Package ‘LFabs’

February 22, 2021

Type Package

Title LFabs

Version 1.1.0

Author Yiming Liu [aut,cre], Xiao Zhang [aut], Xingjie Shi [aut]

Maintainer Yiming Liu <liuyimingsufe@163.com>

Description
High-dimensional smoothed partial rank estimation with sparse laplacian shrinkage for nonparametric transformation survival model. The laplacian penalty is to incorporate the correlation patterns among predictors. The adaptive lasso is used to yield a sparse estimator.

License GPL (>= 2)

Imports Matrix, mvtnorm

Repository github

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

R topics documented:

`adjmat` *Generate a sparse adjacency matrix*

Description

This function generates a sparse adjacency matrix for predictors. The elements of the matrix indicate whether pairs of vertices are adjacent or not in the graph. And each elements is based on the Pearson’s correlation coefficients and proper cut-off value. To save the storage space, this function only retains the information of nonzero ones

Usage

```
adjmat(x)
```

Arguments

x	The n x p design matrix.

Value

A list

- edge - The nonzero elements of the adjacency matrix.
- edgerow - The row index of each nonzero elements, which has the same length as edge.
- edgecol - The column index of each nonzero elements, which has the same length as edge.
- gamma - A p vector. Each elements is the sum of the column of the adjacency matrix.
- ledge - The number of the nonzero elements.

References

Jian Huang. Shuangge Ma. Hongzhe Li and Cun-Hui Zhang. (2011) *The sparse Laplacian shrinkage estimator for high-dimensional regression*

Examples

```
x = matrix(rnorm(200), 10, 20)
A <- adjmat(x)
```

cv.LFabs — *A Forward and Backward Stagewise algorithm for High-dimensional smoothed partial rank estimation with sparse laplacian shrinkage.*

Description

The laplacian shrinkage level is choosed by the cross-validation. This function uses the testing loss to select the turning.

Usage

```
cv.LFabs(
  X,
  y,
  s = NULL,
  sigma = NULL,
  weight = NULL,
  model = c("spr", "cox", "lm"),
  ntau = 5,
  nfold = 5
)
```

Arguments

`X` The design matrix.

`y` The response.

`s` The status.

`sigma` The smoothing parameter in SPR.

`weight` The weight vector of adaptive lasso.

`model` The loss function. Quantitative for model="spr", model="cox", model="lm".

`ntau` The length of the grid of the turning parameter before the laplacian penalty. Default is 5.

`nfold` The `nfold`-fold cross-validation. Default is 5.

Value

A list.

- Beta - The standardized estimator along the solution path.
- beta - The optimal standardized estimator.
- lambda - Lambda sequence generated by LFabs.
- tau - The optimal laplacian shrinkage level choosed by cross-validation.
- direction - Direction of LFabs. `1` indicates a forward step. `0` indicates a backward step.
- active - Active set for each step.
- iter - Iterations.
- bic - The bic for each solution.
- opt - Position of the optimal lambda based on bic.

See Also

`LFabs`.

Examples

```
library(mvtnorm)
library(Matrix)

n = 400
p = 500
d = 15
g = 5
sig = c(0.5, 1.5)
rho = 0.9
error = "contaminate"
tran = "log"
censor.rate = 0.1
block = "Auto"

set.seed(2021)
dat = generator(n, p, d, g, sig, rho, error, tran, censor.rate, block)
x = dat$x
y = dat$y
status = dat$status
```

```
sigma = 1/sqrt(n)
w = abs(1/drop(cor(x, y, method = "pearson")))
w[which(w=="NaN"|w=="Inf")] = max(w[which(w!="NaN"&w!="Inf")])
model = "spr"
fit <- cv.LFabs(x, y, status, sigma, w, model)
```

`generator` *Generate survival samples*

Description

This function generates survival data including response, covariates and status. Data can be right-censored and heavy-tailed. Morever, there may be a strong network structure and high-correlated pattern among covariates.

Usage

```
generator(
  n,
  p,
  d,
  g,
  sig,
  rho,
  error = c("norm", "contaminate", "t2", "ev"),
  tran = c("lm", "log"),
  censor.rate = 0,
  block = c("Block", "Auto", "BAND")
)
```

Arguments

`n` The number of samples.

`p` The number of covariates.

`d` The number of nonzero values of true coefficients.

`g` The group size.

`sig` The two parameters of the uniform distribution. For example, `sig=c(a,b)`, which means the elements of true coefficients are from `U(a,b)`.

`rho` The strength of correlation among covariates.

`error` The error distribution. Quantitative for error="norm", error="contaminate", error="t2" or error="ev". For error="contaminate", y is from `0.7N(0,1)+0.3Cauchy`.

`tran` The transformation function. Quantitative for tran="lm" or tran="log"

`censor.rate` The censor rate of survival sample. Default is 0.

`block` The network structure of predictors. Quantitative for block="Block", block="Auto", block="BAND". For block="BAND", the number of groups `g` is equal to `p`.

Value

A list

- x - The n x p design matrix.
- y - The n repsonse.
- b - The true coefficients.
- status - The survival status. 1 is censored and 0 is uncensored.

Examples

```
library(mvtnorm)
library(Matrix)

n = 400
p = 500
d = 15
g = 5
sig = c(0.5, 1.5)
rho = 0.9
error = "contaminate"
tran = "log"
censor.rate = 0.1
block = "Auto"

dat = generator(n, p, d, g, sig, rho, error, tran, censor.rate, block)
x = dat$x
y = dat$y
b = dat$b
status = dat$status
```

LFabs *A Forward and Backward Stagewise algorithm for High-dimensional smoothed partial rank estimation with sparse laplacian shrinkage.*

Description

The laplacian shrinkage level is fixed, which means tau is predetermined.

Usage

```
LFabs(
  X,
  y,
  s = NULL,
  sigma = NULL,
  weight = NULL,
  model = c("spr", "cox", "lm"),
  tau = 0,
  stoping = TRUE,
  eps = 0.02,
  xi = 10^-6,
  iter = 10^4,
  lambda.min = 1e-04
)
```

Arguments

`X`	The design matrix.
`y`	The response.
`s`	The status.
`sigma`	The smoothing parameter in SPR.
`weight`	The weight vector of adaptive lasso.
`model`	The loss function. Quantitative for model="spr", model="cox", model="lm".
`tau`	The turning parameter before the laplacian penalty. Default is 0.
`stoping`	The indicator of whether to stop iteration when lambda is less than lambda.min.
`eps`	The step size for updating coefficients. Default is 0.02.
`xi`	The threshhold for LFabs.
`iter`	The maximum number of outer-loop iterations allowed.
`lambda.min`	The smallest value for lambda, as a stopping Criterion of the solution path.

Value

A list.

- Beta - The standardized estimator along the solution path.
- beta - The optimal standardized estimator.
- lambda - Lambda sequence generated by LFabs.
- direction - Direction of LFabs. `1` indicates a forward step. `0` indicates a backward step.
- active - Active set for each step.
- iter - Iterations.
- bic - The bic for each solution.
- opt - Position of the optimal lambda based on bic.

Examples

```
library(mvtnorm)
library(Matrix)

n = 400
p = 500
d = 15
g = 5
sig = c(0.5, 1.5)
rho = 0.9
error = "contaminate"
tran = "log"
censor.rate = 0.1
block = "Auto"

set.seed(2021)
dat = generator(n, p, d, g, sig, rho, error, tran, censor.rate, block)
```

```
x = dat$x
y = dat$y
status = dat$status

sigma = 1/sqrt(n)
w = abs(1/drop(cor(x, y, method = "pearson")))
w[which(w=="NaN"|w=="Inf")] = max(w[which(w!="NaN"&w!="Inf")])
model = "spr"
tau = 0.5
fit <- LFabs(x, y, status, sigma, w, model, tau)
```

Index