Package ‘SILM’

January 9, 2019

Type Package

Title Simultaneous Inference for Linear Models

Version 1.0.0

Date 2019-01-05

Author Xianyang Zhang, Guang Cheng, Jincheng Bai

Maintainer Xianyang Zhang <zhangxiany@stat.tamu.edu>


Depends scalreg

DependsNote scalreg does not correctly import lars etc, so we need to depend on it

Imports glmnet, hdi, SIS, stats, parallel, utils

License GPL-3

NeedsCompilation no

Encoding UTF-8

Repository CRAN

Date/Publication 2019-01-09 17:50:36 UTC

R topics documented:

Sim.CI ................................................................. 2
SR ................................................................. 3
ST ................................................................. 4
Step ............................................................... 5

Index 6
Sim.CI

Simultaneous Confidence Interval

Description

This function implements the method for constructing simultaneous confidence interval in Zhang and Cheng (2017).

Usage

Sim.CI(x, y, set, M = 500, alpha = 0.95)

Arguments

- **x**: n times p design matrix.
- **y**: Response variable.
- **set**: The set of variables of interest.
- **M**: The number of bootstrap replications (default 500).
- **alpha**: The nominal level alpha (default 0.95).

Value

The de-biased Lasso estimator, the confidence bands (lower bound and upper bound) delivered by the non-studentized and the studentized statistics.

References


Examples

```r
## The function is intended for large n and p.
## Use small p here for illustration purpose only.
n <- 100
p <- 10
s0 <- 3
set <- 1:s0
Sigma <- matrix(NA, p, p)
for (i in 1:p) Sigma[i,i] <- 0.9*(abs(i-1:p))
X <- matrix(rnorm(n*p), n, p)
X <- t(chol(Sigma))%*%t(X)
beta <- rep(0,p)
beta[1:s0] <- runif(s0,0,2)
Y <- X%*%beta+rt(n,4)/sqrt(2)
Sim.CI(X, Y, set)
```
**Support Recovery Procedure**

**Description**

This function implements the support recovery procedure in Zhang and Cheng (2017).

**Usage**

SR(X, Y)

**Arguments**

- **X**: n times p design matrix.
- **Y**: Response variable.

**Value**

The sets of active variables selected by the support recovery procedure and the scaled Lasso.

**References**


**Examples**

```r
## The function is intended for large n and p.
## Use small p here for illustration purpose only.
n <- 100
p <- 10
s0 <- 7
set <- 1:s0
Sigma <- matrix(NA, p, p)
for (i in 1:p) Sigma[i,] <- 0.9*(abs(i-1:p))
X <- matrix(rnorm(n*p), n, p)
X <- t(chol(Sigma))%*%t(X)
beta <- rep(0,p)
betar[1:s0] <- runif(s0,1,2)
Y <- X%*%beta+rt(n,4)/sqrt(2)
SR(X, Y)
```
Description

This function implements the method for testing sparse signals in Zhang and Cheng (2017).

Usage

```R
ST(X.f, Y.f, sub.size, test.set, M = 500, alpha = 0.05)
```

Arguments

- `X.f`: n times p design matrix.
- `Y.f`: Response variable.
- `sub.size`: The sub-sample size used for screening.
- `test.set`: The set of variables of interest.
- `M`: The number of bootstrap replications (default 500).
- `alpha`: The nominal level alpha (default 0.05).

Value

Values of the non-studentized and studentized statistics, and whether the tests get rejected at the level alpha.

References


Examples

```R
## The function is intended for large n and p.
## Use small p here for illustration purpose only.
n <- 100
p <- 10
s0 <- 3
set <- 1:s0
Sigma <- matrix(NA, p, p)
for (i in 1:p) Sigma[i,] <- 0.9*(abs(1:(1:p)))
X <- matrix(rnorm(n*p, n, p), n, p)
X <- t(chol(Sigma))%*%t(X)
beta <- rep(0,p)
beta[1:s0] <- runif(s0,0,2)
Y <- X%*%beta+rt(n,4)/sqrt(2)
test.set <- (s0+1):p
sub.size <- n*0.3
```
Stepdown Method for Multiple Testing

Description

This function implements the stepdown method in Zhang and Cheng (2017).

Usage

```
Step(X, Y, M = 500, alpha = 0.05)
```

Arguments

- `X`: n times p design matrix.
- `Y`: Response variable.
- `M`: The number of bootstrap replications (default 500).
- `alpha`: The nominal level alpha (default 0.05).

Value

A vector indicating which hypotheses are being rejected.

References


Examples

```r
## The function is intended for large n and p.
## Use small p here for illustration purpose only.
n <- 100
p <- 10
s0 <- 3
set <- 1:s0
Sigma <- matrix(NA, p, p)
for (i in 1:p) Sigma[i,] <- 0.9*(abs(i-(1:p)))
X <- matrix(rnorm(n*p), n, p)
X <- t(chol(Sigma))%*%t(X)
beta <- rep(0,p)
beta[1:s0] <- runif(s0,1,2)
Y <- X%*%beta+rt(n,4)/sqrt(2)
Step(X, Y, M=500, alpha=0.05)
```
Index

Sim.CI, 2
SR, 3
ST, 4
Step, 5