Package 'TBFLChangePointDetection'

December 16, 2020

Type Package

Title Change point detection in high-dimensional linear regression models

Version 0.1.0

Author Yue Bai and Abolfazl Safikhani

Maintainer Yue Bai <baiyue@ufl.edu>

Description Change point detection in high-dimensional linear regression models.
Use four spaces when indenting paragraphs within the Description.

License GPL-2

Encoding UTF-8

LazyData true

Imports stats,
graphics,
mvtnorm,
factoextra,
Rcpp,
RcppArmadillo,
lattice,
xtable,
ggplot2,
glmnet,
sparsevar

LinkingTo Rcpp,
RcppArmadillo

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.0.2

R topics documented:

`BIC` *BIC and HBIC function*

Description

BIC and HBIC function

Usage

```
BIC(residual, phi, gamma.val = 1, method = "MLR")
```

Arguments

`residual`	residual matrix
`phi`	estimated coefficient matrix of the model
`gamma.val`	hyperparameter for HBIC, if HBIC == TRUE.
`method`	method name for the model: MLR: Multiple Linear Regression; VAR: Vector autoregression;

`BIC.threshold` *BIC threshold for final parameter estimation*

Description

BIC threshold for final parameter estimation

Usage

```
BIC.threshold(
  method,
  beta.final,
  k,
  m.hat,
  brk,
  data_y,
  data_x = NULL,
  b_n = floor(sqrt(T)),
  nlam = 20
)
```

Arguments

method	method name for the model: Constant: Mean-shift Model; MvLR: Multivariate Linear Regression; MLR: Multiple Linear Regression
beta.final	a combined matrix of estimated parameter coefficient matrices for all stationary segementations
k	dimensions of parameter coefficient matrices
m.hat	number of estimated change points
brk	vector of estimated change points
data_y	input data matrix (response), with each column representing the time series component
data_x	input data matrix (predictor), with each column representing the time series component
b_n	the block size
nlam	number of hyperparameters for grid search

Value

lambda.val.best, the tuning parameter lambda selected by BIC.

BIC.threshold.ggm *BIC threshold for final parameter estimation (GGM)*

Description

BIC threshold for final parameter estimation (GGM)

Usage

```
BIC.threshold.ggm(
  beta.final,
  k,
  m.hat,
  brk,
  data_y,
```

```
  data_x = NULL,
  b_n = floor(sqrt(T)),
  nlam = 20
)
```

Arguments

beta.final	a combined matrix of estimated parameter coefficient matrices for all stationary segementations
k	dimensions of parameter coefficient matrices
m.hat	number of estimated change points
brk	vector of estimated change points
data_y	input data matrix (response), with each column representing the time series component
data_x	input data matrix (predictor), with each column representing the time series component
b_n	the block size
nlam	number of hyperparameters for grid search

Value

lambda.val.best, the tuning parameter lambda selected by BIC.

---

constant.sim.break *Generate the constant model data with break points*

Description

Generate the constant model data with break points

Usage

```
constant.sim.break(nobs, cnst, sigma, brk = nobs + 1)
```

Arguments

nobs	number of time points
cnst	the constant
sigma	covariance matrix of the white noise
brk	vector of break points

Value

A list oject, which contains the followings

series_y matrix of response data

noises matrix of white noise error

`ggm.first.step.blocks` *Threshold block fused lasso step for gaussian graphical model.*

Description

Perform the block fused lasso with thresholding to detect candidate break points.

Usage

```
ggm.first.step.blocks(
  data_y,
  data_x,
  lambda1,
  lambda2,
  max.iteration = max.iteration,
  tol = tol,
  blocks,
  cv.index,
  HBIC = FALSE,
  gamma.val = NULL
)
```

Arguments

`data_y`	input data matrix Y
`data_x`	input data matrix X
`lambda1`	tuning parmaeter lambda_1 for fused lasso
`lambda2`	tuning parmaeter lambda_2 for fused lasso
`max.iteration`	max number of iteration for the fused lasso
`tol`	tolerance for the fused lasso
`blocks`	the blocks
`cv.index`	the index of time points for cross-validation
`HBIC`	logical; if TRUE, use high-dimensional BIC, if FALSE, use orginal BIC. Default is FALSE.
`gamma.val`	hyperparameter for HBIC, if HBIC == TRUE.

`ggm.second.step.search` *Exhaustive search step for gaussian graphical model.*

Description

Perform the exhaustive search to "thin out" redundant break points.

Usage

```
ggm.second.step.search(
  data_y,
  data_x,
  max.iteration = max.iteration,
  tol = tol,
  cp.first,
  beta.est,
  blocks
)
```

Arguments

data_y	input data matrix, with each column representing the time series component
data_x	input data matrix, with each column representing the time series component
max.iteration	max number of iteration for the fused lasso
tol	tolerance for the fused lasso
cp.first	the selected break points after the first step
beta.est	the estiamted parameters by block fused lasso
blocks	the blocks

Value

A list oject, which contains the followings

cp.final a set of selected break point after the exhaustive search step

beta.hat.list the estimated coefficient matrix for each segmentation

---

ggm.sim.break *Generate the gaussian graphical model data with break points*

Description

Generate the gaussian graphical model data with break points

Usage

```
ggm.sim.break(nobs, px, sigma, brk = nobs + 1)
```

Arguments

nobs	number of time points
px	the number of features
sigma	covariance matrix of the X matrix
brk	vector of break points

Value

A list oject, which contains the followings

series_x matrix of data

lambda_warm_up_lm *lambda warm up for linear regression model*

Description

lambda warm up for linear regression model

Usage

```
lambda_warm_up_lm(data_y, data_x, blocks, cv_index)
```

Arguments

data_y	input matrix Y
data_x	input matrix X
blocks	the vector of blocks
cv_index	the vector of indices for validation

Value

a value for parameter lambda

lm.first.step.blocks *Threshold block fused lasso step for linear regression model.*

Description

Perform the block fused lasso with thresholding to detect candidate break points.

Usage

```
lm.first.step.blocks(
  data_y,
  data_x,
  lambda1,
  lambda2,
  max.iteration = max.iteration,
  tol = tol,
  blocks,
  cv.index,
  fixed_index = NULL,
  nonfixed_index = NULL,
  HBIC = FALSE,
  gamma.val = NULL
)
```

Arguments

data_y	input data matrix Y, with each column representing the time series component
data_x	input data matrix X, with each column representing the time series component
lambda1	tuning parmaeter lambda_1 for fused lasso
lambda2	tuning parmaeter lambda_2 for fused lasso
max.iteration	max number of iteration for the fused lasso
tol	tolerance for the fused lasso
blocks	the blocks
cv.index	the index of time points for cross-validation
fixed_index	index for linear regression model with only partial compoenents change.
nonfixed_index	index for linear regression model with only partial compoenents change.
HBIC	logical; if TRUE, use high-dimensional BIC, if FALSE, use orginal BIC. Default is FALSE.
gamma.val	hyperparameter for HBIC, if HBIC == TRUE.

---

lm.second.step.search *Exhaustive search step for linear regression model.*

Description

Perform the exhaustive search to "thin out" redundant break points.

Usage

```
lm.second.step.search(
  data_y,
  data_x,
  max.iteration = max.iteration,
  tol = tol,
  cp.first,
  beta.est,
  blocks
)
```

Arguments

data_y	input data matrix, with each column representing the time series component
data_x	input data matrix, with each column representing the time series component
max.iteration	max number of iteration for the fused lasso
tol	tolerance for the fused lasso
cp.first	the selected break points after the first step
beta.est	the estiamted parameters by block fused lasso
blocks	the blocks

Value

A list oject, which contains the followings

cp.final a set of selected break point after the exhaustive search step

beta.hat.list the estimated coefficient matrix for each segmentation

---

`lm.sim.break` *Generate the linear regression model data with break points*

Description

Generate the linear regression model data with break points

Usage

```
lm.sim.break(
  nobs,
  px,
  cnst = NULL,
  phi = NULL,
  sigma,
  sigma_x = 1,
  brk = nobs + 1
)
```

Arguments

`nobs`	number of time points
`px`	the number of features
`cnst`	the constant
`phi`	parameter coefficient matrix of the linear model
`sigma`	covariance matrix of the white noise
`sigma_x`	variance of the predictor variable x
`brk`	vector of break points

Value

A list oject, which contains the followings

series_y matrix of response data

series_x matrix of predictor data

noises matrix of white noise error

mspe.plot *Plot the cross-validation score*

Description

Plot the cross-validation score

Usage

```
mspe.plot(pred.error, lambda)
```

Arguments

pred.error	prediction error
lambda	indice of tuning parameter lambda

pred *prediction function*

Description

prediction function

Usage

```
pred(X, phi, j, p.x, p.y, h = 1)
```

Arguments

X	data for prediction
phi	parameter matrix
j	the start time point for prediction
p.x	the dimension of data X
p.y	the dimension of data Y
h	the length of observation to predict

pred.block *Prediction function (block)*

Description

Prediction function (block)

Usage

```
pred.block(X, phi, j, p.x, p.y, h)
```

Arguments

X	data for prediction
phi	parameter matrix
j	the start time point for prediction
p.x	the dimension of data X
p.y	the dimension of data Y
h	the length of observation to predict

pred.block.var *Prediction function for VAR (block)*

Description

Prediction function for VAR (block)

Usage

```
pred.block.var(Y, phi, q, T, p, h)
```

Arguments

Y	data for prediction
phi	parameter matrix
q	the AR order
T	the start time point for prediction
p	the number of time series components
h	the length of observation to predict

pred.var *Prediction function for VAR 2*

Description

Prediction function for VAR 2

Usage

```
pred.var(Y, phi, q, T, p, h = 1)
```

Arguments

Y	data for prediction
phi	parameter matrix
q	the AR order
T	the start time point for prediction
p	the number of time series components
h	the length of observation to predict

remove.extra.pts *helper function for detection check*

Description

helper function for detection check

Usage

```
remove.extra.pts(pts, brk)
```

Arguments

pts	the estimated change points
brk	the true change points

soft_full *soft threshold function*

Description

soft threshold function

Usage

```
soft_full(L, lambda)
```

Arguments

L	input matrix
lambda	threshold parameter

Value

thresholded matrix L

tbfl *Threshold block fused lasso (TBFL) algorithm for change point detection*

Description

Perform the threshold block fused lasso (TBFL) algorithm to detect the structural breaks in large scale high-dimensional non-stationary linear regression models.

Usage

```
tbfl(
  method,
  data_y,
  data_x = NULL,
  lambda.1.cv = NULL,
  lambda.2.cv = NULL,
  q = 1,
  max.iteration = 100,
  tol = 10^(-2),
  block.size = NULL,
  blocks = NULL,
  refit = FALSE,
  fixed_index = NULL,
  HBIC = FALSE,
  gamma.val = NULL,
  optimal.block = TRUE,
  optimal.gamma.val = 1.5,
  block.range = NULL
)
```

Arguments

method	method name for the model: Constant: Mean-shift Model; MvLR: Multivariate Linear Regression; MLR: Multiple Linear Regression; VAR: Vector autoregression; GGM: Gaussian graphical model
data_y	input data matrix (response), with each column representing the time series component
data_x	input data matrix (predictor), with each column representing the time series component
lambda.1.cv	tuning parmaeter lambda_1 for fused lasso
lambda.2.cv	tuning parmaeter lambda_2 for fused lasso
q	the AR order
max.iteration	max number of iteration for the fused lasso
tol	tolerance for the fused lasso
block.size	the block size
blocks	the blocks
refit	logical; if TRUE, refit the model, if FALSE, use BIC to find a thresholding value and then output the parameter estimates without refitting. Default is FALSE.
fixed_index	index for linear regression model with only partial compoenents change.
HBIC	logical; if TRUE, use high-dimensional BIC, if FALSE, use orginal BIC. Default is FALSE.
gamma.val	hyperparameter for HBIC, if HBIC == TRUE.
optimal.block	logical; if TRUE, grid search to find optimal block size, if FALSE, directly use the default block size. Default is TRUE.
optimal.gamma.val	hyperparameter for optimal block size, if optimal.blocks == TRUE. Default is 1.5.
block.range	the search domain for optimal block size.

Value

A list oject, which contains the followings

cp.first a set of selected break point after the first block fused lasso step

cp.final a set of selected break point after the final exhaustive search step

beta.hat.list a list of estimated parameter coefficient matrices for each stationary segementation

beta.est a list of estimated parameter coefficient matrices for each block

beta.final a list of estimated parameter coefficient matrices for each stationary segementation, using BIC thresholding or refitting the model.

beta.full.final For GGM only. A list of $p \times p$ matrices for each stationary segementation. The off-diagonal entries are same as the beta.final.

jumps The change (jump) of the values in estimated parameter coefficient matrix.

bn.optimal The optimal block size.

bn.range The values of block size in grid search.

HBIC.full The HBIC values.

pts.full The selected change points for each block size.

Author(s)

Yue Bai, <baiyue@ufl.edu>

Examples

```
T <- 2*10^3; # number of observations/samples
p.y <- 1; # dimension of observed Y
p.x <- 20
brk <- c(floor(T/4), floor(2*T/4), floor(3*T/4), T+1)
m <- length(brk)
d <- 15 #number of non-zero coefficient
###generate coefficient beta
beta.full <- matrix(0, p.y, p.x*m)
set.seed(1)
aa <- c(-3, 5, -3, 3)
for(i in 1:m){beta.full[1, (i-1)*p.x+sample(1:p.x, d, replace = FALSE)] <- aa[i] + runif(d, -1, 1);}
e.sigma <- as.matrix(1*diag(p.y))
try <- lm.sim.break(nobs = T, px = p.x, phi = beta.full, sigma = e.sigma, sigma_x = 1, brk = brk)
data_y <- try$series_y; data_y <- as.matrix(data_y, ncol = p.y)
data_x <- try$series_x; data_x <- as.matrix(data_x)
method <- c("MLR")
temp <- tbfl(method, data_y, data_x)
temp$cp.final #change points
temp$beta.final #final estimated parameters (after BIC threshold)
temp_refit <- tbfl(method, data_y, data_x, refit = TRUE)
temp_refit$beta.final #final estimated parameters (refitting the model)
```

---

var.first.step.blocks *Threshold block fused lasso step for linear regression model.*

Description

Perform the block fused lasso with thresholding to detect candidate break points.

Usage

```
var.first.step.blocks(
  data_y,
  lambda1,
  lambda2,
  q,
  max.iteration,
  tol,
  blocks,
  cv.index,
  HBIC = FALSE,
  gamma.val = NULL
)
```

Arguments

data_y	input data matrix Y, with each column representing the time series component
lambda1	tuning parmaeter lambda_1 for fused lasso
lambda2	tuning parmaeter lambda_2 for fused lasso
q	the AR order
max.iteration	max number of iteration for the fused lasso
tol	tolerance for the fused lasso
blocks	the blocks
cv.index	the index of time points for cross-validation
HBIC	logical; if TRUE, use high-dimensional BIC, if FALSE, use orginal BIC. Default is FALSE.
gamma.val	hyperparameter for HBIC, if HBIC == TRUE.

---

var.second.step.search

Exhaustive search step

Description

Perform the exhaustive search to "thin out" redundant break points.

Usage

```
var.second.step.search(
  data_y,
  q,
  max.iteration = max.iteration,
  tol = tol,
  cp.first,
  beta.est,
  blocks
)
```

Arguments

data_y	input data matrix, with each column representing the time series component
q	the AR order
max.iteration	max number of iteration for the fused lasso
tol	tolerance for the fused lasso
cp.first	the selected break points after the first step
beta.est	the estiamted parameters by block fused lasso
blocks	the blocks

Value

A list oject, which contains the followings

cp.final a set of selected break point after the exhaustive search step

phi.hat.list the estimated coefficient matrix for each segmentation

`var.sim.break` *Generating non-stationary ARMA data.*

Description

Generating non-stationary ARMA data.

Usage

```
var.sim.break(
  nobs,
  arlags = NULL,
  malags = NULL,
  cnst = NULL,
  phi = NULL,
  theta = NULL,
  skip = 200,
  sigma,
  brk = nobs + 1
)
```

Arguments

`nobs`	number of time points
`arlags`	the true AR order
`malags`	the true MA order
`cnst`	the constant
`phi`	parameter matrix of the AR model
`theta`	parameter matrix of the MA model
`skip`	the number of time points to skip at the begining (for stable data)
`sigma`	covariance matrix of the white noise
`brk`	vector of break points

Value

Matrice of time series data and white noise data

Index