,
  exhaustive.maxdim = 5e+05, exhaustive.dmax = NULL, en.lambda = c(0.01,
  0.1, 0.5, 1, 2, 5), ridge.lambda = c(0.01, 0.1, 0.5,
  1, 2, 5), rF.lmtry = 2, pls.ncomp = 5, ALridge.lambda = c(0.01,
  0.1, 0.5, 1, 2, 5), ALpls.ncomp = 5, max.steps = NULL,
  K = 1.1, verbose = TRUE, long.output = FALSE)
```

Arguments

Y	vector with n components : response variable.
X	matrix with n rows and p columns : covariates.
dmax	integer : maximum number of variables in the lasso estimator. $dmax \leq D$ where $D = \min(3*p/4, n-5)$ if $p \geq n$ $D = \min(p, n-5)$ if $p < n$. Default : $dmax = D$.
normalize	logical : if TRUE the columns of X are scaled
method	vector of characters whose components are subset of "lasso", "ridge", "pls", "en", "ALridge", "ALpls", "rF", "exhaustive".
pen.crit	vector with $dmax+1$ components : for $d=0, \dots, dmax$, $penalty[d+1]$ gives the value of the penalty for the dimension d. Default : $penalty = NULL$. In that case, the penalty will be calculated by the function <code>penalty</code> .
lasso.dmax	integer lower than dmax, default = dmax.
ridge.dmax	integer lower than dmax, default = dmax.
pls.dmax	integer lower than dmax, default = dmax.
en.dmax	integer lower than dmax, default = dmax.
ALridge.dmax	integer lower than dmax, default = dmax.
ALpls.dmax	integer lower than dmax, default = dmax.
rF.dmax	integer lower than dmax, default = dmax.
exhaustive.maxdim	integer : maximum number of subsets of covariates considered in the exhaustive method. See details.
exhaustive.dmax	integer lower than dmax, default = dmax
en.lambda	vector : tuning parameter of the ridge. It is the input parameter <code>lambda</code> of function <code>enet</code>
ridge.lambda	vector : tuning parameter of the ridge. It is the input parameter <code>lambda</code> of function <code>lm.ridge</code>
rF.lmtry	vector : tuning parameter <code>mtry</code> of function <code>randomForest</code> , $mtry = p/rF.lmtry$.
pls.ncomp	integer : tuning parameter of the pls. It is the input parameter <code>ncomp</code> of the function <code>pls</code> . See details.
ALridge.lambda	similar to <code>ridge.lambda</code> in the adaptive lasso procedure.
ALpls.ncomp	similar to <code>pls.ncomp</code> in the adaptive lasso procedure. See details.
max.steps	integer. Maximum number of steps in the lasso procedure. Corresponds to the input <code>max.steps</code> of the function <code>enet</code> . Default : $max.steps = 2*\min(p,n)$
K	scalar : value of the parameter K in the LINselect criteria.
verbose	logical : if TRUE a trace of the current process is displayed in real time.
long.output	logical : if FALSE only the component summary will be returned. See Value.

Details

When method is `pls` or `ALpls`, the `LINselect` procedure is carried out considering the number of components in the `pls` method as the tuning parameter.

This tuning parameter varies from 1 to `pls.ncomp`.

When method is `exhaustive`, the maximum number of variate `d` is calculated as follows.

Let `q` be the largest integer such that `choose(p, q) < exhaustive.maxdim`. Then `d = min(q, exhaustive.dmax, dmax)`.

Value

A list with at least `length(method)` components.

For each procedure in `method` a list with components

- `support`: vector of integers. Estimated support of the parameters β for the considered procedure.
- `crit`: scalar equals to the `LINselect` criteria calculated in the estimated support.
- `fitted`: vector with length `n`. Fitted value of the response calculated when the support of β equals `support`.
- `coef`: vector whose first component is the estimated intercept.
The other components are the estimated non zero coefficients when the support of β equals `support`.

If `length(method) > 1`, the additional component `summary` is a list with three components:

- `support`: vector of integers. Estimated support of the parameters β corresponding to the minimum of the criteria among all procedures.
- `crit`: scalar. Minimum value of the criteria among all procedures.
- `method`: vector of characters. Names of the procedures for which the minimum is reached

If `pen.crit = NULL`, the component `pen.crit` gives the values of the penalty calculated by the function `penalty`. If `long.output` is `TRUE` the component named `chatty` is a list with `length(method)` components.

For each procedure in `method`, a list with components

- `support` where `support[[1]]` is a vector of integers containing an estimator of the support of the parameters β .
- `crit`: vector where `crit[1]` contains the value of the `LINselect` criteria calculated in `support[[1]]`.

Note

When method is `lasso`, library `elasticnet` is loaded.

When method is `en`, library `elasticnet` is loaded.

When method is `ridge`, library `MASS` is loaded.

When method is `rF`, library `randomForest` is loaded.

When method is `pls`, library `pls` is loaded.

When method is `ALridge`, libraries `MASS` and `elasticnet` are loaded.

When method is `ALpls`, libraries `pls` and `elasticnet` are loaded.

When method is `exhaustive`, library `gtools` is loaded.

Author(s)

Yannick Baraud, Christophe Giraud, Sylvie Huet

References

See Baraud et al. 2010 <http://hal.archives-ouvertes.fr/hal-00502156/fr/>
Giraud et al., 2013, <http://projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.ss/1356098553>

Examples

```
#source("charge.R")
library("LINselect")

# simulate data with
# beta=c(rep(2.5,5),rep(1.5,5),rep(0.5,5),rep(0,p-15))
ex <- simulData(p=100,n=100,r=0.8,rSN=5)

## Not run: ex1.VARselect <- VARselect(ex$Y,ex$X,exhaustive.dmax=2)

## Not run: data(diabetes)
## Not run: attach(diabetes)
## Not run: ex.diab <- VARselect(y,x2,exhaustive.dmax=5)
## Not run: detach(diabetes)
```

Index

*Topic **package**

LINselect-package, 2

cv.enet, 5

enet, 5, 7

LINselect (LINselect-package), 2

LINselect-package, 2

lm.ridge, 7

penalty, 2

pls, 7

randomForest, 7

simulData, 3

tuneLasso, 4

VARselect, 6