Package 'ADSIHT'

April 17, 2025

Type Package

Title Adaptive Double Sparse Iterative Hard Thresholding

Version 0.2.0

Date 2025-4-12

Maintainer Yanhang Zhang <zhangyh98@ruc.edu.cn>

Description Solving high-dimensional double sparse linear regression via an iterative hard thresholding algorithm. Furthermore, the method is extended to jointly estimate multiple graphical models. For more details, please see <https://www.jmlr.org/papers/v25/23-0653.html> and <doi:10.48550/arXiv.2503.18722>.

License GPL (>= 3)

Depends R (>= 4.1.0)

Imports Matrix, mvnfast, Rcpp, purrr, snowfall

Suggests knitr, rmarkdown, testthat

LinkingTo Rcpp, RcppEigen

Encoding UTF-8

Language en-GB

RoxygenNote 7.3.2

NeedsCompilation yes

Author Yanhang Zhang [aut, cre],
Zhifan Li [aut],
Jianxin Yin [aut],
Shixiang Liu [aut],
Lingren Kong [aut]

Repository CRAN

Date/Publication 2025-04-17 09:50:06 UTC

Contents

ADSIHT *Adaptive Double Sparse Iterative Hard Thresholding Algorithm (ADSIHT)*

Description

An implementation of the sparse group selection in linear regression model via ADSIHT.

Usage

```
ADSIHT(
  x,
  y,
  group,
  s0,
  kappa = 0.9,
  ic.type = c("dsic", "loss"),
  ic.scale = 3,
  ic.coef = 3,
  L = 5,
  weight = rep(1, nrow(x)),
  coef1 = 1,
  coef2 = 1,
  eta = 0.8,
  max_iter = 20,
  method = "ols"
)
```

Arguments

x	Input matrix, of dimension $n \times p$; each row is an observation vector and each column is a predictor.
y	The response variable of n observations.
group	A vector indicating which group each variable belongs to For variables in the same group, they should be located in adjacent columns of x and their corresponding index in group should be the same. Denote the first group as 1, the second 2, etc.
s0	A vector that controls the degrees with group. Default is $d^(l-1)/(L-1)) : 1 \leq l \leq L$, where d is the maximum group size.
kappa	A parameter that controls the rapid of the decrease of threshold. Default is 0.9.
ic.type	The type of criterion for choosing the support size. Available options are "dsic", "loss". Default is "dsic".
ic.scale	A non-negative value used for multiplying the penalty term in information criterion. Default: ic.scale = 3.

ic.coef	A non-negative value used for multiplying the penalty term for choosing the optimal stopping time. Default: ic.coef = 3.
L	The length of the sequence of s0. Default: L = 5.
weight	The weight of the samples, with the default value set to 1 for each sample.
coef1	A positive value to control the sub-optimal stopping time.
coef2	A positive value to control the overall stopping time. A small value leads to larger search range.
eta	A parameter controls the step size in the gradient descent step. Default: eta = 0.8.
max_iter	A parameter that controls the maximum number of line search, ignored if OLS is employed.
method	Whether ols (default) or linesearch method should be employed.

Value

A list object comprising:

beta	A p-by-length(s0) matrix of coefficients, stored in column format.
intercept	A length(s0) vector of intercepts
.	
lambda	A length(s0) vector of threshold values
A_out	The selected variables given threshold value in lambda.
ic	The values of the specified criterion for each fitted model given threshold lamdba.

Author(s)

Yanhang Zhang, Zhifan Li, Shixiang Liu, Jianxin Yin.

Examples

```
n <- 200
m <- 100
d <- 10
s <- 5
s0 <- 5
data <- gen.data(n, m, d, s, s0)
fit <- ADSIHT(data$x, data$y, data$group)
fit$A_out[which.min(fit$ic)]
```

ADSIHT.ML *ADSIHT in multi-task learning framework*

Description

An implementation of the sparse group selection in linear regression model via ADSIHT.

Usage

```
ADSIHT.ML(
  x_list,
  y_list,
  group_list,
  s0,
  kappa = 0.9,
  ic.type = c("dsic", "loss"),
  ic.scale = 3,
  ic.coef = 3,
  L = 5,
  weight,
  coef1 = 1,
  coef2 = 1,
  eta = 0.8,
  max_iter = 20,
  method = "ols",
  center = TRUE,
  scale = 1
)
```

Arguments

x_list	The list of input matrix.
y_list	The list of response variable.
group_list	A vector indicating which group each variable belongs to For variables in the same group, they should be located in adjacent columns of x and their corresponding index in group should be the same. Denote the first group as 1, the second 2, etc.
s0	A vector that controls the degrees with group. Default is $d^(l-1)/(L-1)) : 1 \leq l \leq L$, where d is the maximum group size.
kappa	A parameter that controls the rapid of the decrease of threshold. Default is 0.9.
ic.type	The type of criterion for choosing the support size. Available options are "dsic", "loss". Default is "dsic".
ic.scale	A non-negative value used for multiplying the penalty term in information criterion. Default: ic.scale = 3.

ic.coef	A non-negative value used for multiplying the penalty term for choosing the optimal stopping time. Default: ic.coef = 3.
L	The length of the sequence of s0. Default: L = 5.
weight	The weight of the samples, with the default value set to 1 for each sample.
coef1	A positive value to control the sub-optimal stopping time.
coef2	A positive value to control the overall stopping time. A small value leads to larger search range.
eta	A parameter controls the step size in the gradient descent step. Default: eta = 0.8.
max_iter	A parameter that controls the maximum number of line search, ignored if OLS is employed.
method	Whether ols (default) or linesearch method should be employed.
center	A boolean value indicating whether centralization is required. Default: center = TRUE.
scale	A positive value to control the column-wise L2 norm of each observation matrix. Default: scale=1.

Value

A list object comprising:

beta	A p-by-length(s0) matrix of coefficients, stored in column format.
intercept	A length(s0) vector of intercepts.
lambda	A length(s0) vector of threshold values
A_out	The selected variables given threshold value in lambda.
ic	The values of the specified criterion for each fitted model given threshold lamdba.

Author(s)

Yanhang Zhang, Zhifan Li, Shixiang Liu, Jianxin Yin.

Examples

```
set.seed(1)
n <- 200
p <- 100
K <- 4
s <- 5
s0 <- 2
x_list <- lapply(1:K, function(x) matrix(rnorm(n*p, 0, 1), nrow = n))
vec <- rep(0, K * p)
non_sparse_groups <- sample(1:p, size = s, replace = FALSE)
for (group in non_sparse_groups) {
 group_indices <- seq(group, K * p, by = p)
 non_zero_indices <- sample(group_indices, size = s0, replace = FALSE)
 vec[non_zero_indices] <- rep(2, s0)
}
```

```
y_list <- lapply(1:K, function(i) return(
  y = x_list[[i]] %*% vec[((i-1)*p+1):(i*p)]+rnorm(n, 0, 0.5))
)
fit <- ADSIHT.ML(x_list, y_list)
fit$A_out[, which.min(fit$ic)]
```

`gen.data` *Generate simulated data*

Description

Generate simulated data for sparse group linear model.

Usage

```
gen.data(
  n,
  m,
  d,
  s,
  s0,
  cor.type = 1,
  beta.type = 1,
  rho = 0.5,
  sigma1 = 1,
  sigma2 = 1,
  seed = 1
)
```

Arguments

`n`	The number of observations.
`m`	The number of groups of interest.
`d`	The group size of each group. Only even group structure is allowed here.
`s`	The number of important groups in the underlying regression model.
`s0`	The number of important variables in each important group.
`cor.type`	The structure of correlation. `cor.type = 1` denotes the independence structure, where the covariance matrix has (i, j) entry equals $I(i \neq j)$. `cor.type = 2` denotes the exponential structure, where the covariance matrix has (i, j) entry equals $rho^{|i-j|}$. `cor.type = 3` denotes the constant structure, where the non-diagonal entries of covariance matrix are rho and diagonal entries are 1.
`beta.type`	The structure of coefficients. `beta.type = 1` denotes the homogenous setup, where each entry has the same magnitude. `beta.type = 2` denotes the heterogeneous structure, where the coefficients are drawn from a normal distribution.
`rho`	A parameter used to characterize the pairwise correlation in predictors. Default is `0.5`..

sigma1	The value controlling the strength of the gaussian noise. A large value implies strong noise. Default sigma1 = 1.
sigma2	The value controlling the strength of the coefficients. A large value implies large coefficients. Default sigma2 = 1.
seed	random seed. Default: seed = 1.

Value

A list object comprising:

x	Design matrix of predictors.
y	Response variable.
beta	The coefficients used in the underlying regression model.
group	The group index of each variable.
true.group	The important groups in the sparse group linear model.
true.variable	The important variables in the sparse group linear model.

Author(s)

Yanhang Zhang, Zhifan Li, Jianxin Yin.

Examples

```
# Generate simulated data
n <- 200
m <- 100
d <- 10
s <- 5
s0 <- 5
data <- gen.data(n, m, d, s, s0)
str(data)
```

MIGHT *MIGHT: Milti-task iterative graphical hard thresholding*

Description

An implementation of the sparse group selection in joint graphical model.

Usage

```
MIGHT(
  X,
  ic.coef = 3,
  ic.scale = 3,
  L = 15,
```

```
  coef1 = 1,
  coef2 = 0.1,
  kappa = 0.9,
  eta = 0.8,
  center = TRUE,
  scale = 1,
  parallel = FALSE,
  ncpus = 4
)
```

Arguments

X	The list of input observation matrices.
ic.coef	A non-negative value used for multiplying the penalty term for choosing the optimal stopping time. Default: ic.coef = 3.
ic.scale	A non-negative value used for multiplying the penalty term in information criterion. Default: ic.scale = 3.
L	The length of the sequence of s0. Default: L = 15.
coef1	A positive value to control the sub-optimal stopping time.
coef2	A positive value to control the overall stopping time. A small value leads to larger search range.
kappa	A parameter that controls the rapid of the decrease of threshold. Default is 0.9.
eta	A parameter controls the step size in the gradient descent step. Default: eta = 0.8.
center	A boolean value indicating whether centralization is required. Default: center = TRUE.
scale	A positive value to control the column-wise L2 norm of each observation matrix. Default: scale=1.
parallel	A boolean value indicating whether parallel operation is required. Default: parallel = FALSE.
ncpus	A positive value that controls the numer of cpus. Default: ncpus = 4.

Value

A list object containing the estimated precision matrices for each dataset.

Author(s)

Yanhang Zhang, Zhifan Li, Shixiang Liu, Jianxin Yin.

Examples

```
library(mvnfast)
set.seed(1)
n = 50; p = 10; K = 4
x_list <- lapply(1:K, function(x) rmvn(n, mu=rep(1, p),
```

```
                                        sigma = toeplitz( (x/2/K)^(1:p-1) ) ) )
fit = MIGHT(X=x_list, scale = 10)
solve( toeplitz( 0.5^(0:9) ) )
fit[[4]]
```

Index