Package ‘hdbinseg’

October 13, 2022

Type Package

Title Change-Point Analysis of High-Dimensional Time Series via Binary Segmentation

Version 1.0.1

Date 2018-05-18

Description Binary segmentation methods for detecting and estimating multiple change-points in the mean or second-order structure of high-dimensional time series as described in Cho and Fryzlewicz (2014) <doi:10.1111/rssb.12079> and Cho (2016) <doi:10.1214/16-EJS1155>.

Depends R (>= 3.4.0)

License GPL (>= 3)

LazyData TRUE

Suggests RcppArmadillo

Imports Rcpp (>= 0.12.10), foreach, iterators, doParallel

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

NeedsCompilation yes

Author Haeran Cho [aut, cre],
Piotr Fryzlewicz [aut]

Maintainer Haeran Cho <`haeran.cho@bristol.ac.uk`>

Repository CRAN

Date/Publication 2018-05-29 08:59:41 UTC

R topics documented:

`dcbs.alg` *Double CUSUM Binary Segmentation*

Description

Perform the Double CUSUM Binary Segmentation algorithm detecting change-points in the mean or second-order structure of the data.

Usage

```
dcbs.alg(x, cp.type = c(1, 2)[1], phi = 0.5, thr = NULL, trim = NULL,
  height = NULL, temporal = TRUE, scales = NULL, diag = FALSE,
  B = 1000, q = 0.01, do.parallel = 4)
```

Arguments

`x`	input data matrix, with each row representing the component time series
`cp.type`	`cp.type=1` specifies change-points in the mean, `cp.type=2` specifies change-points in the second-order structure
`phi`	choice of parameter for weights in Double CUSUM statistic; 0 <= phi <= 1 or phi = -1 allowed with the latter leading to the DC statistic combining phi = 0 and phi = 1/2, see Section 4.1 of Cho (2016) for further details
`thr`	pre-defined threshold values; when `thr = NULL`, bootstrap procedure is employed for the threshold selection; when `thr != NULL` and `cp.type = 1`, `length(thr)` should be one, if `cp.type = 2`, `length(thr)` should match `length(scales)`
`trim`	length of the intervals trimmed off around the change-point candidates; `trim = NULL` activates the default choice (`trim = round(log(dim(x)[2]))`)
`height`	maximum height of the binary tree; `height = NULL` activates the default choice (`height = floor(log(dim(x)[2], 2)/2)`)
`temporal`	used when `cp.type = 1`; if `temporal = FALSE`, rows of `x` are scaled by mad estimates, if `temporal = TRUE`, their long-run variance estimates are used
`scales`	used when `cp.type = 2`; negative integers representing Haar wavelet scales to be used for computing `nrow(x)*(nrow(x)+1)/2` dimensional wavelet transformation of `x`; a small negative integer represents a fine scale
`diag`	used when `cp.type = 2`; if `diag = TRUE`, only changes in the diagonal elements of the autocovariance matrices are searched for
`B`	used when `is.null(thr)`; number of bootstrap samples for threshold selection
`q`	used when `is.null(thr)`; indicates the quantile of bootstrap test statistics to be used for threshold selection
`do.parallel`	used when `is.null(thr)`; number of copies of R running in parallel, if `do.parallel = 0`, %do% operator is used, see also foreach

Value

S3 `bin.tree` object, which contains the following fields:

`tree`	a list object containing information about the nodes at which change-points are detected
`mat`	matrix concatenation of the nodes of `tree`
`ecp`	estimated change-points
`thr`	threshold used to construct the tree

References

H. Cho (2016) Change-point detection in panel data via double CUSUM statistic. *Electronic Journal of Statistics*, vol. 10, pp. 2000–2038.

Examples

```
x <- matrix(rnorm(10*100), nrow=10)
dcbs.alg(x, cp.type=1, phi=.5, temporal=FALSE, do.parallel=0)$ecp

x <- matrix(rnorm(100*300), nrow=100)
x[1:10, 151:300] <- x[1:10, 151:300] + 1
dcbs.alg(x, cp.type=1, phi=-1, temporal=FALSE, do.parallel=0)$ecp
```

`dcbs.thr` *Bootstrapping for threshold selection in DCBS algorithm*

Description

Generate thresholds for DCBS algorithm via bootstrapping

Usage

```
dcbs.thr(z, interval = c(1, dim(z)[2]), phi = 0.5, cp.type = 1,
  do.clean.cp = FALSE, temporal = TRUE, scales = NULL, diag = FALSE,
  sgn = NULL, B = 1000, q = 0.01, do.parallel = 4)
```

Arguments

`z`	input data matrix, with each row representing the component time series
`interval`	a vector of two containing the start and the end points of the interval from which the bootstrap test statistics are to be calculated
`phi, cp.type, temporal, scales, diag, B, q, do.parallel`	see `dcbs.alg`
`do.clean.cp`	if `do.clean.cp = TRUE` pre-change-point cleaning is performed
`sgn`	if `diag = FALSE`, wavelet transformations of the cross-covariances are computed with the matching signs

Value

a numeric value for the threshold

`sbs.alg` *Sparsified Binary Segmentation*

Description

Perform the Sparsified Binary Segmentation algorithm detecting change-points in the mean or second-order structure of the data.

Usage

```
sbs.alg(x, cp.type = c(1, 2)[1], thr = NULL, trim = NULL, height = NULL,
  temporal = TRUE, scales = NULL, diag = FALSE, B = 1000, q = 0.01,
  do.parallel = 4)
```

Arguments

`x`	input data matrix, with each row representing the component time series
`cp.type`	`cp.type=1` specifies change-points in the mean, `cp.type=2` specifies change-points in the second-order structure
`thr`	pre-defined threshold values; when `thr = NULL`, bootstrap procedure is employed for the threshold selection; when `thr != NULL` and `cp.type = 1`, `length(thr)` should match `nrow(x)`, if `cp.type = 2`, `length(thr)` should match `nrow(x)*(nrow(x)+1)/2*length(s`
`trim`	length of the intervals trimmed off around the change-point candidates; `trim = NULL` activates the default choice (`trim = round(log(dim(x)[2]))`)
`height`	maximum height of the binary tree; `height = NULL` activates the default choice (`height = floor(log(dim(x)[2], 2)/2)`)
`temporal`	used when `cp.type = 1`; if `temporal = FALSE`, rows of `x` are scaled by mad estimates, if `temporal = TRUE`, their long-run variance estimates are used
`scales`	used when `cp.type = 2`; negative integers representing Haar wavelet scales to be used for computing `nrow(x)*(nrow(x)+1)/2` dimensional wavelet transformation of `x`; a small negative integer represents a fine scale
`diag`	used when `cp.type = 2`; if `diag = TRUE`, only changes in the diagonal elements of the autocovariance matrices are searched for
`B`	used when `is.null(thr)`; number of bootstrap samples for threshold selection
`q`	used when `is.null(thr)`; quantile of bootstrap test statistics to be used for threshold selection
`do.parallel`	used when `is.null(thr)`; number of copies of R running in parallel, if `do.parallel = 0`, %do% operator is used, see also foreach

Value

S3 `bin.tree` object, which contains the following fields:

`tree`	a list object containing information about the nodes at which change-points are detected
`mat`	matrix concatenation of the nodes of `tree`
`ecp`	estimated change-points
`thr`	threshold used to construct the tree

References

H. Cho and P. Fryzlewicz (2014) Multiple-change-point detection for high dimensional time series via sparsified binary segmentation. *JRSSB*, vol. 77, pp. 475–507.

Examples

```
x <- matrix(rnorm(20*300), nrow=20)
sbs.alg(x, cp.type=2, scales=-1, diag=TRUE, do.parallel=0)$ecp

x <- matrix(rnorm(100*300), nrow=100)
x[1:10, 151:300] <- x[1:10, 151:300]*sqrt(2)
sbs.alg(x, cp.type=2, scales=-1, diag=TRUE, do.parallel=0)$ecp
```

`sbs.thr` *Bootstrapping for threshold selection in SBS algorithm*

Description

Generate thresholds for SBS algorithm via bootstrapping

Usage

```
sbs.thr(z, interval = c(1, dim(z)[2]), cp.type = 1, do.clean.cp = TRUE,
  scales = NULL, diag = FALSE, sgn = NULL, B = 1000, q = 0.01,
  do.parallel = 4)
```

Arguments

`z`	input data matrix, with each row representing the component time series
`interval`	a vector of two containing the start and the end points of the interval from which the bootstrap test statistics are to be calculated
`cp.type, scales, diag, B, q, do.parallel`	see `sbs.alg`
`do.clean.cp`	if `do.clean.cp = TRUE` pre-change-point cleaning is performed
`sgn`	if `diag = FALSE`, wavelet transformations of the cross-covariances are computed with the matching signs

Value

if `cp.type = 1`, a vector of length `nrow(z)`, each containing the threshold applied to the CUSUM statistics from the corresponding coordinate of `z` if `cp.type = 2`, a vector of length `length(scales)*nrow(z)` (when `diag = TRUE`) or `length(scales)*nrow(z)*(nrow(z)+1)/2` (when `diag = FALSE`), each containing the threshold applied to the CUSUM statistics of the corresponding wavelet transformation of `z`

Index