Online Structural Change-Point Detection of High-dimensional Streaming Data via Dynamic Sparse Subspace Learning

Ruiyu Xu, Jianguo Wu, Xiaowei Yue & Yongxiang Li

To cite this article: Ruiyu Xu, Jianguo Wu, Xiaowei Yue & Yongxiang Li (2022): Online Structural Change-Point Detection of High-dimensional Streaming Data via Dynamic Sparse Subspace Learning, Technometrics, DOI: 10.1080/00401706.2022.2046171

To link to this article: https://doi.org/10.1080/00401706.2022.2046171

Published online: 01 Apr 2022.

Article views: 123

Submit your article to this journal

View supplementary material

View related articles

View Crossmark data
Online Structural Change-Point Detection of High-dimensional Streaming Data via Dynamic Sparse Subspace Learning

Ruiyu Xu\textsuperscript{a}, Jianguo Wu\textsuperscript{a}, Xiaowei Yue\textsuperscript{b}, and Yongxiang Li\textsuperscript{c}

\textsuperscript{a}Department of Industrial Engineering and Management, Peking University, Beijing, China; \textsuperscript{b}Department of Industrial & Systems Engineering, Virginia Tech, Blacksburg, VA; \textsuperscript{c}Department of Industrial Engineering and Management, Shanghai Jiao Tong University, Shanghai, China

ABSTRACT

High-dimensional streaming data are becoming increasingly ubiquitous in many fields. They often lie in multiple low-dimensional subspaces, and the manifold structures may change abruptly on the time scale due to pattern shift or occurrence of anomalies. However, the problem of detecting the structural changes in a real-time manner has not been well studied. To fill this gap, we propose a dynamic sparse subspace learning approach for online structural change-point detection of high-dimensional streaming data. A novel multiple structural change-point model is proposed and the asymptotic properties of the estimators are investigated. A tuning method based on Bayesian information criterion and change-point detection accuracy is proposed for penalty coefficients selection. An efficient Pruned Exact Linear Time based algorithm is proposed for online optimization and change-point detection. The effectiveness of the proposed method is demonstrated through several simulation studies and a real case study on gesture data for motion tracking. Supplementary materials for this article are available online.

1. Introduction

High-dimensional streaming data are ubiquitous in many fields such as bioinformatics, engineering, finance and social sciences. For example, in biological studies, neurons being monitored could generate hundreds or thousands of time series signals (Qiu et al. 2016). In image signal processing, each dynamic image in a high resolution video could consist of more than 100,000 pixels. In semiconductor manufacturing, hundreds of sensors are installed in the production system for real-time monitoring of the manufacturing condition (Zhangetal. 2020). In gesture tracking, tens of sensors are mounted to dynamically capture the positions of body joints (Jiao, Chen, and Gu2018). The relationship or correlation among these dimensions is of great value for research, as it provides insights into regularities and inter-dependencies between observed variables (Kolar and Xing 2012). Usually, the correlation or dependence structure is sparse, that is, a variable is only correlated with a small proportion of other variables. Besides, the dependence structure may change over time and the change-points often imply events or anomalies occurring at that moment. For instance, changes in the correlation between brain nerves may represent shifts in thinking patterns (Haslbeck and Waldorp 2015) or the onset of seizure (Dhulekar et al. 2015). Changes in the correlation between image pixels may indicate transitions in the subject of the video (Tierney, Gao, and Guo 2014). In tunnel excavation process, the torque of the cutterhead of the Tunnel Boring Machine (TBM) has a linear relationship with the penetration rate, and the change in the regression coefficient may result from the change of geological condition (Shi et al. 2019). Therefore, online detection of change in correlation or inter-dependence is of great importance to determine whether an event or anomaly has recently occurred in the system.

In this article, we refer to the change in linear relationship as a structural change, that is, change of the manifold structure capturing the linear relationship among the high-dimensional streaming data. The structural change-points separate the multivariate streaming data into multiple segments, where each segment has a unique structure modeling their relationship. Multiple change-point problems have been actively studied in many fields, for example, economics, climatic time series (Wu, Chen, and Zhou 2016; Aminikhanghah and Cook 2017; Wu et al. 2019). However, these problems often refer to detection of breaks in trend or distributional parameters, for example, a shift in mean or variance, while structural change detection focuses on detecting the changes of the underlying relationships among different dimensions. The structural change-point detection problem, especially the online one, has not been well explored compared with the traditional multiple change-point detection problems. Due to the “curse of dimensionality” (Bellman 1966), it is often challenging to detect these change-points accurately and timely for high-dimensional streaming data. Too many variables constitute an extremely complex correlation structure that is hard to estimate. Noise contamination and insufficient sample size further increase the difficulty of estimation.

The Gaussian graphic model (GGM, Dempster 1972) is a widely used method and continues to attract much attention...
to study the inter-dependence structure of multiple variables. A common assumption is that the sample $X \sim N_p(0, \Sigma)$ is a $p$-dimension Gaussian vector. Let $\Omega := \Sigma^{-1}$ denote the precision matrix, with entries $(\omega_{ij})$, $1 \leq i, j \leq p$. It can be easily shown that the precision matrix $\Omega$ encodes the conditional independence structure among the variables. Variable $i$ and $j$ are conditionally independent given all other coordinates of $X$ if and only if the entry $\omega_{ij}$ of the precision matrix is zero. Meinshausen and Bühlmann (2006) were the first to combine GGM with LASSO to get a sparse precision matrix $\Omega$, and later a more systematic approach named Graphical LASSO was proposed by Tibshirani et al. (2005). Since then, there has been much similar work on estimating a single precision matrix $\Omega$ based on $n$ independent samples (Yuan and Lin 2006; Drton and Perlman 2008; Rothman et al. 2008; Foygel and Drton 2010; Ren et al. 2015). In addition, some Gaussian graphical models for functional data are also developed to model the correlations of the true values as the length of segments increases, and the estimated coefficients satisfy a LASSO subspace detection property. A Pruned Exact Linear Time (PELT) (Killick, Fearnhead, and Eckley 2012) based dynamic programming algorithm is developed for online model optimization and change-point detection.

The rest of this article is organized as follows. In Section 2, two basic assumptions for sparse subspace learning are given, and a multiple structural change-point model is formulated. The asymptotic properties of the proposed model are investigated in Section 3. In Section 4, we show how to solve the optimization problem sequentially via a customized PELT algorithm, and propose some strategies to determine the penalty coefficients and proper hyperparameters to improve the computational efficiency. In Section 5, numerical experiments with synthetic and real data are conducted to demonstrate and validate the effectiveness of the proposed algorithm. Section 6 presents the discussions and conclusions.

2. Multiple Structural Change-point Modeling via Dynamic Sparse Subspace Learning

In this section, we first illustrate the structural change and the motivation of using dynamic sparse subspace learning through a simple example, and then introduce the subspace assumption and the self-expressive assumption (property), which lay the foundation for SSC. After that, a multiple structural change-point modeling approach is formulated for change detection.
2.1. Illustration of the Structural Change

As mentioned earlier, the structural change here is defined as the change of linear manifold capturing the relationship among the times series of the streaming data. Figure 1 is a simple example with three time series illustrating the concept of structural change. In Figure 1(a), there are two change-points, that is, $t = 1000$ and $2000$. Once a change-point occurs, the linear relationship among these variables changes, for example, from $Z = X - Y$ to $Z = 4X + 2Y$ at $t = 1000$. Figure 1(b) shows the time series in a three-dimensional space. Clearly, each segment in Figure 1(a) corresponds to a plane or a linear manifold in Figure 1(b), and we can easily see the transition from one to another.

In practical applications with high-dimensional streaming data, however, a variable may be related with only a small number of other variables, yielding multiple clusters of much lower dimensions, which makes the subspace learning very challenging. To overcome this issue, we borrow the idea of SSC and develop a dynamic subspace learning approach. The technical details are given in the following sections.

2.2. Notations and Basic Assumptions

Consider a $p$-dimensional (e.g., $p$-channels) streaming data $(Y_1, \ldots, Y_p)$, where each dimension is of length $N$ on the time scale, for example, $Y_t = (Y_{t1}, Y_{t2}, \ldots, Y_{tn})$. We assume

$$Y_{ij} = X_{ij} + \epsilon_{ij}, \quad i = 1, \ldots, p, \quad j = 1, \ldots, N,$$

where $X_{ij}$ is the true value and $\epsilon_{ij}$ is the independent noise with mean $E[\epsilon_{ij}] = 0$ and variance $\text{var}[\epsilon_{ij}] = \sigma^2_{\epsilon}$. We assume there is no autocorrelation in the noise. To facilitate understanding, we could treat a discrete time series $X_i$ as a functional sample of $X_i(t)$, and assume that these functions can be partitioned into $L$ different subspaces $S_i, i = 1, \ldots, L$. Functions in the same subspace have strong cross-correlations, while functions in different subspaces have no cross-correlations. To facilitate subspace learning, we present two required assumptions, which are similar to Zhang’s work (Zhang et al. 2020).

Assumption A1 (Subspace Assumption). It is assumed that the multivariate streaming data can be partitioned into different subspaces. Each subspace or translated subspace $S_i$ is defined as the set of all functions formed by linearly combining the $d_i$ basis functions $\Phi_i = (\phi_{i1}(t), \ldots, \phi_{id_i}(t))^\prime$ with a translation or shift function $\phi_{0i}(t)$

$$S_i \equiv \left\{ X(t)|X(t) = \sum_{q=1}^{d_i} a_q \phi_{iq}(t) + \phi_{0iq}(t), a_q \in \mathcal{R} \right\}. \tag{2}$$

If orthogonal basis functions are considered, then

$$\int \phi_{iq}(t) \phi_{ij}(t) dt = 0, \quad \text{for } i, j = 1, \ldots, d_i, i \neq j.$$

Note that if $\phi_{0i}(t) \neq 0$, it is a translated subspace, as it does not contain the origin (Nowinski 1981). This subspace assumption tells us that the time series within the same subspace share common basis functions, thus, are expected to have strong correlations or linear relationship. This assumption is reasonable in many cases. For example, in EEG signals (Thatcher, North, and Biver 2005), the collected signals in a certain functional zone of the brain may be highly correlated, as they are the superposition of electrical signals from the same set of neurons. The activity of these neurons forms the shared basis functions. As there are many functional zones in the brain, the EEG signals can be partitioned into multiple subspaces. Similar phenomenon can also be found in seismic signals caused by multiple earthquake events (Kitov et al. 2019), and river water flows within the same region and across multiple regions.

In practical applications, the collected streaming data are discrete. In such a case, the basis vectors, for example, $\Phi_i = (\phi_{i1}(t_1), \ldots, \phi_{i1}(t_n))$ instead of the basis functions can be used to represent the subspace assumption. The formed vector subspace is actually an affine space. With this assumption, we have $X = A\Phi$ where $X = (X_1, \ldots, X_p)$ is the true-value time series data in $\mathcal{R}^{p \times N}, \Phi = (\phi_{11}, \ldots, \phi_{1d_1}, \ldots, \phi_{Ld_L})^\prime$ is the discrete basis matrix in $\mathcal{R}^{d \times N}$, $d = \sum_{l=1}^{L} d_l$ and $A$ is a sparse coefficient matrix in $\mathcal{R}^{p \times d}$ with entry $\beta_{ij} \neq 0$ if and only if $X_i$ belong to the subspace $S_l$ which contains the basis function $\phi_{lj}$.

Note that here the dimensionality of the space is the length of the time series $N$ (row space of $X$), not the number of time series $p$ (column space). Each time series here represents a point, with $p$ points in total. This representation is different from Figure 1(b) where the column space is used. In column space, there is only one subspace, while in row space, there may be multiple such spaces, with each one corresponding to a cluster of the time series. Figure 2 is an illustrative example showing three subspaces in $\mathcal{R}^3$. Intuitively, given sufficient time series, each time series can be efficiently represented as a linear combination of other time series in the same subspace. For example, in subspace $S_1$, point $X_2 = X_1 + P$ where $P = a_1 (X_4 - X_3) + a_2 (X_6 - X_5)$ for certain points $X_3, X_4, X_5$ and $X_6$ in $S_1$. This self-expressive property is summarized in Assumption A2 as follows.

Assumption A2 (Self-Expressive Assumption). If there are sufficient time series from each subspace, for example, $p_i > d_i$ for $l = 1, \ldots, L$, where $p_l$ is the number of time series in subspace $S_l$, we have and $d_l$ is the subspace dimension, then $X_i = (X_{i1}, \ldots, X_{IN})$ is self-expressive, that is, for all $i \in P_l$ where $P_l$ is the set of time series indices of subspace $S_l$, we have

$$X_i = \sum_{j \in P_l \backslash i} \beta_{ij} X_j, \quad \text{for } i = 1, \ldots, p. \tag{3}$$

With this assumption, we have $X = BX$ and $A = BA$, where $X = (X_1, \ldots, X_p)$ is the noise-free time series data in $\mathcal{R}^{p \times N}$, $A$ is the sparse coefficient matrix of basis functions in $\mathcal{R}^{d \times N}$, $B$ is a sparse coefficient matrix in $\mathcal{R}^{p \times d}$ with entry $\beta_{ij} = 0$ and $\beta_{ij} = 0$ if $X_i$ and $X_j$ belong to different subspaces. This self-expressive property can be easily derived based on Assumption A1. A detailed discussion is provided in the supplementary materials.

Note that the coefficient matrix $B$ is not unique and could have infinite solutions. Naturally, an optimal $B$ can be obtained by minimizing the objective function with the $l_q$-norm of the solution, that is,

$$\arg \min_B ||B||_q \text{ s.t. } X = BX, \quad \beta_{ii} = 0, \quad \text{for } i = 1, \ldots, p.$$
As $q$ decreases from 1 to 0, the sparsity of $B$ increases. Although a sparsest $B$ could be obtained with $l_0$-norm, the problem turns to a NP-hard problem, which is difficult to solve. As an alternative, $l_1$-norm is widely used with good sparsity property and computational efficiency. It is obvious that this solution has sparsity between subspaces, that is, $\beta_{ij} = 0$ if $X_i$ and $X_j$ belong to different subspaces.

2.3. Model Formulation

The self-expressive property states that each time series can be efficiently reconstructed by a linear combination of other time series in the same subspace. Considering the existence of measurement noise, the sparse representation $B$ can be obtained by minimizing the following objective function with an $l_1$ penalty

$$\min_{\beta_{ij} \neq 0} \sum_{i=1}^{p} \left\{ \frac{1}{2} \sum_{t=1}^{N} \left( Y_{it} - \sum_{j \neq i} \beta_{ij} Y_{jt} \right)^2 + \lambda_1 \sum_{j \neq i} |\beta_{ij}| \right\},$$

s.t. $\beta_{ii} = 0$, for $i = 1, \ldots, p$. (4)

where $\lambda_1$ is the penalty weight to control the sparsity in the representation. As the representation of each time series is independent of those of other time series, the above optimization problem can be efficiently solved efficiently using the LASSO algorithm.

In the traditional sparse subspace clustering, the dimension of the data points and the correlation structure are fixed. However, in our case, the length of the streaming data dynamically increases and there may be abrupt structural changes, that is, changes in $\beta_{ij}, j \neq i$ at some unknown change-points. For a $p$-channel streaming data of length $N$, suppose there are in total $C$ change-points $\{\tau_1, \ldots, \tau_C\}$ with $0 < \tau_1 < \cdots < \tau_C < N$, which partitions the streaming data into $C + 1$ segments. For notational convenience, we define $\tau_0 = 0$ and $\tau_{C+1} = N$.

In the estimation of change-point models, the fused lasso is one of the most popular techniques, which penalizes the $l_1$-norm of both the coefficients and their successive differences (Tibshirani et al. 2005; Tierney, Gao, and Guo 2014; Zhang et al. 2020). Using the fused lasso, the problem can be formulated as

$$\min_{\beta_{ij} \neq 0, \tau_i} \sum_{i=1}^{N} \left[ \frac{1}{2} \left( Y_{it} - \sum_{j \neq i} \beta_{ij} Y_{jt} \right)^2 + \lambda_1 \sum_{j \neq i} |\beta_{ij}| \right] + \lambda_2 \sum_{i=1}^{N} |\beta_{ij} - \beta_{ij-1}|, i = 1, \ldots, p. (5)$$

The above optimization can be achieved by the Fast Iterative Shrinkage-thresholding Algorithm (FISTA, Beck and Teboulle 2009). However, this formulation is inherently offline and cannot be efficiently solved in a real-time or sequential manner for online applications. To overcome this problem, we propose a new model formulation that can be sequentially optimized.
### 3. Asymptotic Properties

In this section, the asymptotic properties of the estimators of DSSL are established. We first discuss the asymptotic properties of the optimal solution $\mathbf{B}$ of Equation (4) with no change-points. The convergence property and sparsity property are proved as the sample size $N_0$ goes to infinity. Then, we prove that, as the sample size $N$ of each segment goes to infinity, the locations of the change-points could be precisely detected and the estimated coefficient matrices $\hat{\mathbf{B}}^{(c)}$ of each segment satisfy the LASSO Subspace Detection Property introduced by (Wang and Xu 2013), a very desirable property for subspace learning. For convenience, we provide the definition of the LASSO Subspace Detection Property as follows.

**Definition 1 (LASSO Subspace Detection Property).** The estimated coefficient matrices $\hat{\mathbf{B}}$ is called satisfying the LASSO Subspace Detection Property if for all $i = 1, \ldots, p$.

1. $\hat{\beta}_i$ is not a zero vector, that is, the solution is nontrivial.
2. Nonzero entries of $\hat{\beta}_i$ correspond to only the time series sampled from the same subspace as $Y_i$.

For simplicity yet without loss of generality, we only consider one sparse regression in the following analysis, for example, $Y_i$ as the response variable. The problem of including all sparse regressions can follow the same approach.

Denote the $A_i \in \mathcal{R}^{1 \times d}$ as the $i$th row of $\mathbf{A}$ and the $A_{-i} \in \mathcal{R}^{(p-1) \times d}$ as the rest rows.

**Theorem 1.** Under (A1–A2), for the optimization problem without change-points

$$\min_{\beta_{ij}\neq 2} \frac{1}{2} \sum_{i=1}^{N_0} \sum_{j=1}^{\beta_i} \left( Y_i - \sum_{j=1}^{\beta_i} \beta_{ij} Y_{jt} \right)^2 + \lambda_1(N_0) \sum_{j=1}^{\beta_i} |\beta_{ij}|, \text{ s.t. } \beta_{ii} = 0,$$

where $\lambda_1(N_0)$ is a function of sample size $N_0$.

1. When $\lambda_1(N_0)/N_0 \rightarrow 0$, the estimated $\hat{\beta}_i$ of (8) converges to a sparse vector $A_iA_{-i}^{-1}(A_{-i}A_{-i}^{-1} + \sigma_0^2I_{p-1})^{-1}$ as $N_0$ goes to infinity.
2. If $\lambda_1(N_0)/N_0 \rightarrow \lambda_{1,0} \geq 0$, the estimated $\hat{\beta}_i$ of (8) converges to

$$\arg \min_{\beta_i} \frac{1}{2} \| A_i - \beta_i A_{-i} \|_2^2 + \frac{\sigma_0^2}{2} \| \beta_i \|_1^2 + \lambda_{1,0} \| \beta_i \|_1$$

as $N_0$ goes to infinity.

3. The estimated $\hat{\beta}_i$ satisfies the LASSO Subspace Detection Property when

$$0 < \lim_{N_0 \to \infty} \lambda_1(N_0)/N_0 = \lambda_{1,0} < \| A_iA_{-i}^{-1} \|_\infty,$$

as $N_0$ goes to infinity.

The proof of the theorem is provided in Appendix A of the supplementary materials. This theorem tells us when $\lambda_1(N_0)/N_0 \to 0$, the optimization is determined only by the first term and thus, it has a closed-form solution. However, it does not guarantee the solution is sparse. On the other hand, if we select a sequence for $\lambda_1(N_0)$ satisfying $\lambda_1(N_0)/N_0 \to \lambda_{1,0} > 0$, $\hat{\beta}_i$ converges to the estimator of an elastic net regularized regression problem, and the sparsity property can be guaranteed. Besides, as long as $0 < \lambda_{1,0} < \| A_iA_{-i}^{-1} \|_\infty$, the estimated representation coefficient $\hat{\beta}_i$ satisfies the LASSO Subspace Detection Property, that is, the time series is only represented by those from the same subspace. To better achieve the sparse structure of the streaming data and guarantee the LASSO Subspace Detection Property, we set $\lambda_1(N_0) = \lambda_{1,0}N_0$, $\lambda_{1,0} > 0$ in this article, that is, $\lambda_1$ linearly increases with the segment length.

For the change detection problem, let $\gamma^0_\epsilon = \tau^0_\epsilon / N$ for $\epsilon = 1, \ldots, C$, $\gamma_0 = (\gamma^0_1, \gamma^0_2, \ldots, \gamma^0_C)$ where the superscript 0 refers to the true value. Similarly, define $\gamma_\epsilon = \tau_\epsilon / N$, $\gamma = \gamma_0$. **
(γ_1, γ_2, ..., γ_C). Define \( \delta(c) = \tau_c^0 - \tau_{c-1}^0 \) as the length of one segment. Note that \( \gamma^0 \) is set to be a constant vector as \( N \) goes to infinity.

**Theorem 2.** Under (A1-A2), for any time series \( Y \), there exists a constant \( \lambda_{2,0} > 0 \), when \( \lambda_2 > \lambda_{2,0} \) and as \( N \) goes to infinity with a fixed \( \gamma^0 \),

1. The optimal solution of optimization problem (6) satisfies,

   \[ \rho(\gamma^0, \hat{\gamma}) \xrightarrow{P} 0, \]

   \[ \rho(\hat{\gamma}, \gamma^0) \xrightarrow{P} 0, \]

   where \( \rho(a, b) = \max_i (\min_j |a_i - b_j|) \) is a function representing the difference of \( a \) and \( b \).

2. If the number of change-points \( \hat{C} \) is correctly estimated with properly specified \( \lambda_2 \), then

   \[ \hat{\gamma}_c \xrightarrow{P} \gamma^0_c, \quad c = 1, \ldots, C, \]

   and the estimated coefficient matrices \( \hat{B}^{(c)} \), \( c = 1, \ldots, C + 1 \) satisfy the asymptotic properties of Theorem 1.

The proof of Theorem 2 is provided in Appendix B of the supplementary materials. Theorem 2(1) shows that for any true change-point, there exists a detected change-point from the optimization problem (7), such that the relative locations of the two converge in probability to each other. Similarly, for each detected change-point, there exists a true change-point, such that their relative locations converge to each other. In other words, although there may appear more than one detected change-point around each true change-point if \( \lambda_2 \) is not large enough, the relative locations of these detected change-points will converge to that of the true one. Theorem 2(2) tells us that if the number of change-point is correctly estimated with properly specified \( \lambda_2 \) in the optimization problem (7), then the relative location of each detected change-point will converge to that of the true change-point, and the estimated coefficient matrix of each segment will satisfy the asymptotic properties stated in Theorem 1. We have to mention that although Theorems 1 and 2 are offline properties, they still provide us insights into the online detection process, for example, with observations being collected continuously after a change-point, the detection becomes more and more accurate.

### 4. Online Optimization via PELT Algorithm

In this section, we show the specific approach to solve the optimization problem (6). We first introduce the PELT algorithm, which is an efficient sequential optimization algorithm. Then, we discuss how to determine the penalty coefficients and proper parameters for the implementation of PELT algorithm in DSSL model.

#### 4.1. The PELT based Optimization Algorithm

Various algorithms have been proposed to solve the optimization problem of the following form for multiple change-points models:

\[
\min_{m, \tau_1, \ldots, \tau_m} \sum_{c=1}^{m+1} \left\{ \text{Cost} \left( \bar{Y}_{\tau_{c-1} + 1: \tau_c} \right) + f(m) \right\},
\]

where \( \text{Cost}(\cdot) \) is a cost function for a segment, \( m \) is the number of change-points and \( f(m) \), for example, \( f(m) = \beta(m + 1) \), is a penalty term to guard against overfitting. Binary Segmentation (BS) algorithm proposed by Scott and Knott (1974) is one of the most established search method with an \( \mathcal{O}(n \log n) \) computational cost for \( n \) samples. It begins by initially applying the single change-point method to the entire dataset. The dataset is split into two segments and the single change-point detection method is carried out again for these two segments independently. This procedure is repeated until no further change-points are detected. The advantage of the BS method is that it is computationally efficient. But it does not guarantee to find the global optimal solution.

The optimal partitioning (OP) algorithm proposed by Jack-son et al. (2005) focuses on finding the latest change-point (LCP) at each time step. It relates the optimal value of the cost function to the cost for the optimal partition of the data prior to the latest change-point plus the cost for the segment from the latest change-point to the end of the data. Let \( F(n) \) denote the optimal value of the objective function for data \( Y_{1:n} \) and \( \tau_n = \{ \tau : 0 = \tau_0 < \tau_1 < \cdots < \tau_n < \tau_{n+1} = n \} \) be the set of all possible vectors of the change-points for this dataset. Set \( f(m) = \beta(m + 1) \) and \( F(0) = -\beta \). Then,

\[
F(n) = \min_{\tau \in \tau_n} \left\{ \sum_{c=1}^{m+1} \left\{ \text{Cost} \left( \bar{Y}_{\tau_{c-1} + 1: \tau_c} \right) + \beta \right\} \right\}
\]

\[
= \min_{\tau \in \tau_n} \left\{ \min_{\tau^0 \in \{0, \ldots, n-1\}} \sum_{c=1}^{m} \left\{ \text{Cost} \left( \bar{Y}_{\tau^0+1: \tau_c} \right) + \beta \right\} + \text{Cost} \left( \bar{Y}_{\tau^0+1: n} \right) + \beta \right\}
\]

\[
= \min_{\tau \in \tau_n} \{ F(t) + \text{Cost} \left( \bar{Y}_{t+1:n} \right) + \beta \}. \quad (10)
\]

As \( F(t) \) only needs to be calculated once and can be used repeatedly in the following steps, this recursion can be solved sequentially or dynamically for \( n = 1, \ldots, N \), and the computational cost is \( \mathcal{O}(N^2) \). Although the OP algorithm can find the global optimal solution, it is still far from being computationally competitive with the BS method. To this end, Killick, Fearnhead, and Eckley (2012) proposed the PELT by adding a pruning step for the OP algorithm to reduce the computational cost to \( \mathcal{O}(N) \). In the OP algorithm, to solve \( F(n) = \min_{t \in \{0, \ldots, n-1\}} \{ F(t) + \text{Cost} \left( \bar{Y}_{t+1:n} \right) + \beta \} \), we need to consider all time points prior to time \( n \). But in the PELT algorithm, we remove the time points that can never be the optimal LCP. Specifically, we optimize \( F(n) = \min_{t \in R(n)} \{ F(t) + \text{Cost} \left( \bar{Y}_{t+1:n} \right) + \beta \} \) where \( R(n) \) is the set of all time points that could be the possible LCP in terms of optimality at time \( n \). The following theorem (Killick, Fearnhead, and Eckley 2012) provides a simple condition under which such pruning can be performed.

**Theorem 3** (Killick, Fearnhead, and Eckley 2012). If there exists a constant \( K \) such that for all \( s < n < T \),

\[
\text{Cost} \left( \bar{Y}_{s+1:n} \right) + \text{Cost} \left( \bar{Y}_{n+1:T} \right) + K \leq \text{Cost} \left( \bar{Y}_{s+1:T} \right). \quad (11)
\]
Then if
\[ F(s) + \text{Cost} \left( Y_{s+1:n} \right) + K \geq F(n) \] (12)
holds, at a future time \( T > n, s \) can never be the optimal last change-point prior to \( T \).

The proof can be found in Section 5 of supplementary materials in (Killick, Fearnhead, and Eckley 2012). This result states that if Equation (12) holds, then for any \( T > n \), the best segmentation with the LCP prior to \( T \) being at \( n \) will be better than any with the LCP at \( s \). Note that there exists a proper constant \( K \) satisfying Equation (11) for almost all cost functions used in practice. For example, if the cost function is the negative log-likelihood, then the constant can be selected as \( K = 0 \). Therefore, we have
\[ R(n+1) = \{ n \} \cup \{ \tau | \tau \in R(n), F(\tau) + \text{Cost} \left( Y_{\tau+1:n} \right) + K < F(n) \}. \] (13)

Our pruning process makes the optimization process very efficient under mild conditions with approximately linear computational cost with \( n \), or on average a constant computational cost at each time step, which is highly desirable for online change-point detection.

In our optimization problem (6), the cost function and the penalty term can be expressed as
\[
\text{Cost} \left( Y_{\tau_{\ell-1}+1:\tau_{\ell}} \right) = \sum_{i=1}^{p} \left[ \sum_{t=\tau_{\ell-1}+1}^{\tau_{\ell}} \left( Y_{it} - \sum_{j \neq i} \hat{\beta}_j^{(c)} Y_{jt} \right)^2 + \lambda_1 \sum_{j \neq i} | \hat{\beta}_j^{(c)} | \right].
\] (14)

where \( \hat{\beta}_j^{(c)} \) is the estimated coefficients via LASSO algorithm for the following optimization problem
\[
\min_{\hat{\beta}_j^{(c)} \neq 0} \left\{ \sum_{t=\tau_{\ell-1}+1}^{\tau_{\ell}} \frac{1}{2} \left( Y_{it} - \sum_{j \neq i} \hat{\beta}_j^{(c)} Y_{jt} \right)^2 + \lambda_1 \sum_{j \neq i} | \hat{\beta}_j^{(c)} | \right\}.
\] (15)

The detailed PELT based algorithm to solve our optimization problem (6) is shown in Algorithm 1. As mentioned before, under certain conditions, most importantly that the number of change-points is increasing linearly with \( N \), the computational efficiency of PELT is \( O(N) \). For calculating \( \text{Cost} (\cdot) \) in each time step of DSSL algorithm, there are \( p \) LASSO optimization problems and the computational efficiency of each problem is about \( O(p) \). Therefore, assuming that the requirements of PELT algorithm are also satisfied for DSSL algorithm, the computational efficiency of DSSL algorithm is about \( O(Np^2) \). In addition, it is worth noting that the computational efficiency is also related with the sparsity of the coefficients. The higher the sparsity of the model, the lower the computational complexity.

### 4.2. Parameter Selection

The selection of optimal penalty weights \( \lambda_{1,0} \) and \( \lambda_2 \) is non-trivial yet important for online change-points detection. In this section, an efficient tuning method combining Bayesian information criterion (BIC) and change-point detection accuracy are proposed for the PELT algorithm.
Figure 3. Basis functions of the two subspaces. (a)–(c): B-spline basis functions; (d)–(f): Fourier basis functions.

Figure 4. The original and fitted traces of six representative time series in one run (top: subspace I; bottom: subspace II). The black solid lines are the raw curves, and the red dashed lines are estimated values using the proposed method.

is negligible. Therefore, in practical applications, $K$ can be set to a larger constant in $(0, \lambda_2)$ with better pruning performance and little influence on the detection accuracy. In the case studies, we set $K$ to $K = \frac{2}{3} \lambda_2$ which yields very good performance.

**Theorem 4.** For the cost function determined by Equations (14) and (15),

1. $K = 0$ satisfies (11) for all $s < n < T$ and any time series $Y$;
2. $K$ cannot be larger than $\lambda_2$ unless all the time points are change-points.

5. Case Studies

In this section, the proposed algorithm is evaluated through numerical experiments with synthetic and real gesture data. All experiments are done using Python on an i7-6800K 3.40 GHz Intel processor with 16 GB RAM.

5.1. Simulation Study

In this section, we test our proposed algorithm on three simulation scenarios. The synthetic data $X$ are generated in two ways, one following the assumed subspace model described
in Section 2.2 (Case I, II) and the other following the GGM model (Case III). The variance of the noise is set to $\sigma^2 = 0.05^2$, $0.1^2$ and $0.2^2$ for each case. The experiment of each scenario is repeated 100 times. In all the scenarios, the two penalty parameters are tuned with three additionally simulated high-dimensional signals.

Case I. In this case, based on the subspace assumptions, two subspaces are simulated, where the first half of these time series are generated by linearly combining three B-spline basis functions, while the other half is generated by Fourier basis functions. The length of the time series is set to $N = 128$ and suppose there are two change-points at $\tau_1 = N/4$ and $\tau_2 = N/2$, respectively. The dimension is set to $p = 40$, that is, 20 time series for each subspace. These basis functions are shown in Figure 3. The coefficients $A$ are randomly generated in $[-0.5,0.5]$ for each segment.

Case II. In this case, we test our algorithm with higher dimension and more change-points. The dimension is set to $p = 400$. The length of the time series is set to $N = 320$, and suppose there are nine change-points at $\tau_i = i \times \frac{N}{10}$, $i = 1, \ldots, 9$

Figure 5. The estimated coefficients for the first three time series of each subspace ($Y_1$, $Y_2$, $Y_3$ and $Y_{21}$, $Y_{22}$, $Y_{23}$) as response variables: (a) subspace I and (b) subspace II.

Figure 6. Comparison of OP algorithm and PELT algorithm. (a) The selected candidates for the LCP at each time step and (b) average computation time and confidence interval per step.

Case III. In this case, the synthetic data are generated based on the GGM model where the covariance matrix or precision matrix changes. The data in each segment are generated from $N(0, \sigma^2 \Sigma^{(c)})$, $c = 1, 2$, where $\Sigma^{(c)} = \text{diag}(\Sigma_{11}^{(c)}, \Sigma_{22}^{(c)})$, $\Sigma_{11}^{(c)}$ and $\Sigma_{22}^{(c)}$ are $20 \times 20$ matrices randomly generated by Vine method (Lewandowski, Kurowicka, and Joe 2009) and $\sigma$ is set to 1, 2 and 4 in three cases. Other settings are the same as Case I.

First, we provide the detection results of DSSL in detail for Case I when the variance of the noise is set to $\sigma^2 = 0.05^2$. The two penalty coefficients are set to $\lambda_{1,0} = 0.0028$ and $\lambda_2 = 2.2$.

To avoid over-pruning when using $K = \frac{2}{\lambda_2}$, we intentionally replace $n$ with $n-5$, $n-10$, $n-15$ in (13) to get the candidate set of the LCP. Figure 4 shows the six representative time series in one run and the fitted ones using the proposed method. Clearly, due to sufficient samples in each subspace, the self-expression is very accurate in all three segments.

Figure 5 shows the estimated coefficients at the final time step for the first three time series of each subspace for illustration. For example, the rows of the first panel represents the coefficients of $Y_2, \ldots, Y_{40}$ when $Y_1$ is used as the response. These
Figure 7. The online detected latest change-point.

coefficients change at two change-points. Only the time series within the same subspace have nonzero coefficients, and each representation is sparse. These coefficients are constant within each segment.

Figure 6(a) shows the selected candidates for the LCP at each time step in one run. Only a very small number of points including the true LCP are selected as the candidates, which can effectively control the computational cost at each step. For instance, at 96 time step, the PELT algorithm only selects \{91, 86, 81, 69, 68, 67, 66, 65, 64, 63, 62, 61\} as LCP candidates. Figure 6(b) shows the average computation time and confidence interval per step, where the PELT algorithm can save more than 90% of the computation time than the OP algorithm.

Figure 7 shows the mean and confidence interval of the detected LCP at each time step. It can be clearly seen that in most of the cases, the change can be timely detected with only about 10 time steps after the change occurs. Besides, as the observations since the latest change accumulates, the detected location of the LCP becomes more and more accurate. Note that in each replication, the detected LCP often abruptly jumps to the true values shortly after the change occurs. As the detection delay may vary from run to run, the change of the mean detection is not as abrupt as a single run.

In order to further demonstrate the superiority and effectiveness of our model, we compare our DSSL model with DFSL model (Zhang et al. 2020), VCVS model (Kolar and Xing 2012) and GFGL model (Gibberd and Nelson 2017), which are state-of-the-art offline methods. Three offline and two online metrics are used for performance evaluation and comparison. The offline metrics include the precision, recall, and running time. The precision and recall are proposed by Peel and Clauset (2015) to evaluate offline multiple change-point detection algorithms and they are defined as follows:

\[
\text{Precision} = \frac{1}{n_e} \sum_{i} n_r \delta \left( \inf_{j} \left| \hat{\tau}^{(i)} - \tau^{(i)} \right| \leq s \right),
\]

\[
\text{Recall} = \frac{1}{n_r} \sum_{j} n_e \delta \left( \inf_{i} \left| \hat{\tau}^{(i)} - \tau^{(i)} \right| \leq s \right),
\]

where \(n_r\) and \(n_e\) are the estimated number of change-points and the real number of change-points, respectively, \(\delta\) is the indicator function, and \(s\) is the detection error bound. The precision and recall are then the proportion of estimated change-points that timely detect true change-points. Similarly, recall is the proportion of true change-points that are timely detected. Here we set \(s = 5\) time steps in the comparison. The two online metrics, that is, detection delay and running time per step, are used to evaluate the detection timeliness in online applications.

All experiments are replicated 100 times, and the results are shown in Table 1. For precision and recall metrics, the proposed DSSL is superior to other approaches in Case I and Case II. While in Case III, other approaches perform better than DSSL approach. The results show that our DSSL performs perfectly if the data satisfies the SSC assumptions. However, it is inferior to the GGM related methods when only the covariance matrix or precision matrix changes. Since the data are generated based on a different pattern, the BIC criterion may not be the optimal choice in such scenarios. DSSL performs better in Case II than Case I because the change is relatively more significant.

In terms of the running time, GFGL approach is the fastest algorithm and our DSSL approach is slightly slower in Case I and II. However, GFGL is an offline method, and if it is applied online, the whole optimization process has to be repeated at each time step, leading to a much higher overall cost. Note that, in Case II, DFSL approach and VCVS approach are unable to detect change-points due to either insufficient memory or extremely long computational time. In Case III, the DSSL approach has a longer running time because the optimization for LASSO problem is much more time-consuming when data are generated by the GGM model. The maximum number of iterations for solving LASSO is set to 1000 in Case I and II while it is set to 10,000 in Case III. The running time can be reduced to one tenth by setting the maximum number of iterations to 1000, at a cost of 10% decreasing for precision and recall metrics. As for the detection delay, we only show the results for DSSL, as all other methods are offline. Similarly, DSSL approach performs better in Case II than Case I.

In addition, as the variance of noise increases, the change-points are more difficult to detect with lower precision, lower recall and larger detection delay generally. However, the performance of the proposed DSSL approach does not change a lot in Case III. The reason is that \(\sigma^2\) in Case III only affects the variation of the data. The correlations among the time series remain the same.

5.2. Human Motion Tracking

In this section, we apply the proposed DSSL to the MSRC-12 Gesture Dataset for gesture tracking (Fothergill et al. 2012). This dataset consists of sequences of human skeletal body part movements and the associated gesture that needs to be recognized by the system. Each sample of the sequences contains 60 variables, which are the three dimensional coordinates of 20 human joints. The body pose is captured at a sample rate of 30 Hz with \(\pm 2\) cm accuracy in joint positions. In the MSRC-12 Gesture Dataset, there are 30 subjects and they perform 12 gestures each for ten times. The position of these 20 joints and some snapshots of shoot gesture and throw gesture are shown in Figure 8.

Motion segmentation is often a very critical step for gesture recognition. Here we combine the sequences of two gestures of the same subject together to demonstrate the effectiveness of the
proposed method. Specifically, we choose “shoot” and “throw” as the two gestures. In the first segment, the subject stretches his arms out in front of him, holds a pistol in both hands, makes a recoil movement and then returns to the original position. In the second segment, the subject uses his right arm to make an overarm throwing movement and then returns to the original position. For the sake of simplicity, we convert coordinates of overarm throwing movement and then returns to the original position. In

<table>
<thead>
<tr>
<th>Case</th>
<th>Sigma</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>Running time (s)</th>
<th>Detection delay (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.05</td>
<td>DSSL 100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>10.17</td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td>DFSL 54.60</td>
<td>56.72</td>
<td>61.24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>VCVS 47.87</td>
<td>45.50</td>
<td>44.85</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 34.76</td>
<td>24.19</td>
<td>23.80</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 100.00</td>
<td>100.00</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 81.00</td>
<td>86.00</td>
<td>86.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 66.00</td>
<td>65.00</td>
<td>74.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 87.50</td>
<td>99.00</td>
<td>99.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 35.29</td>
<td>28.93</td>
<td>31.66</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 199.04</td>
<td>192.93</td>
<td>185.86</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 98.22</td>
<td>91.94</td>
<td>83.41</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 1.54</td>
<td>3.70</td>
<td>7.35</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 0.28</td>
<td>0.23</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 10.17</td>
<td>12.64</td>
<td>15.57</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 199.04</td>
<td>192.93</td>
<td>185.86</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 98.22</td>
<td>91.94</td>
<td>83.41</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 35.29</td>
<td>28.93</td>
<td>31.66</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 199.04</td>
<td>192.93</td>
<td>185.86</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 98.22</td>
<td>91.94</td>
<td>83.41</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 1.54</td>
<td>3.70</td>
<td>7.35</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 0.28</td>
<td>0.23</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 10.17</td>
<td>12.64</td>
<td>15.57</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 199.04</td>
<td>192.93</td>
<td>185.86</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 98.22</td>
<td>91.94</td>
<td>83.41</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 35.29</td>
<td>28.93</td>
<td>31.66</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 199.04</td>
<td>192.93</td>
<td>185.86</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 98.22</td>
<td>91.94</td>
<td>83.41</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 1.54</td>
<td>3.70</td>
<td>7.35</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DSSL 0.28</td>
<td>0.23</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DFSL 10.17</td>
<td>12.64</td>
<td>15.57</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>VCVS 199.04</td>
<td>192.93</td>
<td>185.86</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>GFGL 98.22</td>
<td>91.94</td>
<td>83.41</td>
<td></td>
</tr>
</tbody>
</table>

Figure 8. (a) The position of the 20 joints; (b) five snapshots of the shoot gesture; (c) five snapshots of the throw gesture.

device than the reference point. With this regard, we can infer that these joints lie in some multiple low-dimensional subspaces and the subspace may also be detected, though not often.

6. Discussion and Conclusions

In this article, we proposed a DSSL approach for online structural change-point detection of high-dimensional streaming data. Specifically, it is assumed that the high-dimensional data lie in multiple low-dimensional subspaces and the subspace are generally the same with the data of the left and right hand. So in total $p = 18$ variables are considered. We only use one historical streaming data to tune the two penalty parameters.

Figure 10 shows the original and fitted variables and the sequentially detected LCPs. The change of the gesture is timely detected only after about 10 time steps. The positions of the joints of head, trunk, legs and feet do not change much in the whole motion sequence. The joints of left arm and right arm play a key role to detect the change-point. The subspaces for each gesture are further identified via the clustering of the coefficient matrix, which are shown in Figure 11. This result is consistent with the actual movement of each joint in these two gestures.

The detection performance of the DSSL approach and other baselines is compared in Table 2. The detection error bound $s$ is set to 10 in the calculation of precision ratio and recall ratio. Similar to the simulation study, the proposed DSSL approach outperforms other benchmarks in the detection accuracy with less running time per step. For all methods, the recall is high while the precision is relatively low, indicating that these two methods sometimes detect other change-points besides the one connecting the two gestures. This is understandable because there may exist some changes in one single gesture. For example, for gesture “throw,” the motion can be further divided into three segments due to slight correlation change, that is, raising a hand, throwing something and putting down this hand. These changes may also be detected, though not often.

6. Discussion and Conclusions

In this article, we proposed a DSSL approach for online structural change-point detection of high-dimensional streaming data. Specifically, it is assumed that the high-dimensional data lie in multiple low-dimensional subspaces and the subspace
structures may abruptly change over time. Only the variables from the same subspace correlate with each other and each variable can be sparsely represented by others.

Based on the self-expressive property, we proposed a novel multiple structural change-point model with two penalty terms in the loss function to encourage sparse representation and avoid excessive change-points. The asymptotic consistency of the estimators was further established. A PELT based algorithm was proposed for online optimization and change-point detection. Based on some historical data, the penalty coefficients in our model could be properly selected by a tuning method combining a BIC criterion and detection accuracy. The effectiveness of the proposed method was demonstrated on synthetic data and gesture data for motion tracking.

There are several issues that are worthy of further investigation. First, the proposed method assumes that the measurement noises of all variables are independent and identically distributed. However, the variance heterogeneity, cross-correlation and even auto-correlation may exist in practice. In addition, all the subspaces are assumed to be linear manifolds in the current work. To make it more flexible, nonlinear manifolds can be considered by using nonlinear regressions, such as kernel based methods. Last but not least, when there are not sufficient samples in the subspace, the self-expressive assumption
may not hold. How to learn the subspace sequentially with insufficient samples is a challenging problem that needs to be solved.

Supplementary Materials

Additional proofs: Detailed proofs are included as a pdf file. Python code: The zip file contains Python codes of all case studies.

Acknowledgments

The authors would like to thank the editor, associate editor, and anonymous reviewers for many constructive comments which greatly improved the article.

Funding

This work is supported in part by the NSFC grant NSFC-72171003, NSFC-71932006 and NSFC-51875003.

References


Dhulekar, N., Nambirajan, S., Oztan, B., and Yener, B. (2015), “Seizure Prediction by Graph Mining, Transfer Learning, and Transformation Learning,” in International Workshop on Machine Learning and Data Mining in Pattern Recognition (pp. 32–52), Springer. [1]


