Package 'SNSeg'

May 7, 2026

Title Self-Normalization(SN) Based Change-Point Estimation for Time Series

Version 1.0.3

Description Implementations self-normalization (SN) based algorithms for change-points estimation in time series data. This comprises nested local-window algorithms for detecting changes in both univariate and multivariate time series developed in Zhao, Jiang and Shao (2022) <doi:10.1111/rssb.12552>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Depends R (>= 3.5.0), stats, utils, graphics

LinkingTo Rcpp

Imports Rcpp, mvtnorm

Suggests rmarkdown, knitr

VignetteBuilder knitr

NeedsCompilation yes

Author Shubo Sun [aut],
Zifeng Zhao [aut, cre],
Feiyu Jiang [aut],
Xiaofeng Shao [aut]

Maintainer Zifeng Zhao <zzhao2@nd.edu>

Repository CRAN

Date/Publication 2024-06-02 20:10:02 UTC

Contents

`critical_values_HD` *Critical Values of Self-Normalization (SN) based test statistic for changes in high-dimensional means (SNHD)*

Description

A dataset containing the critical value of SN-based change point estimates based on changes in high-dimensional means.

Usage

```
critical_values_HD
```

Format

A data frame with 6 variables:

`epsilon` value used to compute grid_size_scale and SN-based test statistic

`0.9` critical value at confidence level `0.9`

`0.95` critical value at confidence level `0.95`

`0.99` critical value at confidence level `0.99`

`0.995` critical value at confidence level `0.995`

`0.999` critical value at confidence level `0.999`

critical_values_multi *Critical Values of Self-Normalization (SN) based test statistic for changes in multiple parameters (SNCP)*

Description

A dataset containing the critical value of SN-based change point estimates based on simultaneous changes in multiple parameters.

Usage

```
critical_values_multi
```

Format

A data frame with 7 variables:

`epsilon` value used to compute grid_size_scale and SN-based test statistic

`p` dimension of the multi-parameters

`0.9` critical value at confidence level `0.9`

`0.95` critical value at confidence level `0.95`

`0.99` critical value at confidence level `0.99`

`0.995` critical value at confidence level `0.995`

`0.999` critical value at confidence level `0.999`

critical_values_single *Critical Values of Self-Normalization (SN) based test statistic for the change in a single parameter (SNCP)*

Description

A dataset containing the critical value for SN-based change point estimates based on the change in a single parameter.

Usage

```
critical_values_single
```

Format

A data frame with 6 variables:

`epsilon` value used to compute grid_size_scale and SN-based test statistic

`0.9` critical value at confidence level `0.9`

`0.95` critical value at confidence level `0.95`

`0.99` critical value at confidence level `0.99`

`0.995` critical value at confidence level `0.995`

`0.999` critical value at confidence level `0.999`

`MAR` *A funtion to generate a multivariate autoregressive process (MAR) in time series*

Description

The function `MAR` is used for generating MAR model(s) for examples of the functions `SNSeg_Uni`, `SNSeg_Multi`, and `SNSeg_HD`.

Usage

```
MAR(n, reptime, rho)
```

Arguments

`n`	the size (length) of time series to be generated
`reptime`	the number of time series to be generated
`rho`	value of autocorrelation

Value

Returns a matrix of the simulated MAR processes. The number of columns of this matrix is equivalent to the value of input argument `reptime`, and the number of rows is the value of input argument `n`.

Examples

```
MAR(n = 1000, reptime = 2, rho = -0.7)
```

MAR_MTS_Covariance *A Funtion to generate a multivariate autoregressive process (MAR) model in time series. It is used for testing change-points based on the change in multivariate means or multivariate covariance for multivariate time series. It also works for the change in correlations between two univariate time series.*

Description

The function MAR_MTS_Covariance is used to generate MAR model(s) for examples of the functions SNSeg_Uni, SNSeg_Multi, and SNSeg_HD.

Usage

```
MAR_MTS_Covariance(n, reptime, rho_sets, cp_sets, sigma_cross)
```

Arguments

n	the size of time series to be generated.
reptime	the number of time series to be generated.
rho_sets	autocorrelations for each univariate time series.
cp_sets	numeric values of the true change-point locations (0, change-point locations and the end point).
sigma_cross	a list of matrices to generate the multivariate covariance matrices.

Value

Returns a list of matrices where each matrix is a MAR process. The number of columns for each sub-matrix is equivalent to the value of input argument reptime.

Examples

```
n <- 1000
reptime <- 2
sigma_cross <- list(4*matrix(c(1,0.8,0.8,1), nrow=2),
                    matrix(c(1,0.2,0.2,1), nrow=2),
                    matrix(c(1,0.8,0.8,1), nrow=2))
cp_sets <- round(c(0,n/3,2*n/3,n))
noCP <- length(cp_sets)-2
rho_sets <- rep(0.5, noCP+1)
MAR_MTS_Covariance(n, reptime, rho_sets, cp_sets, sigma_cross)
```

`MAR_Variance` *A funtion to generate a multivariate autoregressive process (MAR) model in time series for testing change points based on variance and autocovariance*

Description

The function `MAR_Variance` is used for generating MAR model(s) for examples of the functions `SNSeg_Uni`, `SNSeg_Multi`, and `SNSeg_HD`.

Usage

```
MAR_Variance(reptime, type = "V3")
```

Arguments

`reptime`	The number of time series to be generated
`type`	The type of time series for simulation, which includes V1, V2, V3 , A1, A2 and A3. The V-beginnings are for testing the variance, and the A-beginnings are for testing the autocorrelation. The simulated time series come from supplement of Zhao et al. (2022) doi:10.1111/rssb.12552. Default `type` is `V3`. The time length and "true change-points locations" (cps) for each `type` are as follows: `V1`: cps at 400 and 750 with a time length of 1024. `V2`: cps at 125, 532 and 704 with a time length of 1024. `V3`: cps at 512 and 768 with a time length of 1024. `A1`: cps at 400 and 750 with a time length of 1024. `A2`: cps at 50 with a time length of 1024. `A3`: cps at 512 and 768 with a time length of 1024.

Value

Returns a matrix of the simulated MAR processes. The number of columns of this matrix is equivalent to the value of input argument `reptime`.

Examples

```
MAR_Variance(reptime = 2, type = "V1")
```

`max_SNsweep` *SN-based test statistic segmentation plot for univariate, mulitivariate and high-dimensional time series*

Description

The function `max_SNsweep` allows users to compute and plot the SN-based test statistics along with the identified change-points from functions SNSeg_Uni, SNSeg_Multi, or SNSeg_HD.

Usage

```
max_SNsweep(SN_result, plot_SN = TRUE, est_cp_loc = TRUE, critical_loc = TRUE)
```

Arguments

SN_result	The output of functions SNSeg_Uni, SNSeg_Multi or SNSeg_HD.
plot_SN	A boolean value to return an SN-based segmentation plot if plot_SN = TRUE.
est_cp_loc	A boolean value to plot a red solid vertical line for estimated change-point locations if est_cp_loc = TRUE.
critical_loc	A boolean value to plot a blue dashed horizontal line for the critical value if critical_loc = TRUE

Value

Returns a vector of numeric values of calculated SN-based statistics for each time point. It also generates a SN-based test statistics segmentation plot with the estimated change-points.

For more examples of max_SNsweep please see the SNSeg vignette: vignette("SNSeg", package = "SNSeg")

Examples

```
set.seed(7)
n <- 2000
reptime <- 2
cp_sets <- round(n*c(0,cumsum(c(0.5,0.25)),1))
mean_shift <- c(0.4,0,0.4)
rho <- -0.7
ts <- MAR(n, reptime, rho)
no_seg <- length(cp_sets)-1
for(index in 1:no_seg){
  tau1 <- cp_sets[index]+1
  tau2 <- cp_sets[index+1]
  ts[tau1:tau2,] <- ts[tau1:tau2,] + mean_shift[index]
}
ts <- ts[,2]
result <- SNSeg_Uni(ts, paras_to_test = "mean", confidence = 0.9,
                    grid_size_scale = 0.05, grid_size = 116,
                    plot_SN = FALSE, est_cp_loc = FALSE)

# Generate SN-based test statistic segmentation plot
# To get the computed SN-based statistics, please run the command "test_stat"
test_stat <- max_SNsweep(result, plot_SN = TRUE, est_cp_loc = TRUE,
                         critical_loc = TRUE)

# For more examples of \code{max_SNsweep} see the help vignette:
# \code{vignette("SNSeg", package = "SNSeg")}
```

`plot.SNSeg_HD` *Plotting the output for high-dimensional time series with dimension greater than 10*

Description

Plotting method for S3 objects of class `SNSeg_HD`

Usage

```
## S3 method for class 'SNSeg_HD'
plot(x, cpts.col = "red", ts_index = c(1:5), ...)
```

Arguments

`x`	a `SNSeg_HD` object
`cpts.col`	a specification for the color of the vertical lines at the change point estimators, see par
`ts_index`	The index number(s) of the univariate time series to be plotted. Users should enter a positive integer or a vector of positive integers that are no greater than the dimension of the input time series. The default is the first 5 time series, i.e., `ts_index = c(1:5)`.
`...`	additional graphical arguments, see plot and abline

Details

The location of each change point estimator is plotted as a vertical line against the input time series.

Examples

```
n <- 500
p <- 50
nocp <- 5
cp_sets <- round(seq(0,nocp+1,1)/(nocp+1)*n)
num_entry <- 5
kappa <- sqrt(4/5)
mean_shift <- rep(c(0,kappa),100)[1:(length(cp_sets)-1)]
set.seed(1)
ts <- matrix(rnorm(n*p,0,1),n,p)
no_seg <- length(cp_sets)-1
for(index in 1:no_seg){
  tau1 <- cp_sets[index]+1
  tau2 <- cp_sets[index+1]
  ts[tau1:tau2,1:num_entry] <- ts[tau1:tau2,1:num_entry] +
    mean_shift[index]
}

# grid_size defined
```

```
result <- SNSeg_HD(ts, confidence = 0.9, grid_size_scale  = 0.05,
                   grid_size = 40)
# plot the 1st, 3rd and 5th time series
plot(result, cpts.col = 'red', ts_index = c(1,3,5))
```

`plot.SNSeg_Multi` *Plotting the output for multivariate time series with dimension no greater than 10*

Description

Plotting method for S3 objects of class `SNSeg_Multi`

Usage

```
## S3 method for class 'SNSeg_Multi'
plot(x, cpts.col = "red", ...)
```

Arguments

`x`	a `SNSeg_Multi` object
`cpts.col`	a specification for the color of the vertical lines at the change point estimators, see par
`...`	additional graphical arguments, see plot and abline

Details

The location of each change point estimator is plotted as a vertical line against the input time series.

Examples

```
# Please run this function before simulation
exchange_cor_matrix <- function(d, rho){
  tmp <- matrix(rho, d, d)
  diag(tmp) <- 1
  return(tmp)
}

# simulation of multivariate time series
library(mvtnorm)
set.seed(10)
d <- 5
n <- 600
nocp <- 5
cp_sets <- round(seq(0, nocp+1 ,1)/(nocp+1)*n)
mean_shift <- rep(c(0,2),100)[1:(length(cp_sets)-1)]/sqrt(d)
rho_sets <- 0.2
```

```
sigma_cross <- list(exchange_cor_matrix(d,0))
ts <- MAR_MTS_Covariance(n, 2, rho_sets, cp_sets = c(0,n), sigma_cross)
ts <- ts[1][[1]]

# Test for the change in multivariate means
# grid_size defined
result <- SNSeg_Multi(ts, paras_to_test = "mean", confidence = 0.99,
                      grid_size_scale = 0.05, grid_size = 45)
# plot method
plot(result)
```

`plot.SNSeg_Uni` *Plotting the output for univariate or bivariate time series (testing the change in correlation between bivariate time series)*

Description

Plotting method for S3 objects of class `SNSeg_Uni`

Usage

```
## S3 method for class 'SNSeg_Uni'
plot(x, cpts.col = "red", ...)
```

Arguments

`x`	a `SNSeg_Uni` object
`cpts.col`	a specification for the color of the vertical lines at the change point estimators, see par
`...`	additional graphical arguments, see plot and abline. Users are allowed to enter their own title for the univariate time series plot. The bivariate time series does not contain this option.

Details

The location of each change point estimator is plotted as a vertical line against the input time series.

Examples

```
set.seed(7)
ts <- MAR_Variance(2, "V1")
ts <- ts[,2]
# test the change in a single parameter (variance)
# grid_size defined
result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                    grid_size_scale = 0.05, grid_size = 67,
```

```
                    plot_SN = FALSE, est_cp_loc = TRUE)
plot(result, cpts.col='red')
```

---

print.SNSeg_HD *Print SN-based change-point estimates for high-dimensional time series with dimension greater than 10*

Description

Print method for objects of class SNSeg_HD

Usage

```
## S3 method for class 'SNSeg_HD'
print(x, ...)
```

Arguments

x	a SNSeg_HD object
...	not in use

Examples

```
n <- 500
p <- 50
nocp <- 5
cp_sets <- round(seq(0,nocp+1,1)/(nocp+1)*n)
num_entry <- 5
kappa <- sqrt(4/5)
mean_shift <- rep(c(0,kappa),100)[1:(length(cp_sets)-1)]
set.seed(1)
ts <- matrix(rnorm(n*p,0,1),n,p)
no_seg <- length(cp_sets)-1
for(index in 1:no_seg){
  tau1 <- cp_sets[index]+1
  tau2 <- cp_sets[index+1]
  ts[tau1:tau2,1:num_entry] <- ts[tau1:tau2,1:num_entry] +
    mean_shift[index]
}

# grid_size defined
result <- SNSeg_HD(ts, confidence = 0.9, grid_size_scale  = 0.05,
                   grid_size = 40)
# print method
print(result)
```

print.SNSeg_Multi *Print SN-based change-point estimates for multivariate time series with dimension no greater than 10*

Description

Print method for objects of class SNSeg_Multi

Usage

```
## S3 method for class 'SNSeg_Multi'
print(x, ...)
```

Arguments

x	a SNSeg_Multi object
...	not in use

Examples

```
# Please run this function before simulation
exchange_cor_matrix <- function(d, rho){
  tmp <- matrix(rho, d, d)
  diag(tmp) <- 1
  return(tmp)
}

# simulation of multivariate time series
library(mvtnorm)
set.seed(10)
d <- 5
n <- 600
nocp <- 5
cp_sets <- round(seq(0, nocp+1 ,1)/(nocp+1)*n)
mean_shift <- rep(c(0,2),100)[1:(length(cp_sets)-1)]/sqrt(d)
rho_sets <- 0.2
sigma_cross <- list(exchange_cor_matrix(d,0))
ts <- MAR_MTS_Covariance(n, 2, rho_sets, cp_sets = c(0,n), sigma_cross)
ts <- ts[1][[1]]

# Test for the change in multivariate means
# grid_size defined
result <- SNSeg_Multi(ts, paras_to_test = "mean", confidence = 0.99,
                      grid_size_scale = 0.05, grid_size = 45)
# print method
print(result)
```

`print.SNSeg_Uni` *Print SN-based change-point estimates for univariate or bivariate time series (testing the change in correlation between bivariate time series)*

Description

Print method for objects of class `SNSeg_Uni`

Usage

```
## S3 method for class 'SNSeg_Uni'
print(x, ...)
```

Arguments

`x`	a `SNSeg_Uni` object
`...`	not in use

Examples

```
set.seed(7)
ts <- MAR_Variance(2, "V1")
ts <- ts[,2]
# test the change in a single parameter (variance)
# grid_size defined
result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                    grid_size_scale = 0.05, grid_size = 67,
                    plot_SN = FALSE, est_cp_loc = TRUE)
print(result)
```

`SNSeg` *SNSeg: An R Package for Time Series Segmentation via Self-Normalization (SN)*

Description

The SNSeg package provides three functions for multiple change point estimation using SN-based algorithms: `SNSeg_Uni`, `SNSeg_Multi` and `SNSeg_HD`. Three critical value tables (`critical_values_single`, `critical_values_multi` and `critical_values_HD`) were attached. Functions `MAR`, `MAR_Variance` and `MAR_MTS_Covariance` can be utilized to generate time series data that are used for the functions `SNSeg_Uni`, `SNSeg_Multi` and `SNSeg_HD`. S3 methods plot(), print() and summary() are available for class "SNSeg_Uni", "SNSeg_Multi" and "SNSeh_HD" objects. The function `max_SNsweep` enables users to compute the SN test statistic and make the segmentation plot for these statistics. The function `SNSeh_estimate` allows users to compute parameter estimates of each segment that is separated by estimated change-points.

SNSeg_Uni

SNSeg_Uni provides SN-based change point estimates for a univariate time series based on changes in a single parameter or multiple parameters.

For the parameters of the SN test, the function SNSeg_Uni offers mean, variance, acf, bivariate correlation and numeric quantiles as available options. It also allows users to enter their own defined function as the input parameter. Besides, users can use a composite set of parameters including one or more from the mean, variance, acf or numeric quantiles quantile. To visualize the estimated change points, users can set "plot_SN = TRUE" and "est_cp_loc = TRUE" to generate the time series segmentation plot. The output comprises of the parameter(s), the window size, and the estimated change point locations. The function returns an S3 object of class "SNSeg_Uni", which can be applied to S3 methods plot(), print() and summary().

SNSeg_Multi

SNSeg_Multi provides SN-based change point estimates for multivariate time series based on changes in multivariate means or covariance matrix. The "plot_SN = TRUE" option allows users to plot each individual time series and the estimated change=points. The function returns an S3 object of class "SNSeg_Multi", which can be applied to S3 methods plot(), print() and summary().

SNSeg_HD

SNSeg_HD provides SN-based change point estimates for a high-dimensional time series based on changes in high-dimensional means. The "plot_SN = TRUE" option allows users to plot each individual time series and the estimated change=points. The input argument "n_plot" enables users to plot the first "n_plot" number of time series. The function returns an S3 object of class "SNSeg_HD", which can be applied to S3 methods plot(), print() and summary().

max_SNsweep

max_SNsweep provides SN based test statistic of each time point and generates a plot for these statistics and the estimated change-points.

SNSeg_estimate

SNSeg_estimate computes the parameter estimates of each segment separated by the estimated change-points.

critical values table

The package SNSeg provides three critical values table.

Table critical_values_single tabulates critical values of SN-based change point estimates based on the change in a single parameter.

Table critical_values_multi tabulates critical values of SN-based change point estimates based on changes in multiple parameters.

Table critical_values_HD tabulates critical values of of SN-based change point estimates based on changes in high-dimensional means.

SNSeg_estimate *Parameter estimates of each segment separated by Self-Normalization (SN) based change-point estimates*

Description

The function SNSeg_estimate computes parameter estimates of each segment that are separated by the SN-based change-point estimates.

Usage

```
SNSeg_estimate(SN_result)
```

Arguments

SN_result	An S3 object served as the output of the functions SNSeg_Uni, SNSeg_Multi, or SNSeg_HD.

Value

SNSeg_estimate returns an S3 object of class "SNSeg_estimate" including the parameter estimates of each segment separated by the SN-based change-point estimates.

1. If the time series is univariate, for a single parameter change, the output contains parameter estimates for one of the followings: mean, variance, acf, quantile, or general, which can be referred to the change in a single mean, variance, autocorrelation, a given quantile level, or a general functional. For multi-parameter changes, the output can be a combination of mean, variance, acf, and a dataframe with each quantile level depending on the type of parameters (argument paras_to_test of SNSeg_Uni, SNSeg_Multi, or SNSeg_HD) that users select.
2. If the time series is multivariate with a dimension no greater than 10, the output contains parameter estimates for one of the followings: bivcor, multi_mean, or covariance, which can be referred to the change in correlation between bivariate time series and the change in multivariate means or covariance between multivariate time series.
3. If the time series is high-dimensional with a dimension greater than 10, the output contains the parameter estimate HD_mean to represent the change in high-dimensional means.

For more examples of SNSeg_estimate see the help vignette: vignette("SNSeg", package = "SNSeg")

Examples

```
# code to simulate a univariate time series
set.seed(7)
ts <- MAR_Variance(2, "V1")
ts <- ts[,2]
# test the change in a single parameter (variance)
# grid_size defined
result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
```

```
                    grid_size_scale = 0.05, grid_size = 67,
                    plot_SN = TRUE, est_cp_loc = TRUE)
# estimated change-point locations
result$est_cp
# variance estimates of the separated segments
SNSeg_estimate(SN_result = result)

# For more examples of SNSeg_estimate, please run
# the command: vignette("SNSeg", package = "SNSeg")
```

SNSeg_HD *Self-normalization (SN) based change points estimation for high dimensional time series for changes in high-dimensional means (SNHD).*

Description

The function SNSeg_HD is a SNHD change point estimation procedure.

Usage

```
SNSeg_HD(
  ts,
  confidence = 0.9,
  grid_size_scale = 0.05,
  grid_size = NULL,
  plot_SN = FALSE,
  est_cp_loc = TRUE,
  ts_index = c(1:5)
)
```

Arguments

ts	A high-dimensional time series represented as a matrix with p columns, where each column is a univariate time series. The dimension p for ts should be at least 10.
confidence	Confidence level of SN tests as a numeric value. Available choices of confidence levels contain 0.9, 0.95, 0.99, 0.995 and 0.999. The default is set to 0.9.
grid_size_scale	numeric value of the trimming parameter and only in use if grid_size = NULL. Users are allowed to choose any grid_size_scale between 0.05 and 0.5. A warning will be given if it is outside the range.
grid_size	Local window size h to compute the critical value for SN test. Since grid_size = n*grid_size_scale, where n is the length of time series, this function will compute the grid_size_scale by diving n from grid_size when it is not NULL.

plot_SN	Boolean value to plot the time series or not. The default setting is FALSE.
est_cp_loc	Boolean value to plot a red solid vertical line for estimated change-point locations if est_cp_loc = TRUE.
ts_index	The index number(s) of the univariate time series to be plotted. Users should enter a positive integer or a vector of positive integers that are no greater than the dimension of the input time series. The default is the first 5 time series, i.e., ts_index = c(1:5).

Value

SNSeg_HD returns an S3 object of class "SNSeg_HD" including the time series, the local window size to cover a change point, the estimated change-point locations, the confidence level and the critical value of the SN test. It also generates time series segmentation plot when plot_SN = TRUE.

ts A numeric matrix of the input time series.

grid_size A numeric value of the window size.

SN_sweep_result A list of n matrices where each matrix consists of four columns: (1) SN-based test statistic for each change-point location (2) Change-point location (3) Lower bound of the window h and (4) Upper bound of the window h.

est_cp A vector containing the locations of the estimated change-points.

confidence Confidence level of SN test as a numeric value.

critical_value Critical value of the SN-based test statistic.

Users can apply the functions summary.SN to compute the parameter estimate of each segment separated by the detected change-points. An additional function plot.SN can be used to plot the time series with estimated change-points. Users can set the option plot_SN = TRUE or use the function plot.SN to plot the time series.

It deserves to note that some change-points could be missing due to the constraint on grid_size_scale or related grid_size that grid_size_scale has a minimum value of 0.05. Therefore, SNCP claims no change-points within the first ngrid_size_scale *or the last n*grid_size_scale time points. This is a limitation of the function SNSeg_HD.

For more examples of SNSeg_HD see the help vignette: vignette("SNSeg", package = "SNSeg")

Examples

```
n <- 500
p <- 50
nocp <- 5
cp_sets <- round(seq(0,nocp+1,1)/(nocp+1)*n)
num_entry <- 5
kappa <- sqrt(4/5)
mean_shift <- rep(c(0,kappa),100)[1:(length(cp_sets)-1)]
set.seed(1)
ts <- matrix(rnorm(n*p,0,1),n,p)
no_seg <- length(cp_sets)-1
for(index in 1:no_seg){
  tau1 <- cp_sets[index]+1
  tau2 <- cp_sets[index+1]
```

```
  ts[tau1:tau2,1:num_entry] <- ts[tau1:tau2,1:num_entry] +
    mean_shift[index]
}

# grid_size defined
result <- SNSeg_HD(ts, confidence = 0.9, grid_size_scale  = 0.05,
                   grid_size = 40)
# Estimated change-point locations
result$est_cp

# For more examples, please run the following command:
# vignette("SNSeg", package = "SNSeg")
```

`SNSeg_Multi` *Self-normalization (SN) based change points estimation for multivariate time series*

Description

The function `SNSeg_Multi` is a SN-based change-points estimation procedure for a multivariate time series based on changes in the multivariate means or covariance matrix.

Usage

```
SNSeg_Multi(
  ts,
  paras_to_test = "mean",
  confidence = 0.9,
  grid_size_scale = 0.05,
  grid_size = NULL,
  plot_SN = FALSE,
  est_cp_loc = TRUE
)
```

Arguments

`ts`	A multivariate time series represented as a matrix with p columns, where each column is a univariate time series. The dimension p for ts should be at least 2.
`paras_to_test`	Type of the parameter as a string for which SN algorithms test. Available choices include `mean` and `covariance`.
`confidence`	Confidence level of SN tests as a numeric value. Available choices of confidence levels contain 0.9, 0.95, 0.99, 0.995 and 0.999. The default is set to 0.9.
`grid_size_scale`	numeric value of the trimming parameter and only in use if grid_size = NULL. Users are allowed to choose any grid_size_scale between 0.05 and 0.5. A warning will be given if it is outside the range.

grid_size	Local window size h to compute the critical value for SN test. Since grid_size = n*grid_size_scale, where n is the length of time series, this function will compute the grid_size_scale by diving n from grid_size when it is not NULL.
plot_SN	Boolean value to plot the time series or not. The default setting is FALSE.
est_cp_loc	Boolean value to plot a red solid vertical line for estimated change-point locations if est_cp_loc = TRUE

Value

SNSeg_Multi returns an S3 object of class "SNSeg_Multi" including the time series, the type of parameter to be tested, the local window size to cover a change point, the estimated change-point locations, the confidence level and the critical value of the SN test. It also generates time series segmentation plot when plot_SN = TRUE.

ts A numeric matrix of the input time series.

paras_to_test the parameter used for the SN test as character.

grid_size A numeric value of the window size.

SN_sweep_result A list of n matrices where each matrix consists of four columns: (1) SN-based test statistic for each change-point location (2) Change-point location (3) Lower bound of the window h and (4) Upper bound of the window h.

est_cp A vector containing the locations of the estimated change-points.

confidence Confidence level of SN test as a numeric value.

critical_value Critical value of the SN-based test statistic.

Users can apply the functions summary.SN to compute the parameter estimate of each segment separated by the detected change-points. An additional function plot.SN can be used to plot the time series with estimated change-points. Users can set the option plot_SN = TRUE or use the function plot.SN to plot the time series.

It deserves to note that some change-points could be missing due to the constraint on grid_size_scale or related grid_size that grid_size_scale has a minimum value of 0.05. Therefore, SNCP claims no change-points within the first ngrid_size_scale *or the last n*grid_size_scale time points. This is a limitation of the function SNSeg_Multi.

For more examples of SNSeg_Multi see the help vignette: vignette("SNSeg", package = "SNSeg")

Examples

```
# Please run this function before simulation
exchange_cor_matrix <- function(d, rho){
  tmp <- matrix(rho, d, d)
  diag(tmp) <- 1
  return(tmp)
}

# simulation of multivariate time series
library(mvtnorm)
set.seed(10)
d <- 5
```

```
n <- 600
nocp <- 5
cp_sets <- round(seq(0, nocp+1 ,1)/(nocp+1)*n)
mean_shift <- rep(c(0,2),100)[1:(length(cp_sets)-1)]/sqrt(d)
rho_sets <- 0.2
sigma_cross <- list(exchange_cor_matrix(d,0))
ts <- MAR_MTS_Covariance(n, 2, rho_sets, cp_sets = c(0,n), sigma_cross)
ts <- ts[1][[1]]

# Test for the change in multivariate means
# grid_size defined
result <- SNSeg_Multi(ts, paras_to_test = "mean", confidence = 0.99,
                      grid_size_scale = 0.05, grid_size = 45)
# Estimated change-point locations
result$est_cp

# For more examples, please run the following command:
# vignette("SNSeg", package = "SNSeg")
```

SNSeg_Uni *Self-normalization (SN) based change point estimates for univariate time series*

Description

The function SNSeg_Uni is a SN change point estimation procedure for a univariate time series based on the change in a single or multiple parameters . It also detect changes in correlation between two univariate time series.

Usage

```
SNSeg_Uni(
  ts,
  paras_to_test,
  confidence = 0.9,
  grid_size_scale = 0.05,
  grid_size = NULL,
  plot_SN = TRUE,
  est_cp_loc = TRUE
)
```

Arguments

ts	A univariate time series expressed as a numeric vector. when the argument paras_to_test is specified as "bivcor", the correlation between bivariate time series, the input ts must be an n by 2 matrix

`paras_to_test` The parameters that SN algorithm aim to examine, which are presented as a string, a number, or a combination of both. Available choices of paras_to_test include "mean", "variance", "acf", "bivcor" and a numeric value of quantile between 0 and 1. In the scenario where the input ts is a univariate time series, users are allowed to enter a combination of parameters for paras_to_test except "bivcor".

Users can also set up their own function as the input of "paras_to_test". If so, the user-fined function should use the univariate time series as the input and return a numeric value as the output. Please see the help vignette for more details by running `vignette("SNSeg", package = "SNSeg")`.

`confidence` Confidence level of SN tests as a numeric value. Available choices of confidence levels contain 0.9, 0.95, 0.99, 0.995 and 0.999. The default is set to 0.9.

`grid_size_scale`

A numeric value of the trimming parameter and only in use if grid_size = NULL. Users are allowed to choose any grid_size_scale between 0.05 and 0.5. A warning will be given if it is outside the range.

`grid_size` Local window size h to compute the critical value for SN test. Since grid_size = n*grid_size_scale, where n is the length of time series, this function will compute the grid_size_scale by dividing n from grid_size when it is not NULL.

`plot_SN` Boolean value to plot the time series or not. The default setting is FALSE.

`est_cp_loc` Boolean value to plot a red solid vertical line for estimated change-point locations if est_cp_loc = TRUE

Value

`SNSeg_Uni` returns an S3 object of class "SNSeg_Uni" including the time series, the type of parameter to be tested, the local window size to cover a change point, the estimated change-point locations, the confidence level and the critical value of the SN test. It also generates a time series segmentation plot when `plot_SN = TRUE`.

`ts` A numeric vector or two-dimensional matrix of the input time series.

`paras_to_test` A character, numeric value, a function or vector of the parameter(s) used for the SN test. If it is a function defined by the user, please refer to the section "test in a general functional" in the help vignette for more details on how to write the function correctly.

`grid_size` A numeric value of the window size.

`SN_sweep_result` A list of matrices where each matrix consists of four columns: (1) SN-based test statistic for each change-point location (2) Change-point location (3) Lower bound of the local window and (4) Upper bound of the local window.

`est_cp` A vector containing the locations of the estimated change-points.

`confidence` Confidence level of SN test as a numeric value.

`critical_value` Critical value of the SN-based test statistic.

Users can apply the functions `summary.SN` to compute the parameter estimate of each segment separated by the detected change-points. An additional function `plot.SN` can be used to plot the time series with estimated change-points. Users can set the option `plot_SN = TRUE` or use the function `plot.SN` to plot the time series.

It deserves to note that some change-points could be missing due to the constraint on grid_size_scale or related grid_size that grid_size_scale has a minimum value of 0.05. Therefore, SNCP claims no change-points within the first ngrid_size_scale *or the last n*grid_size_scale time points. This is a limitation of the function SNSeg_Uni.

For more examples of SNSeg_Uni see the help vignette: vignette("SNSeg", package = "SNSeg")

Examples

```
# code to simulate a univariate time series
set.seed(7)
ts <- MAR_Variance(2, "V1")
ts <- ts[,2]
# test the change in a single parameter (variance)
# grid_size defined
result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                    grid_size_scale = 0.05, grid_size = 67,
                    plot_SN = TRUE, est_cp_loc = TRUE)
# estimated change-point locations
result$est_cp
# For more examples of change in a single or multiple parameters, please run
# the command: vignette("SNSeg", package = "SNSeg")
```

---

summary.SNSeg_HD *Summary of SN-based change-point estimates for high-dimensional time series with dimension greater than 10*

Description

Summary method for objects of class SNSeg_HD

Usage

```
## S3 method for class 'SNSeg_HD'
summary(object, ...)
```

Arguments

object	a SNSeg_HD object
...	not in use

Details

Provide information about estimated change-point locations, the parameter tested by SN-based procedures, the confidence level, the grid_size, and the critical value of the SN-based test.

Examples

```
n <- 500
p <- 50
nocp <- 5
cp_sets <- round(seq(0,nocp+1,1)/(nocp+1)*n)
num_entry <- 5
kappa <- sqrt(4/5)
mean_shift <- rep(c(0,kappa),100)[1:(length(cp_sets)-1)]
set.seed(1)
ts <- matrix(rnorm(n*p,0,1),n,p)
no_seg <- length(cp_sets)-1
for(index in 1:no_seg){
  tau1 <- cp_sets[index]+1
  tau2 <- cp_sets[index+1]
  ts[tau1:tau2,1:num_entry] <- ts[tau1:tau2,1:num_entry] +
    mean_shift[index]
}

# grid_size defined
result <- SNSeg_HD(ts, confidence = 0.9, grid_size_scale  = 0.05,
                   grid_size = 40)
# summary method
summary(result)
```

---

summary.SNSeg_Multi *Summary of SN-based change-point estimates for multivariate time series with dimension no greater than 10*

Description

Summary method for objects of class `SNSeg_Multi`

Usage

```
## S3 method for class 'SNSeg_Multi'
summary(object, ...)
```

Arguments

`object`	a `SNSeg_Multi` object
`...`	not in use

Details

Provide information about estimated change-point locations, the parameter tested by SN-based procedures, the confidence level, the `grid_size`, and the critical value of the SN-based test.

Examples

```
# Please run this function before simulation
exchange_cor_matrix <- function(d, rho){
  tmp <- matrix(rho, d, d)
  diag(tmp) <- 1
  return(tmp)
}

# simulation of multivariate time series
library(mvtnorm)
set.seed(10)
d <- 5
n <- 600
nocp <- 5
cp_sets <- round(seq(0, nocp+1 ,1)/(nocp+1)*n)
mean_shift <- rep(c(0,2),100)[1:(length(cp_sets)-1)]/sqrt(d)
rho_sets <- 0.2
sigma_cross <- list(exchange_cor_matrix(d,0))
ts <- MAR_MTS_Covariance(n, 2, rho_sets, cp_sets = c(0,n), sigma_cross)
ts <- ts[1][[1]]

# Test for the change in multivariate means
# grid_size defined
result <- SNSeg_Multi(ts, paras_to_test = "mean", confidence = 0.99,
                      grid_size_scale = 0.05, grid_size = 45)
# summary method
summary(result)
```

`summary.SNSeg_Uni` *Summary of SN-based change-point estimates for univariate or bivariate time series (testing the change in correlation between bivariate time series)*

Description

Summary method for objects of class `SNSeg_Uni`

Usage

```
## S3 method for class 'SNSeg_Uni'
summary(object, ...)
```

Arguments

`object`	a `SNSeg_Uni` object
`...`	not in use

Details

Provide information about estimated change-point locations, the parameter tested by SN-based procedures, the confidence level, the `grid_size`, and the critical value of the SN-based test.

Examples

```
set.seed(7)
ts <- MAR_Variance(2, "V1")
ts <- ts[,2]
# test the change in a single parameter (variance)
# grid_size defined
result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                    grid_size_scale = 0.05, grid_size = 67,
                    plot_SN = FALSE, est_cp_loc = TRUE)
summary(result)
```

Index