SNSeg: An R Package for Time Series Segmentation via Self-Normalization

Shubo Sun, Zifeng Zhao, Feiyu Jiang, Xiaofeng Shao

Abstract Time series segmentation aims to identify potential change-points in a sequence of temporally dependent data, so that the original sequence can be partitioned into several homogeneous subsequences. This is useful for modelling and predicting of non-stationary time series and is widely applied in natural and social sciences. Existing approaches primarily focus on only one type of parameter changes such as mean and variance, and they typically depend on laborious tuning or smoothing parameters, which can be challenging to choose in practice. The self-normalization (SN) based change-point (CP) estimation framework SNCP by Zhao et al. (2022), however, offers users more flexibility and convenience as it allows for change-point estimation of different types of parameters (e.g. mean, variance, quantile and autocovariance) in a unified fashion, and requires effortless tuning. In this paper, the R package **SNSeg** is introduced to implement SNCP for estimation of change-points and segmentation of univariate and multivariate time series. An extension of SNCP, named SNHD, is also designed and implemented for change-point estimation of mean vector in high-dimensional time series. The estimated change-points as well as segmented time series are available with graphical tools. Detailed examples are given in simulations of multivariate autoregressive processes with change-points.

1 Introduction

Time series segmentation, also known as change-point estimation in time series, has become increasingly popular in various fields such as statistics, bioinformatics, climate science, economics, finance, signal processing, pandemic analysis, among many others. As a result, numerous methods have been proposed to address different types of change-point estimation problems in various settings. This in turn leads to the development of many software tools for their implementation.

Here, we list some commonly used packages for change-point analysis in the **R** programming language. The package **bcp** (Erdman and Emerson, 2007) implements Bayesian methods proposed by Barry and Hartigan (1993); the package **strucchange** (Zeileis et al., 2002) detects structural changes in linear regression models using algorithms in Zeileis et al. (2003) and Zeileis (2006); the package **changepoint** (Killick and Eckley, 2014) applies the pruned exact linear time (PELT) algorithm described in Killick et al.; the package **cpm** (Ross, 2015) implements methods for sequential change-point detection which are first introduced by Hawkins et al. (2003); Hawkins and Zamba (2005) with multiple extensions (see Ross (2014) for a summary of the extensions); the package **ecp** (James and Matteson, 2014) implements the nonparametric change-point methods for multivariate time series from Matteson and James (2014); the package **mosum** (Meier et al., 2021) performs moving sum (MOSUM) procedure from Eichinger and Kirch (2018) and the package **cpss** (Wang and Zou, 2023) focuses on a variety of parametric generalized linear models with the sample-splitted strategy from Zou et al. (2020).

However, the aforementioned methods, as well as their implementation packages, suffer from two main drawbacks when applied to change-point estimation in multivariate time series. First, they are mostly designed for detecting only one specific type of change (e.g. mean or variance) and cannot be easily generalized when changes in different aspects of data are of interest. This means that for the same dataset, different methods are usually required for estimating different types of changes, which may incur inconvenience of implementation for practitioners. Second, they typically require certain tuning or smoothing parameters, such as the bandwidth parameter for consistent estimation of the long-run variance, which is almost inevitable in the presence temporal dependence. How to choose these tuning parameters is important yet highly challenging in practice, and different tuning parameter choices may yield quite different outputs.

Recently, Zhao et al. (2022) have developed a new framework called self-normalization based change-point estimation (SNCP) to deal with the above limitations. The most appealing feature of SNCP is its versatility as it allows for change-point estimation in a broad class of parameters (such as mean, variance, correlation, quantile and their combinations) in a unified fashion. The basic idea of SNCP is to augment the conventional cumulative sum (CUSUM) statistics with the technique called self-normalization (SN). SN was originally introduced by Shao (2010) for confidence interval construction of a general parameter in the stationarity time series setting, and extended to change-point testing by Shao and Zhang (2010). It can bypass the issue of bandwidth selection in the consistent long-run variance estimation. See Shao (2015) for a review. This makes SNCP fully nonparametric, robust to temporal dependence and applicable universally for various parameters of interest for a multivariate time series. Furthermore, based on a series of carefully designed nested local-windows, SNCP can isolate each true change-point adaptively and thus achieves the goal of multiple change-point estimation with respectable detection power and estimation accuracy.

In this paper, we introduce the **R** package **SNSeg** to apply the SNCP framework for univariate and multivariate time series segmentation. This can be achieved by the functions `SNSeg_Uni` and `SNSeg_Multi`. Another contribution of this paper is to extend the SNCP framework into estimation of change-points in mean vector of high-dimensional time series. Since SNCP is only applicable to low/fixed-dimensional time series, a new procedure based on U-statistics (Wang et al., 2022), termed as SNHD, is proposed by modifying SNCP. The implementation of SNHD is available through the `SNSeg_HD` function. Graphical options are also allowed for plotting the estimated change-points and associated test statistics.

In the sections that follow, we first provide the background of SN based statistics and the SNCP/SNHD procedures for change-point estimation in Section 2. In Section 3, we demonstrate the core functions of **SNSeg** by examples of change-point estimation problem in various model parameters. Section 4 concludes.

2 SNCP Framework

This section gives necessary statistical backgrounds of SNCP in change-point estimation problems. We first demonstrate how SN based CUSUM test statistic works for estimating a single change-point, and then introduce the nested local-window based SNCP algorithm for multiple change-point estimation. The extension of SNCP to estimating change-points in high-dimensional mean problem is also provided and we term the related algorithm as SNHD. The issue of how to choose tuning parameters are also discussed.

2.1 Single Change-Point Estimation

Let $\{Y_t\}_{t=1}^n$ be a sequence of multivariate time series of dimension p, which is assumed to be fixed for now. We aim to detect whether there is a change-point in the quantities $\{\theta_t\}_{t=1}^n$ defined by $\theta_t = \theta(F_t) \in \mathbb{R}^d$, where F_t denotes the distribution function of Y_t, and $\theta(\cdot)$ is a general functional such as mean, variance, auto-covariance, quantiles, etc. More specifically, if there is no change-point, then

$$\theta_1 = \cdots = \theta_n. \tag{1}$$

Otherwise, we assume there is an unknown change-point $k^* \in \{1, \cdots, n-1\}$ defined by

$$\theta_1 = \cdots = \theta_{k^*} \neq \theta_{k^*+1} = \cdots = \theta_n, \tag{2}$$

and our interest is to recover the location k^*.

The above setting allows for at most one change-point in $\{\theta_t\}_{t=1}^n$. A commonly used statistic for testing the existence of change-points is based on the CUSUM process, defined by

$$D_n(k) = \frac{k(n-k)}{n^{3/2}} \left(\hat{\theta}_{1,k} - \hat{\theta}_{k+1,n}\right), \;\; k \in \{1, 2, \cdots, n-1\}, \tag{3}$$

where for any $1 \leq a < b \leq n$, $\hat{\theta}_{a,b} = \theta(\hat{F}_{a,b})$ estimates the model parameter with $\hat{F}_{a,b}$ being the empirical distribution of $\{Y_t\}_{t=a}^b$. For example, when $\theta(\cdot)$ is the mean functional, it can be shown that

$$D_n(k) = \frac{1}{\sqrt{n}} \sum_{t=1}^{k} (Y_t - \bar{Y}), \quad \bar{Y} = n^{-1} \sum_{t=1}^{n} Y_t.$$

The CUSUM process in (3) sequentially compares the estimates before and after the time point k, and its norm is expected to attain the maxima when $k = k^*$. Intuitively, if (1) holds, then $D_n(k)$ should fluctuate around zero; otherwise if (2) holds, then $\hat{\theta}_{1,k}$ and $\hat{\theta}_{k+1,n}$ are consistent estimators for θ_1 and θ_n, respectively at $k = k^*$, and the resulting contrast $\|D_n(k^*)\|$ would be most informative about the change signal $\Delta_n = \theta_n - \theta_1$. Therefore, under (2), it is natural to estimate the change-point location via

$$\tilde{k} = \arg \max_{k=1,\cdots,n-1} \|D_n(k)\|^2. \tag{4}$$

However, analyzing the asymptotic distribution of CUSUM process $\{D_n(\lfloor nr \rfloor)\}_{r\in[0,1]}$ for time series data is difficult, as it typically depends on a nuisance parameter called long-run variance (Newey and West, 1987; Andrews, 1991). As we have emphasized before, the estimation of long-run variance is quite challenging, let alone the scenario when a change-point is present.

To bypass this issue, Zhao et al. (2022) propose to estimate the change-point via the self-normalized version of (4), i.e.

$$\hat{k} = \arg\max_{k=1,\cdots,n-1} T_n(k), \quad T_n(k) = D_n(k)' V_n^{-1}(k) D_n(k), \tag{5}$$

where

$$V_n(k) = \sum_{i=1}^{k} \frac{i^2(k-i)^2}{n^2k^2} (\hat{\theta}_{1,i} - \hat{\theta}_{i+1,k})^{\otimes 2} + \sum_{i=k+1}^{n} \frac{(n-i+1)^2(i-k-1)^2}{n^2(n-k)^2} (\hat{\theta}_{i,n} - \hat{\theta}_{k+1,i-1})^{\otimes 2}, \tag{6}$$

is defined as the self-normalizer with $a^{\otimes 2} = aa'$ for a vector a. The self-normalizer $V_n(k)$ is proportional to long run variance, which gets canceled out in the limiting null distribution of $T_n(k)$. Thus the testing/estimation of a single change-point is completely free of tuning parameters.

In practice, we may not know whether the series $\{\theta_t\}_{t=1}^n$ contains a change-point or not, so a testing step is called for prior to the estimation step. Formally speaking, given a pre-specified threshold K_n, we declare a change-point when

$$SN_n := \max_{k=1,\cdots,n-1} T_n(k) > K_n,$$

and then estimate the single change-point via (5). Otherwise, if SN_n is below the threshold K_n, we declare no change-points.

2.2 Multiple Change-Point Estimation

At this point we have introduced how SNCP works in the single change-point setting. Next we further extend it to multiple change-point estimation. The main difficulty of the multiple change-point estimation compared with the single change-point setting lies in how to isolate one change-point from another. In SNCP, this is achieved by a nested local-window approach.

We first introduce some notations. Assume there are $m_0 \geq 0$ unknown number of change-points with $k_0 = 0 < k_1 < \cdots < k_{m_0} < n = k_{m_0+1}$ that partition Y_t into $m_0 + 1$ stationary segments with constant quantity of interest $\theta^{(i)}$ in the ith segment, for $i = 1, \cdots, m_0 + 1$. In other words,

$$\theta_t = \theta^{(i)}, \;\; k_{i-1} + 1 \leq t \leq k_i, \text{ for } i = 1, \cdots, m_0 + 1.$$

Similar to the single change-point estimation framework, for $1 \leq t_1 < k < t_2 \leq n$, we define

$$T_n(t_1, k, t_2) = D_n(t_1, k, t_2)' V_n^{-1}(t_1, k, t_2) D_n(t_1, k, t_2),$$

where $D_n(t_1, k, t_2) = \frac{(k-t_1+1)(t_2-k)}{(t_2-t_1+1)^{3/2}} (\hat{\theta}_{t_1,k} - \hat{\theta}_{k+1,t_2})$, $V_n(t_1, k, t_2) = L_n(t_1, k, t_2) + R_n(t_1, k, t_2)$ and

$$L_n(t_1, k, t_2) = \sum_{i=t_1}^{k} \frac{(i-t_1+1)^2(k-i)^2}{(t_2-t_1+1)^2(k-t_1+1)^2} (\hat{\theta}_{t_1,i} - \hat{\theta}_{i+1,k})^{\otimes 2},$$

$$R_n(t_1, k, t_2) = \sum_{i=k+1}^{t_2} \frac{(t_2-i+1)^2(i-1-k)^2}{(t_2-t_1+1)^2(t_2-k)^2} (\hat{\theta}_{i,t_2} - \hat{\theta}_{k+1,i-1})^{\otimes 2}.$$

Here $T_n(t_1, k, t_2)$ plays the same role as $T_n(k)$ in (5), except for the fact that it is defined on the subsample $\{Y_t\}_{t=t_1}^{t_2}$. In other words, $T_n(t_1, k, t_2) = T_n(k)$ if $t_1 = 1$ and $t_2 = n$.

We now combine the SN framework with a nested local-window segmentation algorithm in Zhao et al. (2022) for multiple change-point estimation. For each k, instead of using the global statistic $T_n(1, k, n)$ which is built with all observations, we compute a maximal SNCP test statistic based on a collection of nested windows covering k. Specifically, we fix a small trimming parameter $\epsilon \in (0, 1/2)$ and define the window size $h = \lfloor n\epsilon \rfloor$. For each $k = h, \cdots, n-h$, we define the nested local-window set $H_{1:n}(k)$ as

$$H_{1:n}(k) = \{(t_1, t_2) | t_1 = k - j_1 h + 1, j_1 = 1, \cdots, \lfloor k/h \rfloor; t_2 = k + j_2 h, j_2 = 1, \cdots, \lfloor (n-k)/h \rfloor.\} \tag{7}$$

Note that for $k < h$ and $k > n - h$, we have $H_{1:n}(k) = \emptyset$. In Figure 1, we plot a graphical illustration of these nested local-windows, where local windows are constructed by combining every pair of red left bracket "[" and blue right bracket "]".

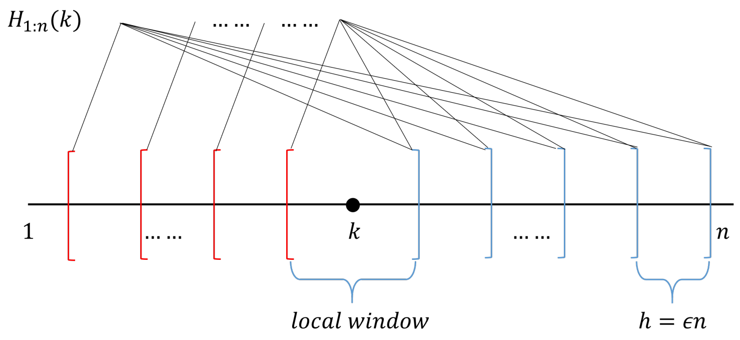

Figure 1: Graphical illustration of nested local-windows. Each pair of the red bracket "[" and blue bracket "]" represents the local window for some (t_1, t_2) in $H_{1:n}(k)$.

For each $k = 1, \cdots, n-1$, based on its nested local-window set $H_{1:n}(k)$, we define a maximal SN test statistic such that

$$T_{1,n}(k) = \max_{(t_1,t_2) \in H_{1:n}(k)} T_n(t_1, k, t_2), \tag{8}$$

where we set $\max_{(t_1,t_2)\in\emptyset} T_n(t_1, k, t_2) = 0$.

Intuitively, with a sufficiently small trimming ϵ, the nested local-window framework ensures that for a change-point location, say k^*, there exists some local window set denoted by (t_1^*, t_2^*) containing k^* as the only change-point. In other words, k^* is isolated by the interval (t_1^*, t_2^*) so that the procedure in the single change-point scenario as Section 2.1 can be applied. This suggests that for at least one pair of (t_1, t_2) in $H_{1:n}(k)$, $T_n(t_1, k^*, t_2)$ is large. The detection power is further enhanced by taking the maxima of these test statistics.

Based on the maximal test statistic $T_{1,n}(k)$ and a prespecified threshold K_n, SNCP proceeds as follows. Starting with the full sample $\{Y_t\}_{t=1}^n$, we calculate $T_{1,n}(k), k = 1, \cdots, n$. Given that $\max_{k=1,\ldots,n} T_n(k) \le K_n$, SNCP declares no change-point. Otherwise, SNCP sets $\widehat{k} = \arg\max_{k=1,\ldots,n} T_{1,n}(k)$ and we recursively perform SNCP on the subsample $\{Y_t\}_{t=1}^{\widehat{k}}$ and $\{Y_t\}_{t=\widehat{k}+1}^n$ until no change-point is declared. Denote $W_{s,e} = \{(t_1, t_2) | s \le t_1 < t_2 \le e\}$ and $H_{s:e}(k) = H_{1:n}(k) \bigcap W_{s,e}$, which is the nested window set of k on the subsample $\{Y_t\}_{t=s}^e$. Define the subsample maximal SN test statistic as $T_{s,e}(k) = \max\limits_{(t_1,t_2)\in H_{s:e}(k)} T_n(t_1, k, t_2)$. Algorithm 1 states the formal description of SNCP in multiple change-points estimation.

Algorithm 1: SNCP procedures for multiple change-point estimation

```
Input: Time Series $\{Y_t\}_{t=1}^n$, threshold $K_n$, window size $h = \lfloor n\epsilon \rfloor$
Output: Estimated change-point set $\hat{\mathbf{k}} = (\hat{k}_1, \cdots, \hat{k}_{\hat{m}})$
Initialization: SNCP$(1, n, K_n, h)$, $\hat{k} = \emptyset$
Procedure: SNCP$(s, e, K_n, h)$ if $e - s + 1 < 2h$ then
    Stop
else
    $\hat{k} = \arg\max_{k=s,\cdots,e} T_{s,e}(k)$;
    if $T_{s,e}(\hat{k}^*) \le K_n$ then
        Stop
    else
        $\hat{\mathbf{k}} = \hat{\mathbf{k}} \cup \hat{k}^*$;
        SNCP$(s, \hat{k}^*, K_n, h)$; SNCP$(\hat{k}^* + 1, e, K_n, h)$;
    end
end
```

2.3 Multiple Change-Point Estimation for High-Dimensional Mean

In this section, we modify the SNCP framework in Section 2.2 to design a new algorithm called SNHD for multiple change-point estimation for high-dimensional mean vector. Different from the SNCP subsample test statistic $T_n(t_1, k, t_2)$ used for a low-dimensional time series, SNHD is designed based on the high-dimensional U-statistic proposed by Wang et al. (2022). Given a p-dimensional time series $\{Y_t\}_{t=1}^n$, we define the subsample contrast statistic as

$$D_n^U(t_1,k,t_2) = \sum_{\substack{t_1 \le j_1,j_3 \le k \\ j_1 \ne j_3}} \sum_{\substack{k+1 \le j_2,j_4 \le j_2 \\ j_2 \ne j_4}} (Y_{j_1} - Y_{j_2})^\top (Y_{j_3} - Y_{j_4}). \tag{9}$$

Note that $D_n^U(t_1,k,t_2)$ is a two-sample U-statistic estimating the squared L_2 norm of the difference between the means of $\{Y_t\}_{t=t_1}^k$ and $\{Y_t\}_{t=k+1}^{t_2}$ (up to some normalizing constant). In addition, the form above is only applicable to high-dimensional data with temporal independence, and in the presence of temporal dependence, a trimming parameter needs to be introduced to alleviate the bias due to serial dependence; see Wang et al. (2022). Define the self-normalizer as

$$V_n^U(t_1,k,t_2) = \frac{1}{n}\Big[\sum_{t=t_1+1}^{k-2} D_n^U(t_1,t,k)^2 + \sum_{k+2}^{t_2-2} D_n^U(k+1,t,t_2)^2\Big]. \tag{10}$$

The subsample SNHD test statistic at time point k, in the same spirit as (8), is defined by

$$T_{1,n}^U(k) = \max_{(t_1,t_2)\in H_{1:n}(k)} T_n^U(t_1,k,t_2), \quad T_n^U(t_1,k,t_2) = D_n^U(t_1,k,t_2)^2 / V_n^U(t_1,k,t_2), \tag{11}$$

where $H_{1:n}(k)$ is the nested local-window set defined in (7). With a pre-specified threshold K_n^U, a change-point is detected at $\hat{k} = \arg\max_{k=1,\cdots,n} T_{1,n}^U(k)$ if $\max_{k=1,\cdots,n} T_{1,n}^U(k)$ is above K_n^U. For multiple change-point estimation, SNHD proceeds similarly as SNCP in Algorithm 1.

2.4 Choice of Trimming Parameter ϵ and Threshold K_n

For practical implementation, there are still two tuning parameters that one has to choose, namely the trimming parameter ϵ and change-point detection threshold K_n. The choice of ϵ reflects one's belief of minimum (relative) spacing between two consecutive change-points. This is usually set to be a small constant such as 0.05, 0.10, 0.15.

To pin down the exact value of K_n (or K_n^U in SNHD), it would be helpful to derive the limiting distributions of the testing statistics under the no change-point scenario.

In particular, for SNCP, recall (8), with certain technical conditions, it can be shown that as $n \to \infty$,

$$\max_{k=1,\cdots,n} T_{1,n}(k) \xrightarrow{\mathcal{D}} G_{\epsilon,d} = \sup_{u\in(\epsilon,1-\epsilon)} \max_{(u_1,u_2)\in H_\epsilon(u)} D(u_1,u,u_2)^\top V(u_1,u,u_2)^{-1} D(u_1,u,u_2), \tag{12}$$

where for $\mathcal{B}_d(\cdot)$ being a d-dimensional Brownian motion with independent entries, and $H_\epsilon(u) = \{(u_1,u_2) | u_1 = u - j\epsilon, j = 1,\cdots,\lfloor u/\epsilon \rfloor;\ u_2 = u + j\epsilon, j = 1,\cdots,\lfloor (1-u)/\epsilon \rfloor\}$, $D(u_1,u,u_2) = \frac{1}{\sqrt{u_2-u_1}}\boldsymbol{\Delta}(u_1,u,u_2)$, $V(u_1,u,u_2) = \frac{1}{(u_2-u_1)^2}\Big(\int_{u_1}^{u}\boldsymbol{\Delta}(u_1,s,u)\boldsymbol{\Delta}(u_1,s,u)^\top ds + \int_u^{u_2}\boldsymbol{\Delta}(u,s,u_2)\boldsymbol{\Delta}(u_1,s,u)^\top ds\Big)$ with $\boldsymbol{\Delta}(u_1,u,u_2) = \mathcal{B}_d(u) - \mathcal{B}_d(u_1) - \frac{u-u_1}{u_2-u_1}[\mathcal{B}_d(u_2) - \mathcal{B}_d(u_1)]$.

Similarly, for SNHD, it can be shown that as $\min(n,p) \to \infty$,

$$\max_{k=1,2,\cdots,n} T_{1,n}^U(k) = \max_{k=1,2,\cdots,n} \max_{(t_1,t_2)\in H_{1:n}(k)} T_n^U(t_1,k,t_2) \xrightarrow{\mathcal{D}} G_\epsilon^U, \tag{13}$$

$$\text{with } G_\epsilon^U = \sup_{u\in(\epsilon,1-\epsilon)} \sup_{(u_1,u_2)\in H_\epsilon(u)} \frac{D^U(u_1,u,u_2)^2}{\int_{u_1}^{u} D^U(u_1,s,u)^2 ds + \int_u^{u_2} D^U(u,s,u_2)^2 ds},$$

where $D^U(u_1,u,u_2) = (u_2-u_1)(u_2-u)Q(u_1,u) + (u-u_1)(u_2-u_1)Q(u,u_2) - (u-u_1)(u_2-u)Q(u_1,u_2)$, and $Q(\cdot,\cdot)$ is a centered Gaussian process on $[0,1]^2$ with its covariance structure as $\mathrm{Cov}(Q(a_1,b_1),Q(a_2,b_2)) = (b_1 \wedge b_2 - a_1 \vee a_2)^2 \mathbb{I}\{b_1 \wedge b_2 > a_1 \vee a_2\}$.

The limiting distributions in (12) and (13) characterize the asymptotic behaviors of SNCP and SNHD under no change-point environment and thus provide practical guidance on how to threshold K_n and K_n^U, respectively. Furthermore, we note that given the trimming parameter ϵ, though complicated, both $G_{\epsilon,d}$ and G_ϵ^U are pivotals, suggesting that their critical values could be obtained via simulations. Since K_n and K_n^U are used to balance one's tolerance of type-I and type-II errors, we recommend choosing K_n and K_n^U as the $\alpha \times 100\%$ quantiles of $G_{\epsilon,d}$ and G_ϵ^U with α typically set as 0.9, 0.95, 0.99 or 0.999.

In the **SNSeg** package, we offer users a wide class of ϵ and K_n (or K_n^U) to choose from. Details are given in the following section.

3 The SNSeg Package

In this section, we introduce the functions within the **SNSeg** package for multiple change-point estimation. In particular, SNSeg_Uni in Section 3.1 implements SNCP procedures to estimate change-points for a univariate time series with changes in a single or multiple parameters, such as mean, variance, auto-correlations, quantiles or even their combinations. It also contains functions for testing the change in correlations between bivariate time series. The function SNSeg_Multi in Section 3.2 utilizes the SNCP algorithm to estimate the changes in multivariate means or covariance functions for multivariate time series, where the dimension of time series is at most ten. In Section 3.3, the function SNSeg_HD estimates change-points in mean of high-dimensional time series using SNHD procedures. In addition to these major functions for change-point estimation, we also offer the function max_SNsweep in Section 3.4 to obtain the SN test statistics, and create a segmentation plot based on the output of above functions.

3.1 SNCP Procedures for a Univariate Time Series: SNSeg_Uni

For a univariate time series, the estimation of change-points in a single or multiple functionals can be implemented through the function SNSeg_Uni. This function is also capable of identifying change-points associated with the change in correlation between bivariate time series. The **R** code is given as:

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

It takes the following input arguments.

- ts: Input time series $\{Y_t\}_{t=1}^n$, i.e., a univariate time series expressed as a numeric vector with length n. However, when the argument paras_to_test is specified as bivcor, the correlation between bivariate time series, the input ts must be an $n \times 2$ 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 options of paras_to_test include:
  - "mean": The function performs SN tests based on the change in mean of the time series.
  - "variance": The function performs SN tests based on the change in variance.
  - "acf": The function performs SN tests based on the change in autocorrelation.
  - "bivcor": The function performs SN tests based on the change in bivariate correlation.
  - A numeric value within the range (0,1): The function performs SN tests based on the change in a specific quantile level.
  - In the scenario where the input ts is a univariate time series, the argument paras_to_test allows a combination of parameters including mean, variance, acf, and one or more quantile levels. Accordingly, SNSeg_Uni is capable of estimating change-points in either a single parameter or a combination of multiple parameters.
- 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. It automatically obtains the threshold (K_n, the critical value) corresponding to the input confidence level. The default value of confidence is set at 0.9.
- grid_size_scale: A numeric value of the trimming parameter ϵ and only in use if grid_size = NULL. Available choices include 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45 and 0.5. The default value of grid_size_scale is 0.05.
  - In the function, any input less than 0.05 will be set to exactly 0.05 and similarly, any input greater than 0.5 will be set to 0.5. In such cases, a warning that "Detected the grid_size_scale is greater than 0.5" or "less than 0.05" will be generated.
- grid_size: Local window size h. It should be noted that $h = \lfloor n \times \epsilon \rfloor$, i.e., grid_size= $\lfloor n \times$ grid_size_scale$\rfloor$. By default, the value of grid_size is set to NULL, and the function computes the critical value using the argument grid_size_scale. However, users have the option to set the grid_size manually, in which case the function computes the corresponding grid_size_scale by dividing n from grid_size, and then determines the critical value using this computed value.
- plot_SN: Boolean value to plot the time series or not. The default setting is TRUE.

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

The function `SNSeg_Uni` provides users with flexibility by allowing them to select parameter types that they want to target at. Additionally, users can specify the window size or choose an appropriate value for ϵ to achieve the desired theoretical critical value. It also permits casual input of values for the arguments `grid_size_scale` and `grid_size`. If the calculated trimming parameter ϵ by any of the two arguments falls within the range [0.05, 0.5], but is not provided in the critical value table, the function performs a linear interpolation by identifying two nearest `grid_size_scale` that are below and above the calculated ϵ and then computes the weighted average of the corresponding critical values. The resulting interpolated value is used as the final critical value for the SN test.

When called, `SNSeg_Uni` returns a list of objects containing the following entries:

- `paras_to_test`: The parameter(s) used for the SN test.
- `grid_size`: A numeric value of the window size h.
- `SN_sweep_result`: A list of n matrices where each matrix consists of four columns: (1) SN test statistic T_n for each change-point location k (2) Location k (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 the SN test as a numeric value.
- `critical_value`: Critical value of the SN test statistic given ϵ and the confidence level.

It is worth noting that the output of the function `SNSeg_Uni` can serve as an input of the function `max_SNsweep` (to be described in Section 3.4) to generate a segmentation plot for SN test statistics, and the same also holds for functions `SNSeg_Multi` and `SNSeg_HD`.

To illustrate this function, we present examples demonstrating change-point estimation based on changes in both single and multiple parameters.

Example: variance change in univariate time series

We start with the example of the change-point model (V1) in Section S.2.5 in supplement of Zhao et al. (2022), where two variance changes occur at $k = 400$ and 750, respectively. Specifically,

$$(\mathrm{V1}): \quad Y_t = \begin{cases} 0.5Y_{t-1} + \epsilon_t, & t \in [1, 400], \\ 0.5Y_{t-1} + 2\epsilon_t, & t \in [401, 750], \\ 0.5Y_{t-1} + \epsilon_t, & t \in [751, 1024], \end{cases}$$

where ϵ_t is a sequence of i.i.d. $N(0, 1)$ random variables.

We set `grid_size_scale` at 0.067, and `confidence` at 90%. Subsequently, we visualize the input time series by setting `plot_SN` as TRUE, and generate an SN test statistics segmentation plot using the `max_SNsweep` function (to be introduced in Section 3.4).

```
# Generate model (V1)
set.seed(7)
ts <- MAR_Variance(2, "V1")
ts <- ts[,2]
par(mfcol = c(2, 1), mar = c(4, 2.5, 2.5, 0.5))

# SNCP in the change of univariate variance
result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                grid_size_scale = 0.067, grid_size = NULL, plot_SN = TRUE,
                est_cp_loc = TRUE)

# Segmentation plot for SN-based test statistics
max_SNsweep(result, plot_SN = TRUE, est_cp_loc = TRUE, critical_loc = TRUE)
```

The estimated locations of the change-points, the window size, and the critical value can be accessed using the following commands:

```
 result$est_cp
[1] 401 738
 result$grid_size
[1] 68
 result$critical_value
[1] 129.1731
```

The estimated change-point locations are $\hat{k} = 401$ and 738 with a local window size of $h = 68$ and critical value of $K_n = 129.1731$ when setting the parameters $\epsilon = 0.067$ and a confidence level of 90%. It is clear that the estimated change-points align with the original setting, which demonstrates the accuracy of the SNCP procedure.

Figure 2 displays segmentation plots for the input time series and the SN test statistics regarding the changes in univariate variance. It reveals that the SN statistics associated with the detected change-points surpass the critical value, and can be deemed as local maximas.

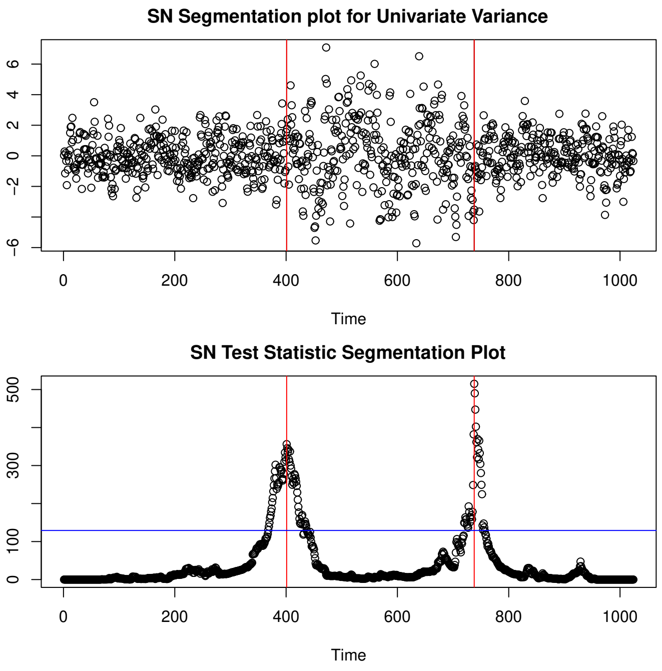

Figure 2: An example of simulated time series of model (V1). The upper panel illustrates the detection of change-points for the input time series, while the lower panel displays the segmentation of SN test statistics with the estimated change-points. The detected change-point locations are indicated by a red vertical line and the critical value is represented by a blue horizontal line.

We further manually specify the value of `grid_size` to calculate the critical value and estimate change-points. The function `SNSeg_Uni` is applied to the same time series, with the only difference being that the window size h is set to 65 instead of `NULL`. The estimated change-points and the critical value can be obtained using the following commands:

```
 result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                     grid_size_scale = 0.067, grid_size = 65, plot_SN = FALSE,
                     est_cp_loc = FALSE)

 result$est_cp
[1] 401 738
 result$critical_value
[1] 131.4857
```

The estimated change-points remain at $\hat{k} = 401$ and 738, but the critical value differs from the previous example. The discrepancy in critical values arises due to the variation in the window size h (or equivalently, the corresponding ϵ value), which reflects the influence of the chosen window size or the trimming parameter ϵ on the estimation results.

In addition to identifying changes in a single parameter, `SNSeg_Uni` also allows for estimating change-points by simultaneously combining information across multiple parameters. This is done using the `paras_to_test` argument. For example, users can specify `paras_to_test = c("mean", "acf", 0.6, 0.9)` to test the simultaneous changes in mean, autocorrelation, 60% and 90% quantile of the input time series. We note that the multiple-parameters setting is only applicable to univariate time series, and hence the option for bivariate correlation (`bivcor`) is not available.

Example: multiple-parameter change in univariate time series

We consider the simulated univariate time series of (MP1) setting in Section 4.4 of Zhao et al.

(2022), where the true change-points are located at $k = 333$ and 667. In particular,

$$(\mathrm{MP1}) : Y_t = \begin{cases} X_t, & t \in [1, 333] \\ F^{-1}\left(\Phi\left(X_t\right)\right), & t \in [334, 667] \\ X_t, & t \in [668, 1000], \end{cases}$$

where $\{X_t\}_{t=1}^n$ follows an AR(1) process with $X_t = 0.2X_{t-1} + \sqrt{1-\rho^2}\epsilon_t$, ϵ_t is a sequence of i.i.d. $N(0,1)$ random variables, $\Phi(\cdot)$ denotes the CDF of $N(0,1)$, and $F(\cdot)$ denotes a mixture of a truncated normal and a generalized Pareto distribution such that $F^{-1}(q) = \Phi^{-1}(q)$ for $q \leq 0.5$ and $F^{-1}(q) \neq \Phi^{-1}(q)$ for $q > 0.5$.

We aim to detect change-points based on the information from mean, variance, and 90% and 95% quantiles, and we set the `grid_size_scale` at 0.05 and `confidence` at 0.9.

```
# Generate model (MP1)
set.seed(7)
n <- 1000
cp_sets <- c(0, 333, 667, 1000)
no_seg <- length(cp_sets) - 1
rho <- 0
ts <- MAR(n, 2, rho)*sqrt(1-rho^2)
no_seg <- length(cp_sets)-1
sd_shift <- c(1,1.6,1)
for(index in 1:no_seg){ # Mean shift
  tau1 <- cp_sets[index]+1
  tau2 <- cp_sets[index+1]
  ts[tau1:tau2,] <- ts[tau1:tau2,]*sd_shift[index]
}
ts <- ts[,2]

# SNCP in the change of mean, variance, 90% quantile and 95% quantile
result <- SNSeg_Uni(ts, paras_to_test = c("mean","variance",0.9,0.95), confidence = 0.9,
                    grid_size_scale = 0.05, grid_size = NULL, plot_SN = FALSE,
                    est_cp_loc = FALSE)

# Output
result$est_cp
[1] 331 664
result$grid_size
[1] 50
result$critical_value
[1] 344.3622
```

As observed in the example, the estimated change-points take place at $\hat{k} = 331$ and 664, which are close to the locations of true change-points.

3.2 SN Procedures for Multivariate Time Series

The SN change-point detection for multivariate time series can be obtained via the function `SNSeg_Multi`. Multivariate means or covariance matrix of the input multivariate time series are parameters for this function.

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

The input arguments of `confidence`, `grid_size`, `grid_size_scale` are similar to those of the function `SNSeg_Uni`, and the difference lies in `ts` and `paras_to_test`.

- `ts`: Input time series $\{Y_t = (Y_{t1}, \cdots, Y_{tp})\}_{t=1}^n$ as a matrix, i.e., a multivariate time series represented as a matrix with n rows and 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. Only one parameter can be selected at a time. Available options of `paras_to_test` include:

- "mean": The function performs SN tests based on the change in multivariate means of the time series.
- "covariance": The function performs SN tests based on the change in the covariance matrix of the time series.

When necessary, the function SNSeg_Multi applies the same linear interpolation rule as SNSeg_Uni to determine the critical value for SN tests. We note that SNSeg_Multi does not include the option to plot the input multivariate time series ts.

When called, SNSeg_Multi returns a list comprising the grid_size, SN_sweep_result, est_cp, confidence and critical_value, which have been described in the context of the function SNSeg_Uni.

Example: mean change in multivariate time series

We consider model (M2) in Section 4.2 of Zhao et al. (2022), which is generated by

$$(\text{M2}) : Y_t = \begin{cases} -3/\sqrt{5} + X_t, & t \in [1,75], [526,575], \\ 0 + X_t, & t \in [76,375], [426,525], [576,1000], \\ 3/\sqrt{5} + X_t, & t \in [376,425]. \end{cases}$$

where X_t is a VAR(1) process with $X_t = 0.5X_{t-1} + \epsilon_t$, and ϵ_t is a sequence of i.i.d. $N(0, I_5)$ random vectors.

The five true change-points occur at $k = 75, 376, 425, 525$ and 575. We analyze it by examining the change in multivariate means using grid_size_scale at 0.05 and confidence at 0.9. The code implementation is as follows:

```
# Generate model (M2)
exchange_cor_matrix <- function(d, rho){
  tmp <- matrix(rho, d, d)
  diag(tmp) <- 1
  return(tmp)
}
set.seed(1)
d <- 5
n <- 1000
cp_sets <- round(n*c(0,cumsum(c(0.075,0.3,0.05,0.1,0.05)),1))
mean_shift <- c(-3,0,3,0,-3,0)/sqrt(d)
mean_shift <- sign(mean_shift)*ceiling(abs(mean_shift)*10)/10
rho_sets <- 0.5
sigma_cross <- list(exchange_cor_matrix(d,0))
ts <- MAR_MTS_Covariance(n, 2, rho_sets, cp_sets=c(0,n), sigma_cross)
noCP <- length(cp_sets)-2
no_seg <- length(cp_sets)-1
for(rep_index in 1:2){
  for(index in 1:no_seg){ # Mean shift
    tau1 <- cp_sets[index]+1
    tau2 <- cp_sets[index+1]
    ts[[rep_index]][,tau1:tau2] <- ts[[rep_index]][,tau1:tau2] + mean_shift[index]
  }
}
ts <- ts[1][[1]]

# SNCP in the change of multivariate means
result <- SNSeg_Multi(ts, paras_to_test = "mean", confidence = 0.9,
                      grid_size_scale = 0.05, grid_size = NULL)

# Output
result$est_cp
[1] 73 369 426 525 575
result$grid_size
[1] 50
result$critical_value
[1] 415.8649
```

According to the output, the estimated change-points are $\hat{k} = 73, 369, 426, 525$ and 575 with a window size $h = 50$ and a critical value of 415.8649. This result matches the true change-point locations.

The output of `SNSeg_Multi` also allows for the use of function `max_SNsweep` for plotting the segmentation test statistics. The code implementation is as follows:

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

and Figure 3 plots the associated SN test statistics and identified change-points.

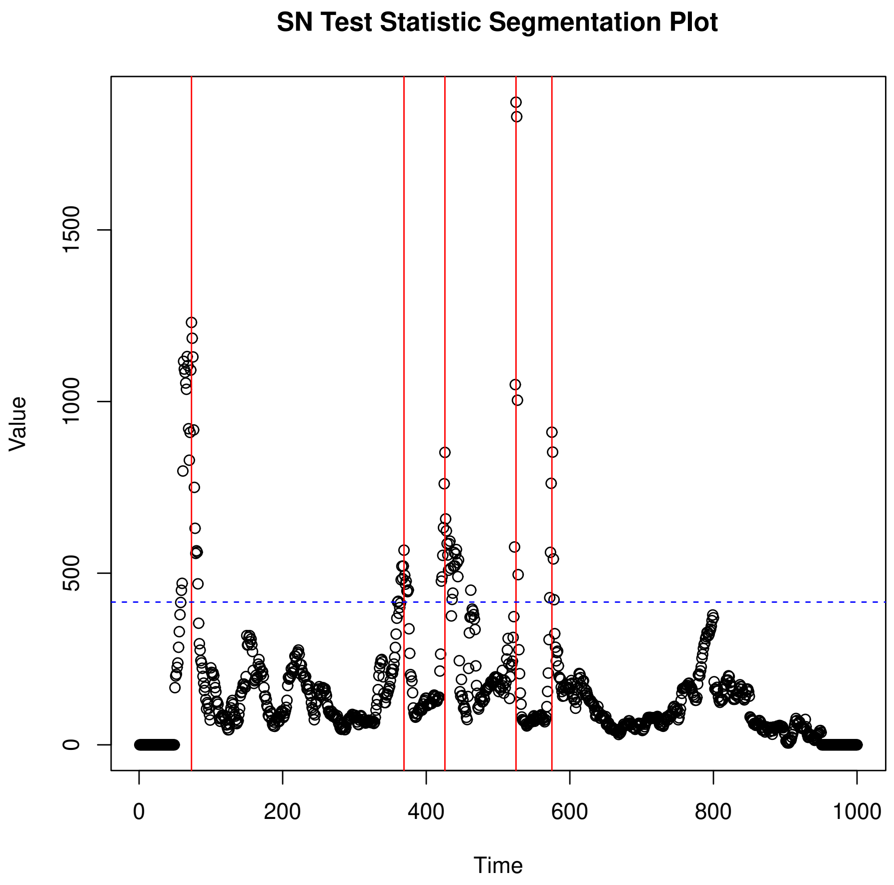

Figure 3: SN-based segmentation plot for multivariate time series based on the change in multivariate means. The blue horizontal line represents the critical value and the red solid lines are detected change-point locations.

3.3 SNHD Procedures for High-Dimensional Time Series

The function `SNSeg_HD` is specifically designed to estimate change-points in the mean functional of high-dimensional time series.

```
    SNSeg_HD(ts, confidence = 0.9, grid_size_scale = 0.05, grid_size = NULL)
```

Its input arguments are the same as the function `SNSeg_Multi` except for one difference that the dimension of the input time series `ts` must be at least 10. When called, `SNSeg_HD` returns a list containing `grid_size`, `SN_sweep_result`, `est_cp`, `confidence` and `critical_value` that are similar to those described by `SNSeg_Uni` and `SNSeg_Multi`.

Example: mean change in high-dimensional time series

We generate high-dimensional time series data based on the following simulation setting:

$$(\text{HD}) : Y_t = \mu_i + X_t, \;\; \tau_{i-1} + 1 \leq t \leq \tau_i, \;\; i = 1, 2, \cdots, 6,$$

where X_t is a sequence of i.i.d. $N(0, I_{100})$ random vectors, five change-points are evenly located at $(\tau_1, \tau_2, \cdots, \tau_5) = (100, 200, \cdots, 500)$, and $\mu_1 = \mathbf{0}_{100}$, $\theta_i = \mu_{i+1} - \mu_i$, $\theta_i = (-1)^i(\mathbf{1}_5^\top, \mathbf{0}_{95}^\top)^\top \times \sqrt{4/5}$ for $i = 1, 2, \cdots, 5$.

We apply `SNSeg_HD` to analyze this time series with `grid_size_scale` at 0.05 and `confidence` at 0.9.

```
# Generate model (HD)
n <- 600
p <- 100
nocp <- 5
cp_sets <- round(seq(0,nocp+1,1)/(nocp+1)*n)
num_entry <- 5
kappa <- sqrt(4/5) # Wang et al(2020)
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){ # Mean shift
  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]
}

# SNHD for high-dimensional means
result <- SNSeg_HD(ts, confidence = 0.9, grid_size_scale = 0.05, grid_size = NULL)

# Output
result$est_cp
[1]  94 199 301 406 500
```

As observed in the example, `SNSeg_HD` successfully detects all the change-points in this high-dimensional time series. This result demonstrates the effectiveness and feasibility of using SN algorithms for change-point detection in the context of high-dimensional time series.

3.4 Generate SN Test Statistics

As shown in Section 2, the success of SNCP and SNHD depends on the local SN test statistic $T_{1:n}(k)$ and $T_{1,n}^U(k)$ for $k = 1, \cdots, n$. To facilitate, the function `maxSNsweep` allows users to compute and plot these test statistics along with the identified change-points.

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

It takes the following arguments:

- `SN_result`: A list of elements served as output from the functions `SNSeg_Uni`, `SNSeg_Multi`, or `SNSeg_HD`.
- `plot_SN`: A boolean value to return an SN test statistics 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 SN critical value if `critical_loc = TRUE`.

When called, `max_SNsweep` returns the maximal SN test statistic, namely `SN_test_statistic`, for each time point k. More importantly, it generates a segmentation plot for these SN test statistics to analyze the time series data. Users are able to determine whether to mark the change-point locations and the critical value within the plot.

As an illustration of `max_SNsweep`, suppose we apply `SNSeg_Uni` to estimate change-points and saved the output as `SN_result`. The following code can be implemented to generate the segmentation plot, which is already given in the lower panel of Figure 2.

```
SN_result <- SNSeg_Uni(ts, paras_to_test = "variance", confidence = 0.9,
                grid_size_scale = 0.067, grid_size = NULL, plot_SN = FALSE)
max_SNsweep(SN_result, plot_SN = TRUE, est_cp_loc = TRUE, critical_loc = TRUE)
```

4 Summary

In this paper, we have introduced the **R** package **SNSeg** that provides implementations of SN procedures for change-points detection in univariate, multivariate, and high-dimensional time series. We have described the main functions of the package, namely `SNSeg_Uni`, `SNSeg_Multi`, `SNSeg_HD` and `max_SNsweep`, which enable the detection of multiple change-points based on single or multiple parameters. Furthermore, we have presented examples demonstrating the usage of the package, including visualizing both the time series data and the segmentation plots of SN test statistics.

The **SNSeg** package offers a comprehensive set of tools for researchers and practitioners to effectively identify change-points in time series data, facilitating the analysis and understanding of temporal patterns and dynamics.

Bibliography

D. W. Andrews. Heteroskedasticity and autocorrelation consistent covariance matrix estimation. *Econometrica*, pages 817–858, 1991. [p3]

D. Barry and J. A. Hartigan. A bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88:309–319, 1993. [p1]

B. Eichinger and C. Kirch. A mosum procedure for the estimation of multiple random change points. *Bernoulli*, 24(1):526–564, 2018. [p1]

C. Erdman and J. W. Emerson. bcp: An R package for performing a bayesian analysis of change point problems. *Journal of Statistical Software*, 23(3):1–13, 2007. [p1]

D. M. Hawkins and K. Zamba. A change-point model for a shift in variance. *Journal of Quality Technology*, 37(1):21–31, 2005. [p1]

D. M. Hawkins, P. Qiu, and C. W. Kang. The changepoint model for statistical process control. *Journal of Quality Technology*, 35(4):355–366, 2003. [p1]

N. A. James and D. S. Matteson. ecp: An R package for nonparametric multiple change point analysis of multivariate data. *Journal of Statistical Software*, 62(7):1–25, 2014. [p1]

R. Killick and I. A. Eckley. changepoint: An R package for changepoint analysis. *Journal of Statistical Software*, 58(3):1–19, 2014. [p1]

R. Killick, P. Fearnhead, and I. A. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598. [p1]

D. S. Matteson and N. A. James. A nonparametric approach for multiple change point analysis of multivariate data. *Journal of the American Statistical Association*, 109(505):334–345, 2014. [p1]

A. Meier, C. Kirch, and H. Cho. mosum: A package for moving sums in change-point analysis. *Journal of Statistical Software*, 97(8):1–42, 2021. [p1]

W. K. Newey and K. D. West. A simple, positive semi-definite, heteroskedasticity and autocorrelation-consistent covariance matrix. *Econometrica*, 55:703–708, 1987. [p3]

G. J. Ross. Sequential change detection in the presence of unknown parameters. *Statistics and Computing*, 24(6):1017–1030, 2014. [p1]

G. J. Ross. Parametric and nonparametric sequential change detection in R: The cpm package. *Journal of Statistical Software*, 66(3):1–20, 2015. [p1]

X. Shao. A self-normalized approach to confidence interval construction in time series. *Journal of the Royal Statistical Society: Series B*, 72(3):343–366, 2010. [p1]

X. Shao. Self-normalization for time series: a review of recent developments. *Journal of the American Statistical Association*, 110:1797–1817, 2015. [p1]

X. Shao and X. Zhang. Testing for change points in time series. *Journal of the American Statistical Association*, 105(491):1228–1240, 2010. [p1]

G. Wang and C. Zou. cpss: an package for change-point detection by sample-splitting methods. *Journal of Quality Technology*, 55:61–74, 2023. [p1]

R. Wang, C. Zhu, S. Volgushev, and X. Shao. Inference for change points in high-dimensional data via selfnormalization. *The Annals of Statistics*, 50(2):781–806, 2022. [p2, 5]

A. Zeileis. Implementing a class of structural change tests: An econometric computing approach. *Computational Statistics & Data Analysis*, 50(11):2987–3008, 2006. [p1]

A. Zeileis, F. Leisch, K. Hornik, and C. Kleiber. strucchange: An r package for testing for structural change in linear regression models. *Journal of Statistical Software*, 7(2):1–38, 2002. [p1]

A. Zeileis, C. Kleiber, W. Krämer, and K. Hornik. Testing and dating of structural changes in practice. *Computational Statistics & Data Analysis*, 44(1–2):109–123, 2003. [p1]

Z. Zhao, F. Jiang, and X. Shao. Segmenting time series via self-normalisation. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(5):1699–1725, 2022. [p1, 3, 7, 8, 10]

C. Zou, G. Wang, and R. Li. Consistent selection of the number of change-points via sample-splitting. *Annals of Statistics*, 48(1):413, 2020. [p1]

Shubo Sun
University of Miami
Address
Country
(ORCiD if desired)
author1@work

Zifeng Zhao
University of Notre Dame
Mendoza College of Business
Notre Dame, IN, USA
zifeng.zhao@nd.edu

Feiyu Jiang
Fudan University
Department of Statistics and Data Science,School of Management
Shanghai, China
jiangfy@fudan.edu.cn

Xiaofeng Shao
University of Illinois at Urbana-Champaign
Department of Statistics
Champaign, IL, USA
ORCiD:0000-0001-5822-8209
xshao@illinois.edu