MODELING AND DETECTION USING HIGH-DIMENSIONAL TIME SERIES DATA

A Dissertation
Presented to
The Academic Faculty

By

Tingnan Gong

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Engineering
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

May 2025

MODELING AND DETECTION USING HIGH-DIMENSIONAL TIME SERIES DATA

Thesis committee:

Dr. Yao Xie, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Seong-Hee Kim, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Kamran Paynabar
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Nagi Gebraeel
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Simon Mak
Department of Statistical Science
Duke University

Date approved: May 1, 2025

Not desiring the rarity of gems. Or the manyness of grains of sand.

Lao Zi, Dao De Jing

To my family.

ACKNOWLEDGMENTS

I would like to express my deep gratitude to my advisor, Dr. Yao Xie, for her belief in me and her invaluable guidance throughout my Ph.D. journey. I am also profoundly thankful to my co-advisor, Dr. Seong-Hee Kim, for her unwavering support and insightful mentorship. Their trust, encouragement, and dedication have been instrumental to my progress, and none of my achievements would have been possible without their understanding and help.

I am sincerely grateful to my committee members—Dr. Kamran Paynabar, Dr. Nagi Gebraeel, and Dr. Simon Mak—for their thoughtful feedback and constructive suggestions, which have greatly enriched the quality of my dissertation. I also thank Dr. Alon Kipnis, Dr. Xiuyuan Cheng, and Dr. Tuo Zhao for their mentorship, whether through collaborations or as instructors.

My heartfelt thanks go to all my colleagues at Georgia Tech, including Dr. Shixiang Zhu, Dr. Liyan Xie, Dr. Minshuo Chen, Dr. Minghe Zhang, Dr. Henry Yuchi, Dr. Song Wei, Dr. Qunzhi Xu, Dr. Yuyang Shi, Dr. Zheng Dong, Xiaoyue Liu, Dr. Xin Jing, Dr. Chen Xu, Dr. Xiangyu Mao, Dr. Di Liu, Dr. Qingru Zhang, Zexian Zeng, Yuhao Wang, Che-Yi Liao, Kai Tan, Junghwan Lee, Hanyang Jiang, and many others, for the valuable time we spent working and growing together. I deeply appreciate their encouragement and support throughout my studies.

I am also thankful to the H. Milton Stewart School of Industrial and Systems Engineering at the Georgia Institute of Technology. The school and its community have provided a nurturing and intellectually stimulating environment that enabled me to grow both academically and professionally. I gratefully acknowledge the financial support received through various fellowships and travel grants, which empowered me to explore my research with confidence and independence.

My sincere thanks also go to my alma mater, the University of Science and Technology of China (USTC), and to the professors and instructors who taught and guided me there, especially Jingzhi Xu, Dr. Yu Chen, and Dr. Weiping Zhang. Their dedication laid a strong foundation for my academic development.

Last but not least, I wish to thank my parents for their unconditional love, encouragement, and patience. Their greatness and grace toward me are beyond words. I am also deeply grateful to my fiancée, Ms. Wu, a strong and steadfast companion who stood by me through all challenges. I could not have reached this point without their support. To them, I dedicate this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Change-point detection and point process modeling play a vital role in time series analysis, particularly when the data are high-dimensional, correlated, or observed under time uncertainty. For high-dimensional data with complex spatio-temporal correlations and minimal distributional assumptions, real-time detection of distributional shifts requires carefully designed procedures that address multiple statistical and computational challenges. For event data with time uncertainty, to the best of our knowledge, no existing work has established a point process model that explicitly incorporates time uncertainty. This thesis develops new methods for online change-point detection in complex high-dimensional data and for point process modeling under time uncertainty. We propose a distribution-free CUSUM procedure for low-rank image data, which avoids parametric noise assumptions and demonstrates reliable detection performance in manufacturing applications. We also develop a neural network-based detection procedure via a binary classification proxy to flexibly adapt to various types of distributional shifts. Additionally, a higher-criticism-based approach is developed for detecting sparse and weak changes in high-dimensional signals, achieving asymptotically optimal detection delay under rare moderate departure regimes with theoretical guarantees. Finally, a novel framework is introduced for modeling point processes with time uncertainty; through a carefully designed training scheme and kernel-based parameterization, the model predicts event occurrence probabilities and recovers dynamic causal structures in both simulations and real-world datasets such as Sepsis medical records and urban burglary incidents.

CHAPTER 1
INTRODUCTION

Change-point detection and point process modeling play a vital role in time series analysis, particularly when the data are high-dimensional, correlated, or observed under time uncertainty. From industrial process monitoring to healthcare diagnostics, timely detection of changes and accurate predictions of arriving events can be vital. As data becomes increasingly high-dimensional, structured, and temporally complex, traditional parametric methods often struggle to maintain reliability, scalability, and interpretability across modern applications.

This dissertation investigates a range of methodological innovations in change-point detection and point process modeling, addressing several open challenges: distributional robustness, high-dimensional scalability, sparse signal recovery, and uncertainty in event timing.

We begin with the development of a distribution-free CUSUM procedure tailored for low-rank image time series data. By exploiting the spatial-temporal structure of matrix-valued observations and avoiding strong parametric assumptions, this method enables robust online detection of mean shifts while accounting for both spatial and temporal dependencies. Theoretical insights into the statistical properties of the detection statistic are provided, alongside empirical validation using simulated and real data.

Building on the limitations of classical CUSUM methods in high-dimensional and model-mismatched scenarios, we next introduce a neural-network-based CUSUM (NN-CUSUM) framework. This approach combines the recursive computational efficiency of CUSUM with the expressive power of neural networks. This work underscores how deep learning can be meaningfully integrated with classic statistical procedures to handle large-scale, nonparametric change detection problems. The method is justified through extensive experiments.

Recognizing that real-world changes may affect only a sparse subset of variables, we then turn to the problem of sparse multi-stream quickest detection. Leveraging the higher criticism (HC) statistics, we propose an aggregation framework that performs local testing on individual data streams and then pools these results to achieve sensitivity to weak and sparse signals. The method is shown to achieve optimal detection delay in the information-theoretic sense and can empirically present small detection delays compared to other methods.

Finally, we turn to point process data with time uncertain in the events, a scenario common in applications like medical event logs and social interactions. We propose a novel framework that models such data on both continuous time and discrete-time grids, incorporating uncertainty via a fundamental likelihood model. Our method accommodates both stationary and non-stationary dynamics, and supports parameter recovery using first-order optimization techniques. Applied to a real-world sepsis dataset, this framework uncovers latent causal patterns that are obscured by noisy or imprecise event timing.

Together, these contributions advance the theory and practice of change-point detection across distribution-free matrices or streams, and of point process modeling on events with time uncertainty, offering a unifying theme of scalable, structure-aware, and robust statistical methodology for modern high-dimensional time series data.

CHAPTER 2

DISTRIBUTION-FREE ONLINE CHANGE DETECTION FOR LOW-RANK IMAGES

2.1 Overview

In modern manufacturing, rapid improvements in sensor technology allow the industry to acquire data with dimensions much higher than those of decades ago. For example, images of layers in 3D printing [1] can be obtained approximately every five seconds. For monitoring purposes, the image data can be viewed as matrix data. However, online monitoring of matrix data faces multifaceted challenges:

- *Temporal independence:* Due to the high sampling rate, weak autocorrelation among consecutive images is nearly inevitable. Long-lasting autocorrelation is also common in specific industrial problems. For example, in the case of in-situ detection of laser power bed fusion (LPBF) [2], an anomaly called a hot spot may occur. When the laser beam repeatedly irradiates a thermally insulated region, heat builds up exceptionally quickly, leading to the formation of a hot spot. In the view of a thermographic camera, the pixels in the center of the overheating area remain hot (high intensity) while the edges slowly cool down, causing durable auto-correlation and nonstationary spatial correlation.

- *Spatial independence:* Individual components of a high-resolution matrix tend to exhibit weak spatial correlation, thus violating the assumption of spatial independence. As mentioned in the aforementioned in-situ detection problem of LPBF, the hot spot anomaly can lead to unusual spatial correlation.

- *Data normality:* Violations of normality in data can result in a deviation from theoretical assumptions made by methods with the normality assumption, as discussed in [3]. [4] tackle the normality issue by transforming low-dimensional vectors into nearly normally distributed data. However, this technique turns out to be computationally prohibitive in high dimensions.

- *High dimensionality:* Matrix data with large dimensions can drastically increase the complexity of practical algorithm. Additionally, rank deficiency is common which may cause performance degradation if not properly handled, which calls for algorithms that are designed for low-rank matrices. Many techniques, combined with dimensionality reduction techniques such as the principal components analysis (PCA) [5] and process projection and fusion [6] may improve computational and statistical efficiency of the methods.

One way to handle high-dimensional matrix data is to adopt the profile monitoring perspective, as in [7], [8] and [9]. Procedures that employ this approach make a compromise by treating high-dimensional matrix data as a long vector of predictors, sacrificing the spatial data structure. Practitioners construct the response in single or multiple channels and develop appropriate models to characterize the functional relationship between the response and the predictors. The parametric regression model serves as one of the primary instruments in profile monitoring to describe this functional relationship. [10], [11], [12] and [13] perform profile monitoring using linear regression models. [14], [15], and [16] introduce nonlinear profile monitoring to enhance interpretability. Other profile monitoring works utilize dimension reduction techniques to extract features, such as PCA and independent component analysis (ICA). [17] discuss comparisons between regression-based and PCA-based profile monitoring procedures.

Another approach is to use wavelet-based distribution-free profile monitoring procedures [18, 19] for high-dimensional vectors. These methods can partially address the mentioned challenges, including general marginal distributions and high dimensionality. However, profile monitoring tends to alter the original matrix structure and results in a loss of spatial information. Additionally, none of the aforementioned works incorporates general temporal correlation.

To mitigate the loss of information caused by data structure transformation, some works exploit matrix characteristics directly and construct matrix-based monitoring procedures. [20] categorize various matrix-based monitoring procedures, including spatial and multivariate-matrix-analysis-based control charts.

Recent works either combine multiple popular techniques or develop new models on matrix data from images. Among the procedures that integrate multiple popular techniques, [21] employ low-rank tensor decomposition to achieve dimension reduction and feature extraction and monitor the extracted features using Hotelling T^2 and Q control charts; the method does not strictly require data normality and independence. However, their Phase-I calibration of the control limits is time-consuming due to the estimation of the empirical distribution of statistics. [22] and [23] utilize wavelet transformations on matrix data to extract features and then build control charts. [22] perform a generalized likelihood ratio (GLR) control chart on extracted features. [23] further extend the approach in [22] by combining it with a regression-based parametric model to accommodate underlying data correlation. However, they still require data normality. [24] compress data using partial least squares discriminant analysis and then construct the control statistics using Delaunay triangulation [25] to segment the squared error matrix into triangles, computing the area within each triangle as control statistics. Despite the novelty of this technique, the assumption of data independence limits its broad application. Region of interest (ROI) is a popular data compression technique for matrix processing. Based on this tool, [26] incorporate the GLR control chart to monitor the average intensity vector calculated from these regions, assuming the presence of at most one cluster of defects in the images. [27] extend this approach further to detect multiple clusters of defects in images. [28] consider the combination of ROIs and one-way analysis of variance (ANOVA).

On the other hand, some procedures develop new models on the matrices. For a series of images with a smooth background, [29] propose a smooth-sparse decomposition (SSD) model to decompose observations into the background and potential sparse anomalies. However, the SSD model cannot incorporate temporal information into the data. Building on this work, [30] extend the SSD methodology with a spatio-temporal smooth-sparse decomposition (ST-SSD) model to tackle temporal information and spatial patterns.

Although initially not developed for matrix monitoring, some spatio-temporal monitoring charts [31, 32, 33] used in the environment and public health surveillance can be adapted for matrix monitoring. Another thread of research addresses specific matrix monitoring problems in the industry. One of the applications that has garnered significant attention is matrix monitoring in metal additive manufacturing [34, 35, 36, 37, 38].

In this chapter, we propose the distribution-free low-rank image monitoring (DFLIM) procedure to perform online change detection for a time series of matrices. This work extends our earlier work [39], which considers a special case when the in-control mean image matrix is rank-one; here, we consider the general case where in-control mean image matrix can have a rank higher than one but still low rank. Our procedure's monitoring statistics are constructed based on singular value decomposition (SVD) of the in-control mean matrix and projected observations. These statistics are then used in the CUSUM recursion, which can be computed recursively online. We analyze the theoretical properties of the DFLIM procedure in terms of the in-control average run length (ARL_0) and out-of-control average run length (ARL_1). These metrics evaluate how frequently a false alarm occurs when a monitored process is in control and how quickly a change can be detected when the process is out of control. Furthermore, we empirically study the capability of the procedure against temporal dependence and non-normality of the data. The effectiveness of the proposed procedure is demonstrated through simulated and real data experiments.

The remainder of the paper is organized as follows: Section 2.2 describes our problem and assumptions. Section 2.3 proposes the DFLIM procedure, which utilizes a CUSUM chart on T^2 statistics extracted from the high-dimensional matrix data. In Section 2.4, we conduct theoretical analysis on the mean shift size in our statistics and study the ARL behavior of the DFLIM procedure. In Section 2.5, we design simulated experiments designed to demonstrate the performance of the DFLIM procedure and empirically support some claims that are challenging to prove analytically. Section 2.6 applies the DFLIM procedure to real data sets to demonstrate its broad applications, followed by concluding remarks in Section 2.7.

2.2 Problem Setup

At each time t, we observe a matrix denoted as $X_t \in \mathbb{R}^{p_1 \times p_2}$. At an unknown change point $k \in \{1, \ldots, \infty\}$, the mean of X_t shifts from M_0 to M_1. We can formulate the problem in terms of online hypothesis testing:

$$
\begin{aligned}
&\mathrm{H}_0 : X_t = M_0 + \epsilon_t, \quad t = 1, 2, \ldots \\
&\mathrm{H}_1 : X_t = \begin{cases} M_0 + \epsilon_t, & t = 1, 2, \ldots, k-1, \\ M_1 + \epsilon_t, & t = k, k+1, \ldots. \end{cases}
\end{aligned}
\tag{2.1}
$$

where M_0 and M_1 are p_1-by-p_2 matrices representing the in-control and out-of-control mean matrices, respectively. We assume that $p_1 \leq p_2$, which can always be achieved by transposing the image matrices if necessary. The noise matrix $\epsilon_t \in \mathbb{R}^{p_1 \times p_2}$ is assumed to have the same marginal distribution across time t, whereas the parametric form of the distribution is unrestricted, allowing for spatial correlation within ϵ_t for each t and temporal correlations across ϵ_t across time. We denote expectations of the observations before and after the change as $\mathbb{E}_0[\cdot]$ and $\mathbb{E}_1[\cdot]$, respectively; The covariance functions $\mathrm{Cov}_0(\cdot)$ and $\mathrm{Cov}_1(\cdot)$ are defined similarly. Note that in (2.1), all the randomness are due to ϵ_t; however, the observations X_t have different mean before and after the change.

Before the change, X_t follows marginal distribution F_0 with mean

$$\mathbb{E}_0[X_t] = M_0,$$

where we assume M_0 to be known; this assumption is a reasonable simplification, since usually have in-control data to estimate the pre-change parameters accurately. For example, in a printed circuit board (PCB) manufacturing process, the in-control pattern is designed before production, or sufficient in-control data is available to accurately estimate the process parameters (further discussion can be found in Section 2.3.2).

After the change (i.e., $t \geq k$), the observations X_t experience a mean shift

$$\mathbb{E}_1[X_t] = M_1 := M_0 + A,$$

where $A \in \mathbb{R}^{p_1 \times p_2}$ represents the unknown non-zero and deterministic mean shift. Throughout the paper, we assume that the following assumptions hold:

Assumption 2.1. *The marginal distributions of X_t for $t = 1, 2, \ldots, k-1$ and $X_t - A$ for $t = k, k+1, \ldots$ are F_0; i.e., the marginal distribution of ϵ_t stays the same for all t.*

Assumption 2.2. *The in-control mean M_0 is low-rank, with rank $r \ll \min\{p_1, p_2\}$, and its SVD is given as follows:*

$$M_0 = \sum_{i=1}^{r} \lambda_i u_i v_i^\top$$

where λ_i, u_i, and v_i represent the singular values, left singular vectors, and right singular vectors, respectively.

2.3 Distribution-Free Low-Rank Image Monitoring

Given the matrix data X_t, we extract two types of projections for detection statistics. First, using the left and right singular vectors, u_i and v_i, of the in-control mean M_0, we compute the *first-type* projected observations $\beta_{i,t}$ for $i = 1, \ldots, r$ as follows:

$$\beta_{i,t} = u_i^\top X_t v_i. \tag{2.2}$$

Next, we calculate the residual matrix

$$R_t = X_t - M_0,$$

and perform its SVD to obtain its singular values in descending order. We use the first r singular values as the *second-type* projected observations $\gamma_{i,t}$ for $i = 1, \dots, r$.

The statistic consisting of both types of projected observations at time t is denoted as y_t:

$$y_t = \left[\beta_t^\top, \gamma_t^\top\right]^\top = \left[\beta_{1,t}, \dots, \beta_{r,t}, \gamma_{1,t}, \dots, \gamma_{r,t}\right]^\top \in \mathbb{R}^{2r}, \tag{2.3}$$

where $\beta_t = \left[\beta_{1,t}, \dots, \beta_{r,t}\right]^\top$ and $\gamma_t = \left[\gamma_{1,t}, \dots, \gamma_{r,t}\right]^\top$. Note that $\beta_{i,t}$ is the projection of X_t onto the static directions u_i and v_i, whereas $\gamma_{i,t}$ represents the projection onto temporally varying directions, i.e., the singular vectors of R_t. We discuss why it is necessary to incorporate both types of projections into y_t to ensure detection power in Section 2.4.

To construct a CUSUM procedure, we compute a T^2-type statistics T_t using y_t:

$$T_t = \left(y_t - \mathbb{E}_0[y_t]\right)^\top \mathrm{Cov}_0^{-1}(y_t) \left(y_t - \mathbb{E}_0[y_t]\right), \tag{2.4}$$

which will be used in the increment in the CUSUM statistics. Note that $\mathrm{Cov}_0(y_t)$ is short-handed for $\mathrm{Cov}_0(y_t, y_t)$, the covariance matrix of the vector y_t.

The CUSUM statistics for the DFLIM procedure are defined recursively with $S_0 = 0$:

$$S_t = \max\left\{0, S_{t-1} + \left(T_t - \mathbb{E}_0[T_t] - c\sigma_T\right)\right\}, \quad t = 1, \dots, \tag{2.5}$$

where c is a pre-selected constant, and $\sigma_T > 0$ is the in-control marginal standard deviation of T_t (under the marginal distribution F_0). The DFLIM procedure stops and raises an out-of-control signal at time τ when the monitoring statistic exceeds a control limit $H > 0$:

$$\tau = \inf\{t > 0 : S_t \geq H\}. \tag{2.6}$$

Given the stopping time τ, we define $\mathrm{ARL}_0 = \mathbb{E}_0[\tau]$ and $\mathrm{ARL}_1 = \mathbb{E}_1[\tau]$. Here, ARL_0 represents the average time to raise a false alarm when the process is in-control. ARL_1 measures how quickly a monitoring procedure can detect a change when the process is out-of-control.

The detailed description of the proposed DFLIM procedure is given in Algorithm 1. Note that in the inputs of Algorithm 1, the in-control parameters, including M_0, $\mathbb{E}_0[y_t]$, and $\mathrm{Cov}_0(y_t)$, are assumed to be known for theoretical analysis. In practice, the in-control parameters are estimated from in-control data, as described in Section 2.3.2 and Algorithm 2.

Algorithm 1: Distribution-free Low-rank Image Monitoring

Input : Sequence of observations $\{X_t, t = 1, 2, \ldots\}$, in-control mean matrix M_0, in-control mean vector $\mathbb{E}_0[y_t]$, in-control covariance $\mathrm{Cov}_0(y_t)$, constant c, standard deviation σ_T, and control limit H (whose determination is discussed in Section 2.3.1).

Output: stopping time τ.

Initialize $\tau = +\infty$, $t = 0$, and $S_0 = 0$.

Perform SVD on M_0 to obtain directions u_i and v_i for projections with $i = 1, \ldots, r$.

while $S_t < H$ **do**

    Set $t = t + 1$ and obtain X_t.

    Compute $\beta_{i,t}$ for $i = 1, \ldots, r$ as in (2.2).

    Obtain r largest singular values of $R_t = X_t - M_0$, i.e. $\gamma_{i,t}$ for $i = 1, \ldots, r$.

    Form y_t as in (2.3) and compute T_t as in (2.4).

    Update the monitoring statistic S_t as in (2.5).

end

Set $\tau = t$ and raise an out-of-control alarm.

2.3.1 Control limit determination and setup phase

In this section, we derive an expression for ARLs of the DFLIM procedure and explain how this expression can be used to determine the control limit H. Additionally, we explain how to estimate the parameters necessary for implementing the DFLIM procedure.

Define the mean and variance parameters of T_t as follows:

$$m := \mathbb{E}[T_t], \quad \Omega^2 := \lim_{t\to\infty} t\mathrm{Var}\left(\frac{\sum_{\ell=1}^{t} T_\ell}{t}\right) = \sum_{t=-\infty}^{\infty} \mathrm{Cov}(T_0, T_t),$$

where the expectation and the variance can be taken under either in-control or out-of-control phase. Here, Ω^2 represents the limiting variance parameter for T_t, which provides a better measure of process variability compared to marginal variance in the presence of temporal correlations. We can then define the standardized time-series of the first t observations $\{T_1, T_2, \ldots, T_t\}$ as follows:

$$\mathcal{C}_t(s) \equiv \frac{\sum_{\ell=1}^{\lfloor ts \rfloor} T_\ell - tsm}{\sqrt{t\Omega^2}}, \quad s \in [0,1].$$

In addition to Assumptions 2.1 and 2.2, we assume that $\{T_t : t = 1, 2, \ldots\}$ satisfies the Functional Central Limit Theorem (FCLT):

Assumption 2.3 (FCLT). *Given $\{T_t : t = 1, 2, \ldots\}$, the standardized time-series process $\mathcal{C}_t(\cdot)$ satisfies:*

$$\mathcal{C}_t(\cdot) \xrightarrow{\mathcal{D}} \mathcal{W}(\cdot) \quad \textit{as} \quad t \to \infty,$$

in the space $D[0,1]$ where $\xrightarrow{\mathcal{D}}$ denotes convergence in distribution, $\mathcal{W}(\cdot)$ denotes a standard Brownian process, and the space $D[0,1]$ contains functions defined on $[0,1]$ that are right-continuous with left-hand limits.

For conditions under which the FCLT is applicable, one can refer to [40]. In Chapter 4.4 of [41], it is suggested that, in practical terms, it is generally justifiable to presume the validity of the FCLT when Ω^2 is finite.

Under the FCLT, [42] show that the limiting process of CUSUM statistics is closely related to a standard Brownian motion process, regardless of the parametric form of X_t, forming the basis for a distribution-free procedure. Furthermore, they derive an approximate expression of ARLs, including both ARL_0 and ARL_1, based on the properties of the converged process of CUSUM statistics. The DFLIM procedure also achieves the distribution-free property under the FCLT and determines the control limit H using the approximate expression of ARLs of the converged process, as provided in Lemma 2.3.1. Similar results were reported in [43], however without a clear characterization of the process variability using Ω^2.

Lemma 2.3.1 ([42]). *If $\{T_t : t = 1, 2, \ldots\}$ satisfies Assumption 2.3, then*

$$\mathrm{ARL} \approx \begin{cases} H^2/\Omega^2, & \textit{if } d_T = 0, \\ \frac{\Omega^2}{2d_T^2}\left[\exp\left(-\frac{2Hd_T}{\Omega^2}\right) - 1 + \frac{2Hd_T}{\Omega^2}\right], & \textit{otherwise}, \end{cases}$$

for large H, where $d_T = \mathbb{E}[T_t] - \mathbb{E}_0[T_t] - c\sigma_T$.

When the monitored process is in-control, we have $d_T = -c\sigma_T$. By plugging $d_T = -c\sigma_T$ in the ARL expression in Lemma 2.3.1 and setting it equal to a target ARL_0, one can analytically solve for the value of the control limit H. As suggested in [44], a more accurate control limit H can be obtained by solving the following equation:

$$\mathrm{ARL}_0 = \frac{\Omega_0^2}{2c\sigma_T^2}\left\{\exp\left[\frac{2c\sigma_T(H + 1.166\Omega_0)}{\Omega_0^2}\right] - 1 - \frac{2c\sigma_T(H + 1.166\Omega_0)}{\Omega_0^2}\right\}, \qquad (2.7)$$

where Ω_0^2 is the in-control Ω^2. More details on the calculation of Ω_0^2 are in Appendix A.1. We summarize the calculation of parameters and the control limit H, along with other parameters required for the implementation of the DFLIM procedure, in Algorithm 2.

Algorithm 2: Setup phase for the DFLIM procedure

Input : Sequence of in-control observations $\{X_t, t = 1, 2, \dots, n\}$, in-control mean matrix M_0, rank r, constant c, and target ARL_0.

Output: $\mathbb{E}_0[y_t]$, $\mathrm{Cov}_0(y_t)$, σ_T, and H.

Perform SVD on M_0 to obtain directions u_i and v_i for projections with $i = 1, \dots, r$. **while** $t < n$ **do**

  Set $t = t + 1$ and obtain X_t.

  Compute $\beta_{i,t}$ for $i = 1, \dots, r$ as in (2.2).

  Obtain r largest singular values of $R_t = X_t - M_0$, i.e. $\gamma_{i,t}$ for $i = 1, \dots, r$.

  Form y_t as in (2.3).

end

Approximate $\mathbb{E}_0[y_t]$ by $\bar{y} = 1/n \sum_{t=1}^{n} y_t$ and $\mathrm{Cov}_0(y_t)$ by $1/(n-1) \sum_{t=1}^{n} (y_t - \bar{y})(y_t - \bar{y})^\top$.

Compute T_t as in (2.4) for $t = 1, \dots, n$.

Compute $\bar{T} = 1/n \sum_{t=1}^{n} T_t$.

Approximate σ_T by $\sqrt{1/(n-1) \sum_{t=1}^{n} (\bar{T} - T_t)^2}$.

Estimate variance parameter Ω_0^2 of T_t using Cramér-von Mises (CvM) estimator.

Calculate control limit H by solving equation (2.7).

2.3.2 Additional parameter calibration

In this subsection, we discuss how to estimate M_0 when the target mean image is not well-defined and how to determine r from the estimated M_0. Additionally, we provide a recommendation for choosing a constant c.

Estimation of M_0. With training in-control data $X_1, \dots, X_n$, we use an estimator for M_0 being the sample average of in-control observations, namely $\widehat{M_0} = \sum_{t=1}^{n} X_t/n$. Due to the Law of Large Numbers, $\widehat{M_0}$ approximates the true in-control mean matrix M_0 well when n is large.

Selection of r. The rank r of M_0 is obtained numerically using singular value hard thresholding [45]. Specifically, for the singular values of M_0, denoted as $\lambda_1 \geq \ldots \geq \lambda_{\min\{p_1,p_2\}}$, and given a threshold $\lambda > 0$, we find $r = \max\{i : \lambda_i \geq \lambda\}$. Alternatively, this threshold can be replaced by q where $r = \min\{i : \sum_{i'=1}^{i} \lambda_{i'}^2 / \sum_{i'=1}^{\min\{p_1,p_2\}} \lambda_{i'}^2 \geq q\}$. We use q since it is standardized in the interval $[0, 1]$, whereas λ is unbounded.

A commonly used rule-of-thumb is to choose $q = 0.9$ if the training size is sufficiently large. To avoid rejecting the null hypothesis H_0 in (2.1) too frequently (i.e., experiencing high false alarms) when the data is noisy and the training size is small, a small value of q can be chosen, resulting in a smaller r. Following the definitions in (2.2), we obtain

$$\widehat{M}_0 = \sum_{i=1}^{\min\{p_1,p_2\}} \hat{\lambda}_i \hat{u}_i \hat{v}_i^\top \quad \text{and} \quad \hat{\beta}_{i,t} = \hat{u}_i^\top X_t \hat{v}_i \quad \text{for } i = 1, \dots, r.$$

We also obtain $\hat{\gamma}_{i,t}$ for $i = 1, \dots, \min\{p_1, p_2\}$ as the singular values of $\widehat{R}_t = X_t - \widehat{M}_0$.

Selection of c*.* The constant c in (2.5) is related to the behavior of the ARL, according to the analysis in Section 2.4.3. It is preferable to choose a small, non-negative value of c so that $\mathbb{E}_0[d_T] < 0$ and $\mathbb{E}_1[d_T] > 0$ hold. [42] recommend that practitioners choose c within the range of 0.01 to 0.1 based on empirical studies.

2.4 Theoretical Analysis

In this section, we analyze the shift size in the expected values of the monitoring statistics T_t from the in-control to an out-of-control state. We also discuss the behaviors of ARL_0 and ARL_1 of the DFILM procedure. For theoretical analysis, we focus on high-dimensional matrices with large dimensions p_1, p_2 and p_1/p_2 bounded away from zero. But the algorithm works for small dimension matrices as well.

2.4.1 High-dimensional asymptotic analysis

For any fixed $i = 1, \dots, r$, it is straightforward to derive the in-control and out-of-control expectations of $\beta_{i,t}$:

$$\mathbb{E}_0[\beta_{i,t}] = u_i^\top \mathbb{E}_0[X_t] v_i = \lambda_i, \quad \mathbb{E}_1[\beta_{i,t}] = u_i^\top \mathbb{E}_1[X_t] v_i = \lambda_i + \alpha_i,$$

where α_i is defined as the static projection of the unknown shift A, namely

$$\alpha_i = u_i^\top A v_i, \quad i = 1, \dots, r. \tag{2.8}$$

Theorem 2.4.1 (Mean shift in $\beta_{i,t}$)**.** *For any time t and index $i = 1, \dots, r$, the difference between the in-control and out-of-control expectations of $\beta_{i,t}$ is*

$$\mathbb{E}_1[\beta_{i,t}] - \mathbb{E}_0[\beta_{i,t}] = \alpha_i,$$

where α_i is defined in (2.8).

From Theorem 2.4.1, the statistic $\beta_{i,t}$ can capture the change, provided that the differences α_i are non-zero for some i. By the definition of α_i, when A do not lie in the null space of M_0. In such cases, constructing statistics for detecting a shift in the expectation of $\beta_{i,t}$ can be effective. On the other hand, if A lies near the null space of M_0, then α_i oscillates around zero. Due to this consideration, to enhance the robustness of the detection procedure by incorporating another statistic, $\gamma_{i,t}$.

To analyze the in-control and out-of-control properties of the statistic $\gamma_{i,t}$, We need the following two lemmas. The following lemma is a direct corollary of Theorem 2 in [46]:

Lemma 2.4.2. *For any time $t < k$, suppose $R_t \in \mathbb{R}^{p_1 \times p_2}$ has independent and identically distributed (i.i.d.) entries with mean zero and variance σ^2, and their fourth moments are finite. As $p_1, p_2 \to \infty, p_1/p_2 \to \eta \in (0,1)$, we have:*

$$\sigma\left(1 - \sqrt{\eta}\right) \leq \lim_{p_1, p_2 \to \infty} \frac{\gamma_{i,t}}{\sqrt{p_2}} \leq \sigma\left(1 + \sqrt{\eta}\right), \quad i = 1, \dots, r, \text{ a.s.},$$

where $\gamma_{i,t}$ is the ith largest singular value of R_t.

The lemma provides asymptotic upper and lower bounds for the in-control $\gamma_{i,t}$. Specifically, the order of $\gamma_{i,t}$ is $\Theta\left(\sqrt{p_2}\right)$. In the out-of-control case, the residual matrix can be decomposed into the shift matrix A and the zero-mean matrix $R_t - A$. Lemma 2.4.3 indicates that the asymptotic behavior of R_t is dominated by the deterministic limiting behavior of A, rather than the random matrix $R_t - A$. The following lemma is a consequence of Theorem 1.1 in [47]:

Lemma 2.4.3. *For $t \geq k$, suppose that $R_t - A$ have i.i.d. entries with zero mean and variance σ^2, their fourth moments are finite, and the limit $\lim_{p_1,p_2\to\infty} p_1/p_2 = \eta > 0$ exists. For $i = 1, \ldots, r$, let ρ_i denote the ith largest singular value of A. Furthermore, assume $\lim_{p_1,p_2\to\infty} \rho_i/\sqrt{p_1 p_2} = \bar{\rho}_i$ exists and is distinct and strictly positive, satisfying $\bar{\rho}_1 > \ldots > \bar{\rho}_r > 0$. Then $\gamma_{i,t}$ can be decomposed as*

$$\gamma_{i,t} = \rho_i + z_i + m_i + \varepsilon_i, \quad i = 1, \ldots, r \tag{2.9}$$

where z_i is a random variable dependent on the dimensions p_1 and p_2 with zero mean and bounded variance, ε_i is a random variable converging to zero in probability with respect to the dimensions p_1 and p_2, and the deterministic term m_i satisfies

$$m_i = \frac{\sigma^2}{2}\left(\frac{\sqrt{\eta}}{\bar{\rho}_i^3 p_1 p_2} - \frac{1}{\sqrt{\eta}\bar{\rho}_i}\right).$$

This lemma decomposes the out-of-control $\gamma_{i,t}$ defined in (2.9) into several terms. The random noise ε_i vanishes as the dimensions grow, and z_i has zero mean. The deterministic term m_i diminishes with the dimensions p_1 and p_2, converging to a constant of $-\sigma^2/(2\sqrt{\eta}\bar{\rho}_i)$. Given that the singular value of the underlying anomaly ρ_i is of the order $\Theta(\sqrt{p_1 p_2})$, it dominates the asymptotic behavior of $\gamma_{i,t}$.

Theorem 2.4.4 (Asymptotic mean shift in $\gamma_{i,t}$)**.** *With the assumptions in Lemmas 2.4.2 and 2.4.3, the limit of the expected difference between in-control and out-of-control $\gamma_{i,t}$ is expressed as follows:*

$$\lim_{p_1,p_2\to\infty} \frac{\mathbb{E}_1[\gamma_{i,t}] - \mathbb{E}_0[\gamma_{i,t}]}{\sqrt{p_1 p_2}} = \bar{\rho}_i.$$

The proof of the theorem is available in Appendix A.2. Asymptotically, the expectation of the singular values of R_t under an out-of-control state is dominated by the singular values of the shift A. Thus, incorporating γ_t into statistics enhances the robustness of the detection capability of the DFLIM procedure against the algebraic relationship between M_0 and A.

As mentioned in Theorem 2.4.1, if the left or right singular space of the shift A is orthogonal to span$(u_1, \ldots, u_r)$ or span$(v_1, \ldots, v_r)$, $\beta_{i,t}$ may not accurately reflect the shift, and $\gamma_{i,t}$ may help with detection. However, we choose not to rely solely on $\gamma_{i,t}$ for detection because β_t demonstrates stronger detection power when the shift A is close in shape to the in-control mean M_0. Consider an extreme scenario where, for some non-zero constant c', the equation $A = c'M_0$ holds. For large dimensions p_1 and p_2, Theorem 2.4.1, Lemma 2.4.2, and Lemma 2.4.3 imply

$$\begin{aligned}
&(\mathbb{E}_1[\beta_{i,t}] - \mathbb{E}_0[\beta_{i,t}]) - (\mathbb{E}_1[\gamma_{i,t}] - \mathbb{E}_0[\gamma_{i,t}]) \\
&= \alpha_i - \left(\rho_i + \frac{\sigma^2}{2}\left(\frac{\sqrt{\eta}}{\bar{\rho}_i^3 p_1 p_2} - \frac{1}{\sqrt{\eta}\bar{\rho}_i}\right) + \mathbb{E}_1[\varepsilon_i] - \mathbb{E}_0[\gamma_{i,t}]\right) \\
&= c'\lambda_i - \left(c'\lambda_i + \frac{\sigma^2}{2}\left(\frac{\sqrt{\eta}}{\bar{\rho}_i^3 p_1 p_2} - \frac{1}{\sqrt{\eta}\bar{\rho}_i}\right) + \mathbb{E}_1[\varepsilon_i] - \mathbb{E}_0[\gamma_{i,t}]\right) \\
&= \frac{\sigma^2}{2}\left(\frac{1}{\sqrt{\eta}\bar{\rho}_i} - \frac{\sqrt{\eta}}{\bar{\rho}_i^3 p_1 p_2}\right) - \mathbb{E}_1[\varepsilon_i] + \mathbb{E}_0[\gamma_{i,t}] \\
&\geq \sigma(1-\sqrt{\eta})\sqrt{p_2} + \mathrm{o}(\sqrt{p_2}) + \frac{\sigma^2}{2}\left(\frac{1}{\sqrt{\eta}\bar{\rho}_i} - \frac{\sqrt{\eta}}{\bar{\rho}_i^3 p_1 p_2}\right) - \mathbb{E}_1[\varepsilon_i]. \qquad (2.10)
\end{aligned}$$

If we impose the assumption that for any dimensions p_1 and p_2, the random variable ε_i is uniformly integrable, then the convergence of ε_i to zero in probability implies the convergence of the expectation of ε_i, namely $\mathbb{E}_1[\varepsilon_i] = \mathrm{o}(1)$. On the right-hand side of inequality (2.10), if $\eta > 0$, the dominant term is $\sigma(1-\sqrt{\eta})\sqrt{p_2}$, which grows to infinity with respect to dimensions p_1 and p_2. In a special case where η is close to 1, the term $\sigma^2\left(1/\sqrt{\eta}\bar{\rho}_i - \sqrt{\eta}/(\bar{\rho}_i^3 p_1 p_2)\right)/2$ remains positive and still shows the advantage of the feature vector $\beta_{i,t}$ compared with $\gamma_{i,t}$. Therefore, both statistics in y_t are necessary for effective and robust detection.

2.4.2 Non-asymptotic analysis allowing spatial dependence

We next present an alternative non-asymptotic derivation for the expected increment term of the CUSUM procedure (2.4), before and after the change. This approach does not rely on lemmas derived from the asymptotic behavior of random matrices; instead, it utilizes second-order information from the detection statistics involving y_t.

Although the connection between the spatial dependence in X_t and the structure of $\mathrm{Cov}_\bullet(y_t)$, $\bullet \in \{0,1\}$ is challenging to characterize– the covariances $\mathrm{Cov}_\bullet(\gamma_t)$ and $\mathrm{Cov}_\bullet(\beta_t, \gamma_t)$ remain unknown– because γ_t consists of the singular values of a (possibly non-central) random matrix with correlated entries. Instead of delving into the structure of $\mathrm{Cov}_\bullet(y_t)$ based on specific spatial dependencies, we express the covariances of β_t and γ_t in a sufficiently general form to capture all potential structures arising from the spatial dependence in X_t. This approach allows the analysis of T_t to accommodate cases where spatial dependence exists in X_t.

The in-control and out-of-control covariance matrices of y_t as follows:

$$\mathrm{Cov}_0(y_t) = \Sigma = \begin{bmatrix} \Sigma_\beta & P \\ P^\top & \Sigma_\gamma \end{bmatrix}, \quad \mathrm{Cov}_1(y_t) = \widetilde{\Sigma} = \begin{bmatrix} \Sigma_\beta & \widetilde{P} \\ \widetilde{P}^\top & \widetilde{\Sigma}_\gamma \end{bmatrix},$$

where

$$\Sigma_\beta = \mathrm{Cov}_0(\beta_t),\ P = \mathrm{Cov}_0(\beta_t, \gamma_t),\ \widetilde{P} = \mathrm{Cov}_1(\beta_t, \gamma_t),\ \Sigma_\gamma = \mathrm{Cov}_0(\gamma_t),\ \text{ and } \widetilde{\Sigma}_\gamma = \mathrm{Cov}_1(\gamma_t).$$

Clearly, $\Sigma_\beta = \mathrm{Cov}_1(\beta_t)$ when only a mean shift is assumed, as in a typical mean-shift detection problem. Corresponding to the block structures that differentiate β and γ, we also define the expected shift size in them as $\mathbb{E}_1[y_t] - \mathbb{E}_0[y_t] = \delta = [\delta_\beta^\top, \delta_\gamma^\top]^\top$.

Theorem 2.4.5. *The difference between the in-control and out-of-control expectations of* *the T_t statistics is expressed as follows:*

$$\mathbb{E}_1[T_t] - \mathbb{E}_0[T_t] = \Delta_1 + \Delta_2 + \Delta_3,$$

where

$$\begin{aligned}
&\Delta_1 = \operatorname{tr}\left[(\Sigma/\Sigma_\beta)^{-1}\left(\widetilde{\Sigma}/\Sigma_\beta\right)\right] - r, \\
&\Delta_2 = \operatorname{tr}\left\{(\Sigma/\Sigma_\beta)^{-1}\left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]^\top \left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]\right\}, \\
&\Delta_3 = \left\|\Sigma_\beta^{-1/2}\delta_\beta\right\|^2 + \left\|(\Sigma/\Sigma_\beta)^{-1/2}\left(P^\top \Sigma_\beta^{-1}\delta_\beta + \delta_\gamma\right)\right\|^2, \\
&\Sigma/\Sigma_\beta := \Sigma_\gamma - P^\top \Sigma_\beta^{-1} P, \quad \textit{and} \quad \widetilde{\Sigma}/\Sigma_\beta := \widetilde{\Sigma}_\gamma - \widetilde{P}^\top \Sigma_\beta^{-1}\widetilde{P}.
\end{aligned}$$

The proof of the theorem is provided in Appendix A.2. According to Theorem 2.4.5, Δ_1 is small when $\widetilde{\Sigma}$ deviates slightly from Σ. In contrast, Δ_2 is positive since it is the trace of the multiplication of two positive definite matrices. The last term, Δ_3, increases quadratically with the shift sizes in β_t and γ_t. Assuming non-zero shift sizes in β_t or γ_t, Δ_3 is also positive as validated by Theorems 2.4.1 and 2.4.4 under the assumption of independence among matrix entries. Overall, Theorem 2.4.5 demonstrates the approximate positivity of the mean shift in T_t, enabling DFLIM to raise an alarm after the change point quickly.

2.4.3 ARL_0 and ARL_1

In this section, we provide expressions of ARL_0 and ARL_1. Recall that ARL_0 and ARL_1 represent $\mathbb{E}_0[\tau]$ and $\mathbb{E}_1[\tau]$, respectively. According to Lemma 2.3.1, for the in-control case, we have:

$$\mathrm{ARL}_0 = \mathbb{E}_0[\tau] \approx \frac{\Omega_0^2}{2(c\sigma_T)^2}\left[\exp\left(\frac{2c\sigma_T H}{\Omega_0^2}\right) - 1 - \frac{2c\sigma_T H}{\Omega_0^2}\right],$$

where ARL_0 increases exponentially with respect to the control limit H. For an out-of-control case, we obtain:

$$\mathrm{ARL}_1 = \mathbb{E}_1\left[\tau\right] \approx \frac{\Omega_1^2}{2d_1^2}\left[\exp\left(-\frac{2Hd_1}{\Omega_1^2}\right) - 1 + \frac{2d_1H}{\Omega_1^2}\right],$$

where Ω_1^2 represents Ω^2 under the out-of-control probability measure and $d_1 = \mathbb{E}_1[T_t] - \mathbb{E}_0[T_t] - c\sigma_T$. Given the approximate positivity of d_1 as validated in Theorem 2.4.5, ARL_1 increases on the order of $\mathrm{O}(H)$ (i.e., increases linearly with respect to H). Consequently, for a large H, we have $\mathrm{ARL}_1 \ll \mathrm{ARL}_0$.

To study the behavior of ARL_1 when the assumption of unchanged Ω^2 is violated, we consider the case where the process variability shifts from Ω_0^2 to Ω_1^2. Performing a Taylor expansion on $\mathbb{E}_1[\tau]$, we obtain

$$\begin{aligned}\mathbb{E}_1[\tau] &\approx \frac{\Omega_1^2}{2d_1^2}\left[1 - \frac{2d_1H}{\Omega_1^2} + \left(\frac{2d_1H}{\Omega_1^2}\right)^2 + o\left(\frac{1}{\Omega_1^4}\right) - 1 + \frac{2d_1H}{\Omega_1^2}\right] \\ &= \frac{2H^2}{\Omega_1^2} + o\left(\frac{1}{\Omega_1^2}\right).\end{aligned}$$

This approximation illustrates that if $\Omega_1^2 > \Omega_0^2$, the detection of a shift becomes even faster compared to the case of unchanged Ω^2. On the other hand, Ω_1^2 is smaller than Ω_0^2, detection becomes slower. However, even with a decrease in Ω_1^2, the reduction is typically not substantial compared to the size of Ω_0^2, which tends to be large in practice. Therefore, the slowdown in detection is usually not significant.

2.5 Simulated Experiments

In this section, we perform numerical experiments to compare the DFLIM procedure with existing procedures. Specifically, we consider the following three baselines: (1) MEWMA [48], which applies parallelized multivariate EWMA control charts to matrix data for efficient detection of local changes; (2) MGLR [49], which constructs monitoring statistics based on ROI and corresponding likelihood ratios; (3) ST-SSD [30], which decomposes the data into informative characteristics and noises.

2.5.1 Settings

First, we generate in-control matrices $\{X_t \in \mathbb{R}^{p_1 \times p_2} : t = 1, 2, \ldots\}$ using the following equation:

$$X_t = M_0 + \sum_{j=0}^{\ell} \phi^j \varepsilon_{t-j}, \tag{2.11}$$

where M_0 is the low-rank in-control mean matrix, ε_t denote noise matrices, and $\phi > 0$ is the auto-correlation parameter. Equation (2.11) represents a moving average model of order ℓ. In our simulated experiments, we set $p_1 = 100$ and $p_2 = 200$. The in-control data are generated as follows:

- We vary the ranks of M_0 within the set $\{2, 5\}$. The rank-two M_0 is constructed as a chessboard with elements $M_0[j_1, j_2]$ for $j_1 = 1, 2, \ldots, 100$ and $j_2 = 1, 2, \ldots, 200$:

$$M_0[j_1, j_2] = \begin{cases} 0.1, & 10k_1 + 1 \leq j_1 \leq 10k_1 + 5,\ 40k_2 + 11 \leq j_2 \leq 40k_2 + 20, \\ 0.1, & 10k_1 + 6 \leq j_1 \leq 10k_1 + 10,\ 40k_2 + 21 \leq j_2 \leq 40k_2 + 30, \\ -0.1, & 10k_1 + 1 \leq j_1 \leq 10k_1 + 5,\ 40k_2 + 31 \leq j_2 \leq 40k_2 + 40, \\ -0.1, & 10k_1 + 6 \leq j_1 \leq 10k_1 + 10,\ 40k_2 + 1 \leq j_2 \leq 40k_2 + 10, \\ 0, & \text{otherwise,} \end{cases}$$

  where $0 \leq k_1 \leq 4$ and $0 \leq k_2 \leq 9$. To extend the rank to five, we superimpose the rank-three approximation of the truncated first image of solar flare image data [50] onto the chessboard signal. Figure 2.1 displays the resulting images of the rank-two and rank-five M_0.

- For the distribution of ε_t, we select one from the following two distributions:

(1) *Normal distribution:* We generate ε_t using the matrix normal distribution [51, Chapter 2]. Thus,

$$\varepsilon_t \overset{\text{i.i.d.}}{\sim} \mathcal{MN}\left(0, \Sigma_{\text{row}}, \Sigma_{\text{col}}\right), \tag{2.12}$$

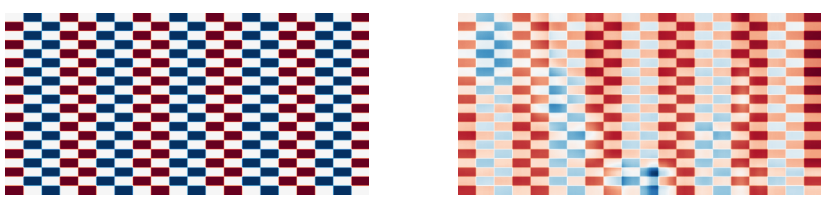
where $\Sigma_{\text{row}} \in \mathbb{R}^{p_1 \times p_1}$ and $\Sigma_{\text{col}} \in \mathbb{R}^{p_2 \times p_2}$ are specified to capture the spatial correlations.

(2) *Non-normal distribution:* We first generate $\tilde{\varepsilon}_t$ from (2.12) and then transform each entry $\tilde{\varepsilon}_t[j_1, j_2]$ into $\varepsilon_t[j_1, j_2]$ as follows:

$$\varepsilon_t[j_1, j_2] = -\log(1 - \Psi(\tilde{\varepsilon}_t[j_1, j_2])),$$

where $\Psi(\cdot)$ represents the cumulative distribution of the standard normal random variable. Thus, ε_t has entries $\varepsilon_t[j_1, j_2]$ that are exponentially distributed with an expectation of 1 and exhibit correlations among them.

- To incorporate spatial correlation, both covariance matrices Σ_{row} and Σ_{col} are specified using either tri-diagonal covariance or exponential covariance, defined as fol-



(a) Rank$(M_0) = 2$ (b) Rank$(M_0) = 5$

Figure 2.1: In-control mean matrix M_0 with rank $r \in \{2, 5\}$.

lows:

$$
\begin{aligned}
\text{Tri-diagonal} &= \begin{bmatrix} 1 & 0.3 & \cdots & 0 & 0 \\ 0.3 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0.3 \\ 0 & 0 & \cdots & 0.3 & 1 \end{bmatrix}, \\
\text{Exponential} &= \begin{bmatrix} 1 & 0.3 & \cdots & 0.3^{p-2} & 0.3^{p-1} \\ 0.3 & 1 & \cdots & 0.3^{p-3} & 0.3^{p-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0.3^{p-2} & 0.3^{p-3} & \cdots & 1 & 0.3 \\ 0.3^{p-1} & 0.3^{p-2} & \cdots & 0.3 & 1 \end{bmatrix},
\end{aligned}
\tag{2.13}
$$

where the dimensions of the covariance matrices vary in $\{p_1, p_2\}$, corresponding to the row and column sides.

- To incorporate auto-correlation, we vary the lag order $\ell \in \{5, 20\}$ for the moving average term $\sum_{j=0}^{\ell} \phi^j \varepsilon_{t-j}$ in (2.11), with the parameter ϕ fixed as $\phi = 0.5$.

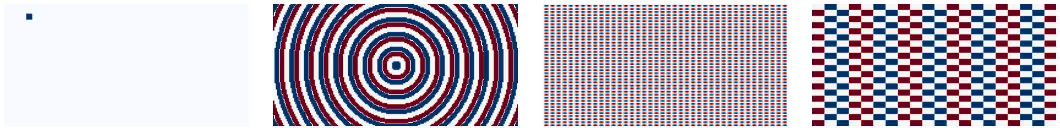
We generate the out-of-control matrices according to the following equation:

$$X_t = M_0 + A + \sum_{j=0}^{\ell} \phi^j \varepsilon_{t-j},$$

where the mean M_0 and the noise $\sum_{i=0}^{\ell} \phi^i \varepsilon_{t-i}$ follow the same settings as in the in-control matrices. We test the following four mean shifts for A:

(i) Sparse:

$$A[j_1, j_2] = \begin{cases} 3, & \text{if } 8 \le j_1 \le 13 \text{ and } 18 \le j_2 \le 23, \\ 0, & \text{otherwise.} \end{cases}$$



(a) Sparse (b) Ring (c) Sine (d) Chessboard

Figure 2.2: Four patterns of shift matrix A.

(ii) Ring:

$$A[j_1, j_2] = \begin{cases} 0.173, & \text{if } \lfloor \sqrt{(j_1-50)^2 + (j_2-100)^2} \rfloor = 12k_1 + k_2, \\ -0.173, & \text{if } \lfloor \sqrt{(j_1-50)^2 + (j_2-100)^2} \rfloor = 12k_1 + 8 + k_2, \\ 0, & \text{otherwise,} \end{cases}$$

where k_1 is some non-negative integer and $0 \leq k_2 \leq 3$.

(iii) Sine:

$$A[j_1, j_2] = 0.283 \sin \frac{j_2 \pi}{5} \sin \frac{2 j_1 \pi}{5}.$$

(iv) Chessboard: identical to the rank-two M_0.

The visual representation of the four shift patterns is provided in Figure 2.2.

2.5.2 In-control performance

In each in-control setting, we generate 1000 sequences, each consisting of 800 matrix observations. The target ARL_0 is set to 200. Baselines MEWMA, MGLR, and ST-SSD tune their control limits through trial-and-error using these 1000 sequences. For DFLIM, we set $c = 0.01$ for all experiments and determine the control limit by solving equation (2.7) for H. Each procedure is then applied to these 1000 sequences using its determined control limit to estimate ARL_0.

In each in-control setting, we generate 1000 sequences, each consisting of 800 matrix observations. The target ARL_0 is set to 200. Baseline methods MEWMA, MGLR, and

Table 2.1: Control limit (H) and estimated ARL_0 for various settings of simulated processes with target $\mathrm{ARL}_0 = 200$ (standard errors in parentheses).

				MEWMA		MGLR		ST-SSD		DFLIM	
Distribution	Rank	Lag	Covariance	H	ARL_0	H	ARL_0	H	ARL_0	H	ARL_0
Normal	2	5	Tri-diagonal	180.202	198.62 (6.248)	6.062	201.04 (5.495)	2.800	201.05 (5.680)	36.507	201.48 (5.321)
Normal	2	5	Exponential	180.219	200.41 (5.910)	6.043	200.72 (5.528)	2.799	199.42 (5.692)	36.654	197.26 (5.119)
Normal	2	20	Tri-diagonal	180.455	200.20 (5.866)	6.155	198.46 (5.465)	2.796	199.87 (6.036)	36.776	203.02 (5.188)
Normal	2	20	Exponential	180.415	198.12 (5.890)	6.236	200.17 (5.574)	2.804	200.17 (5.698)	36.935	200.81 (5.103)
Normal	5	5	Tri-diagonal	184.765	198.72 (8.941)	6.039	200.94 (5.694)	2.779	198.83 (5.576)	36.551	201.30 (5.297)
Normal	5	5	Exponential	183.968	201.29 (9.059)	6.015	198.83 (5.469)	2.796	201.10 (5.715)	36.688	201.28 (5.173)
Normal	5	20	Tri-diagonal	184.236	201.88 (9.081)	6.221	199.75 (5.640)	2.808	200.98 (5.687)	36.632	205.51 (5.354)
Normal	5	20	Exponential	184.509	198.50 (9.053)	6.203	200.92 (5.574)	2.783	198.36 (5.827)	36.641	197.76 (5.167)
Non-normal	2	5	Tri-diagonal	294.912	199.32 (10.903)	6.042	201.49 (5.562)	2.837	198.39 (5.935)	37.208	202.81 (5.167)
Non-normal	2	5	Exponential	296.649	201.74 (10.953)	6.013	200.26 (5.526)	2.814	200.38 (5.988)	37.416	200.32 (5.292)
Non-normal	2	20	Tri-diagonal	297.494	199.15 (10.911)	6.191	201.17 (5.839)	2.828	198.72 (5.883)	37.359	204.95 (5.187)
Non-normal	2	20	Exponential	299.497	200.49 (10.930)	6.173	198.74 (5.435)	2.834	200.51 (5.846)	37.510	202.34 (5.425)
Non-normal	5	5	Tri-diagonal	389.186	199.95 (10.926)	6.017	199.69 (5.446)	2.821	199.74 (5.596)	37.369	205.32 (5.383)
Non-normal	5	5	Exponential	391.227	199.95 (10.926)	5.988	200.12 (5.441)	2.832	199.70 (5.634)	37.283	206.09 (5.357)
Non-normal	5	20	Tri-diagonal	394.891	199.15 (10.911)	6.212	200.55 (5.578)	2.818	200.67 (5.817)	37.483	207.76 (5.351)
Non-normal	5	20	Exponential	394.642	199.95 (10.926)	6.215	201.63 (5.533)	2.834	200.08 (5.867)	37.565	206.17 (5.253)

ST-SSD tune their control limits through trial-and-error using these 1000 sequences. For DFLIM, we determine the control limit by solving equation (2.7) for H. Subsequently, each procedure is applied to the same dataset to find the control limits.

Table 2.1 summarizes the control limits (H) and estimated ARL_0 values for different procedures. MEWMA, MGLR, and ST-SSD achieve ARL_0 values close to the target 200, as expected, because their control limits are determined through trial-and-error calibration using the same data sequences. The table also demonstrates that the analytically determined control limits for DFLIM provide relatively accurate ARL_0 values close to the target. The simulated data has complicated spatial correlations as defined in (2.13), and the target ARL_0 is set to a small value, potentially increasing the risk that the time horizon might not be long enough for the FCLT to be applicable. Despite these slight theoretical violations, the empirical ARL_0 values obtained for DFLIM are still quite accurate. These results suggest that (2.7) is widely applicable in practice.

2.5.3 Out-of-control performance

In out-of-control experiments, we consider shift matrix A with sparse, ring, sine, and chessboard patterns as described in Section 2.5.1, which are added to the in-control data. The experiment is repeated 1000 times for each procedure. The results of the out-of-control

Table 2.2: Sparse shift: ARL_1 for various settings of simulated processes with target $\mathrm{ARL}_0 = 200$ (standard errors in parentheses).

Distribution	Rank	Lag	Covariance	MEWMA	MGLR	ST-SSD	DFLIM
Normal	2	5	Tri-diagonal	198.93 (6.227)	207.34 (5.756)	79.17 (2.347)	15.06 (0.232)
Normal	2	5	Exponential	197.69 (6.249)	200.58 (5.425)	85.01 (2.603)	16.66 (0.289)
Normal	2	20	Tri-diagonal	196.08 (6.190)	188.46 (5.211)	80.64 (2.597)	15.61 (0.264)
Normal	2	20	Exponential	201.89 (6.215)	200.11 (5.618)	84.05 (2.600)	16.49 (0.278)
Normal	5	5	Tri-diagonal	180.50 (9.163)	202.62 (5.683)	74.58 (2.334)	14.77 (0.233)
Normal	5	5	Exponential	155.01 (8.373)	197.18 (5.409)	84.31 (2.661)	16.09 (0.271)
Normal	5	20	Tri-diagonal	153.96 (8.399)	198.62 (5.486)	83.17 (2.589)	15.18 (0.258)
Normal	5	20	Exponential	175.24 (8.724)	202.35 (5.428)	77.40 (2.417)	16.39 (0.275)
Non-normal	2	5	Tri-diagonal	156.04 (9.991)	197.86 (5.533)	83.06 (2.675)	20.52 (0.353)
Non-normal	2	5	Exponential	178.38 (10.501)	194.10 (5.516)	84.74 (2.624)	21.11 (0.387)
Non-normal	2	20	Tri-diagonal	163.70 (10.163)	191.24 (5.293)	82.38 (2.530)	21.03 (0.372)
Non-normal	2	20	Exponential	174.38 (10.415)	189.54 (5.334)	85.97 (2.708)	21.20 (0.388)
Non-normal	5	5	Tri-diagonal	164.00 (10.182)	202.14 (5.522)	85.50 (2.744)	21.03 (0.372)
Non-normal	5	5	Exponential	176.78 (10.467)	182.21 (5.268)	87.27 (2.784)	21.78 (0.396)
Non-normal	5	20	Tri-diagonal	182.37 (10.584)	195.56 (5.465)	80.65 (2.769)	20.80 (0.379)
Non-normal	5	20	Exponential	156.80 (10.011)	196.41 (5.581)	90.09 (2.824)	22.01 (0.397)

performances are summarized in Tables 2.2 –2.5 for each shift pattern A.

The DFLIM procedure consistently outperforms the MEWMA and MGLR procedures across all shift patterns. The MEWMA procedure detects shifts only in a few cases but with significant delay. MEWMA employs a profile monitoring technique, where the matrix is flattened into a long vector and segmented for separate handling. This approach risks losing spatial correlations due to flattening and segmentation. Another drawback is that each segment remains high-dimensional, making covariance estimation for each segment challenging. In our experiment, with 800 in-control data and segments of dimension 200, MEWMA's performance suffers due to poor marginal covariance estimation across segments. Additionally, MEWMA assumes temporal independence, which is invalid in our complex auto-correlations experiments.

The MGLR procedure reduces data dimensionality by defining ROIs, but this process can result in local information loss within each ROI. Considering the nature of our shifts (sparse or alternating positive and negative), the mean of entries in each ROI tends to be close to zero. This cancellation of informative entries by taking the mean of each ROI in MGLR undermines its ability to detect a shift effectively.

Table 2.3: Ring shift: ARL_1 for various settings of simulated processes with target $\mathrm{ARL}_0 = 200$ (standard errors in parentheses).

Distribution	Rank	Lag	Covariance	MEWMA	MGLR	ST-SSD	DFLIM
Normal	2	5	Tri-diagonal	196.30 (6.062)	194.71 (5.712)	21.50 (0.687)	28.69 (0.498)
Normal	2	5	Exponential	197.92 (6.041)	209.54 (5.775)	23.73 (0.758)	27.41 (0.476)
Normal	2	20	Tri-diagonal	199.30 (6.110)	190.31 (5.435)	20.05 (0.610)	29.10 (0.490)
Normal	2	20	Exponential	197.52 (5.880)	198.82 (5.691)	24.04 (0.734)	26.81 (0.435)
Normal	5	5	Tri-diagonal	204.50 (9.416)	198.12 (5.554)	20.22 (0.579)	27.86 (0.461)
Normal	5	5	Exponential	159.62 (8.229)	194.84 (5.631)	24.05 (0.728)	26.20 (0.440)
Normal	5	20	Tri-diagonal	175.98 (8.595)	203.60 (5.518)	22.42 (0.661)	28.15 (0.490)
Normal	5	20	Exponential	184.53 (8.824)	195.35 (5.491)	24.35 (0.718)	27.19 (0.459)
Non-normal	2	5	Tri-diagonal	160.79 (10.106)	195.68 (5.555)	21.16 (0.648)	47.58 (0.929)
Non-normal	2	5	Exponential	193.44 (10.800)	208.74 (5.761)	24.00 (0.746)	40.47 (0.726)
Non-normal	2	20	Tri-diagonal	149.26 (9.814)	200.90 (5.598)	20.34 (0.626)	45.90 (0.869)
Non-normal	2	20	Exponential	181.07 (10.550)	193.64 (5.278)	23.34 (0.710)	42.59 (0.813)
Non-normal	5	5	Tri-diagonal	190.36 (10.744)	195.06 (5.407)	20.66 (0.633)	45.75 (0.852)
Non-normal	5	5	Exponential	192.76 (10.791)	185.25 (5.259)	25.85 (0.794)	39.06 (0.709)
Non-normal	5	20	Tri-diagonal	178.38 (10.501)	200.02 (5.537)	20.61 (0.639)	44.73 (0.869)
Non-normal	5	20	Exponential	191.16 (10.760)	190.57 (5.134)	26.90 (0.839)	41.36 (0.805)

Table 2.4: Sine shift: ARL_1 for various settings of simulated processes with target $\mathrm{ARL}_0 = 200$ (standard errors in parentheses).

Distribution	Rank	Lag	Covariance	MEWMA	MGLR	ST-SSD	DFLIM
Normal	2	5	Tri-diagonal	201.12 (5.933)	199.93 (5.511)	31.12 (0.996)	5.29 (0.081)
Normal	2	5	Exponential	209.35 (6.448)	198.88 (5.576)	37.34 (1.169)	16.17 (0.244)
Normal	2	20	Tri-diagonal	191.17 (5.963)	187.88 (5.256)	32.41 (1.047)	5.54 (0.086)
Normal	2	20	Exponential	194.94 (6.011)	203.80 (5.514)	35.90 (1.062)	16.45 (0.263)
Normal	5	5	Tri-diagonal	210.28 (9.451)	192.23 (5.592)	29.65 (0.923)	5.45 (0.083)
Normal	5	5	Exponential	176.80 (8.450)	187.72 (5.215)	35.86 (1.076)	16.31 (0.273)
Normal	5	20	Tri-diagonal	172.29 (8.513)	203.99 (5.598)	30.92 (0.909)	5.54 (0.087)
Normal	5	20	Exponential	195.50 (9.001)	192.17 (5.401)	35.05 (1.029)	16.5 (0.266)
Non-normal	2	5	Tri-diagonal	156.80 (10.011)	205.74 (5.920)	35.32 (1.079)	6.50 (0.097)
Non-normal	2	5	Exponential	192.28 (10.775)	197.62 (5.561)	37.04 (1.116)	16.87 (0.255)
Non-normal	2	20	Tri-diagonal	165.11 (10.197)	184.36 (5.086)	35.08 (1.060)	6.59 (0.098)
Non-normal	2	20	Exponential	170.39 (10.327)	186.27 (5.357)	39.12 (1.249)	16.68 (0.254)
Non-normal	5	5	Tri-diagonal	183.17 (10.600)	198.31 (5.558)	33.06 (0.970)	6.60 (0.096)
Non-normal	5	5	Exponential	209.54 (11.097)	187.94 (5.486)	40.92 (1.226)	17.27 (0.266)
Non-normal	5	20	Tri-diagonal	218.33 (11.243)	193.34 (5.370)	34.40 (1.094)	6.40 (0.100)
Non-normal	5	20	Exponential	169.59 (10.309)	203.86 (5.566)	39.21 (1.262)	16.41 (0.260)

Table 2.5: Chessboard shift: ARL_1 for various settings of simulated processes with target $\mathrm{ARL}_0 = 200$ (standard errors in parentheses).

Distribution	Rank	Lag	Covariance	MEWMA	MGLR	ST-SSD	DFLIM
Normal	2	5	Tri-diagonal	190.65 (6.146)	205.80 (5.895)	1.54 (0.030)	1.70 (0.017)
Normal	2	5	Exponential	193.27 (6.081)	207.11 (5.744)	1.78 (0.044)	1.97 (0.018)
Normal	2	20	Tri-diagonal	186.63 (6.145)	184.59 (4.887)	1.52 (0.031)	1.69 (0.017)
Normal	2	20	Exponential	199.48 (6.307)	199.14 (5.624)	1.74 (0.037)	1.97 (0.018)
Normal	5	5	Tri-diagonal	163.28 (8.942)	203.00 (5.583)	1.49 (0.030)	2.28 (0.021)
Normal	5	5	Exponential	135.63 (7.913)	199.87 (5.557)	1.76 (0.039)	2.62 (0.024)
Normal	5	20	Tri-diagonal	133.09 (7.833)	192.75 (5.604)	1.54 (0.031)	2.28 (0.021)
Normal	5	20	Exponential	146.96 (8.206)	198.12 (5.508)	1.77 (0.041)	2.59 (0.025)
Non-normal	2	5	Tri-diagonal	162.08 (10.129)	207.74 (5.870)	1.56 (0.030)	2.47 (0.023)
Non-normal	2	5	Exponential	171.71 (10.349)	196.14 (5.447)	1.75 (0.036)	2.77 (0.026)
Non-normal	2	20	Tri-diagonal	131.17 (9.328)	197.06 (5.611)	1.57 (0.031)	2.53 (0.023)
Non-normal	2	20	Exponential	157.46 (10.022)	195.98 (5.507)	1.91 (0.047)	2.80 (0.028)
Non-normal	5	5	Tri-diagonal	160.00 (10.088)	195.82 (5.581)	1.58 (0.031)	2.52 (0.024)
Non-normal	5	5	Exponential	169.59 (10.309)	192.56 (5.322)	1.82 (0.042)	2.85 (0.029)
Non-normal	5	20	Tri-diagonal	151.21 (9.872)	199.70 (5.491)	1.53 (0.032)	2.55 (0.025)
Non-normal	5	20	Exponential	145.62 (9.728)	200.69 (5.775)	1.86 (0.043)	2.86 (0.030)

The ST-SSD procedure is the most competitive baseline compared to DFLIM. In Table 2.2, DFLIM detects sparse shifts faster than ST-SSD, saving approximately 70 observations. Regarding the ring shift in Table 2.3, under normal noise distribution, DFLIM performs slightly worse than ST-SSD, with a lag of less than 10 observations. However, with non-normal noise distribution, ST-SSD outperforms DFLIM by approximately 20 observations, although both achieve significantly smaller ARL_1 values compared to MEWMA and MGLR. In Table 2.4, both DFLIM and ST-SSD detect the chessboard shift almost instantly, showing negligible differences between them. For the chessboard shift in Table 2.5, both DFLIM and ST-SSD procedures detect the shift nearly instantly, with a negligible difference between them.

The ST-SSD procedure employs a regression framework to decompose observations into three components: the in-control mean matrix M_0, the shift A, and the noise. Except for the sparse shift, ST-SSD achieves this decomposition effectively, resulting in successful detection. DFLIM incorporates static and dynamic statistics to construct monitoring statistics for change detection. In scenarios where a shift is algebraically similar to the in-control mean matrix, static features play a significant role in change detection, as seen in the chessboard shift of Figure 2.2(d). On the other hand, dynamic features dominate when a shift is algebraically different from the in-control mean matrix, as demonstrated in the sparse, ring, and sine shifts of Figures 2.2(a)-2.2(c). Hence, DFLIM robustly and efficiently detects changes across various settings.

As stated in Theorem 2.4.1, the first-type features $\beta_{i,t}$ consistently help achieve change detection when M_0 and M_1 are algebraically similar. This is empirically supported by the effectiveness of DFLIM in detecting the chessboard shift, which resembles the in-control mean. On the other hand, the effectiveness of the second-type features $\gamma_{i,t}$ becomes evident when dealing with large matrix dimensions, approaching the asymptotic theory outlined in Theorem 2.4.4. Experimental results show that normally distributed noises often lead to smaller ARL_1 compared to non-normal noises, likely due to slower convergence to the asymptotic theory associated with non-normal noises.

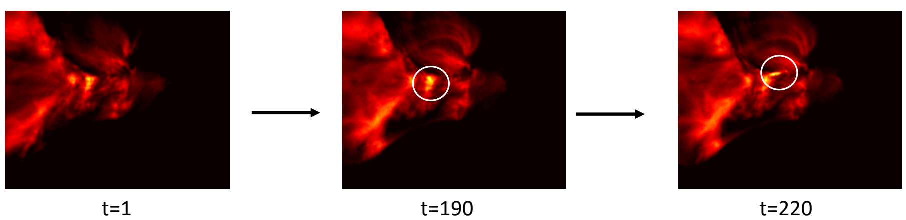


Figure 2.3: Solar flare images at $t = 1, 190, 220$. The white circles mark outbursts.

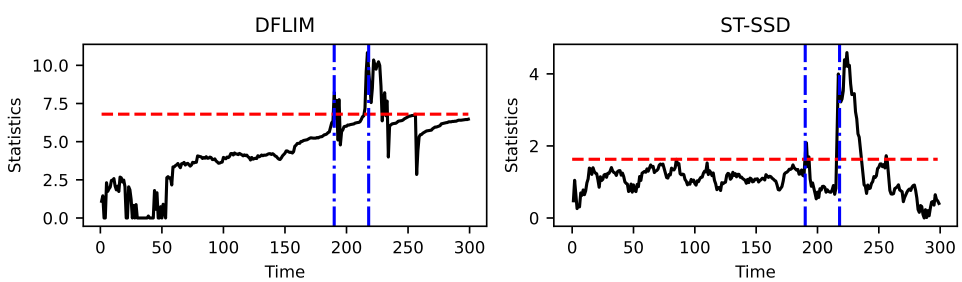


Figure 2.4: The monitoring process of DFLIM and ST-SSD on the solar flare images. Two outbursts occur around times 190 and 220, respectively, marked by vertical blue dashed lines. In each procedure, the control limit is indicated by a red dashed line, with the monitoring statistics shown in black.

2.6 Real Data Experiments

In this section, we apply DFLIM to real datasets to illustrate its broad applicability. More specifically, we analyze solar flare images in Section 2.6.1 and stochastic textured surface images in Section 2.6.2. We compare DFLIM with ST-SSD, excluding MEWMA and MGLR, because their control limits cannot be determined with a single in-control sequence.

We determine the control limit of DFLIM analytically by solving equation (2.7) for H. For ST-SSD, we determine its control limit using an empirical $(1 - 1/\mathrm{ARL}_0)$ quantile estimate of the in-control monitoring statistics [30, Section 5.2].

2.6.1 Solar flare outburst

In this example, we aim to detect solar flare outbursts. Figure 2.3 shows solar flare images at times $t = 1, 190, 220$. The solar flare outbursts are represented by the bright spots in the images, indicated by the white circles. Prior knowledge indicates that two outbursts happen around times 190 and 220, respectively. The first outburst around time 190 is relatively moderate, while the second one around time 220 is more intense. The sequence consists of 300 images, each represented by a 232-by-292 matrix. We use the first 150 matrices as the training dataset and then perform monitoring on the entire sequence.

Detecting solar flare outbursts presents several challenges. First, each image is high-dimensional, containing nearly 70,000 pixels. Second, the low-rank property is not inherently applicable to solar flare images. To address this issue, we employ a patch technique that breaks the data into *patches* to promote the low-rank structure [52]. Each patched image exhibits a numerical rank of 10. Third, the dynamics of the changes are complex due to multiple change points and slowly evolving backgrounds. Specifically, many time points around 190 and 220 experience outbursts. To handle these multiple outbursts, we restart monitoring once an alarm is raised. Additionally, the changes in the data involve not only intense outbursts but also slow shifts in the background. To address the dynamic background, we process the data by taking consecutive differences of the images after the patch technique is performed, and we set the target $\mathrm{ARL}_0 = 50,000$.

Figure 2.4 shows the results of ST-SSD and DFLIM to the solar flare images. ST-SSD effectively detects both moderate and intense outbursts without triggering false alarms. However, we observe that the monitoring statistic of ST-SSD is very close to its control limit around $t = 70, 80, 100$, corresponding to periods of normal solar flare activity. DFLIM demonstrates superior performance compared to ST-SSD, effectively identifying the outbursts around $t = 190$ and $t = 220$ with the detection statistics away from control limits prior to $t = 190$.

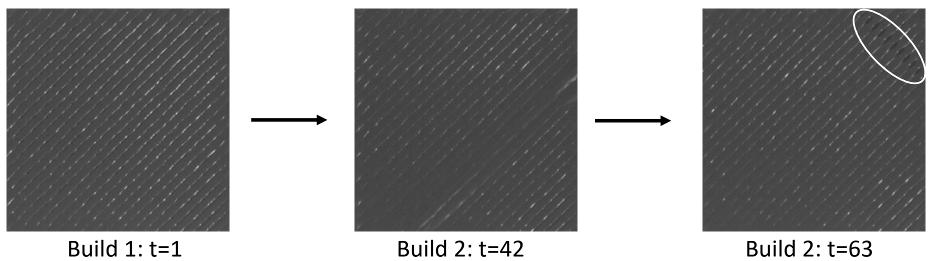


Figure 2.5: The material extrusion process is depicted at $t = 1, 42, 63$. At $t = 42$, there is a transition from the first to the second build. The bead orientation is $45°$, and the white ellipse marks a defect.

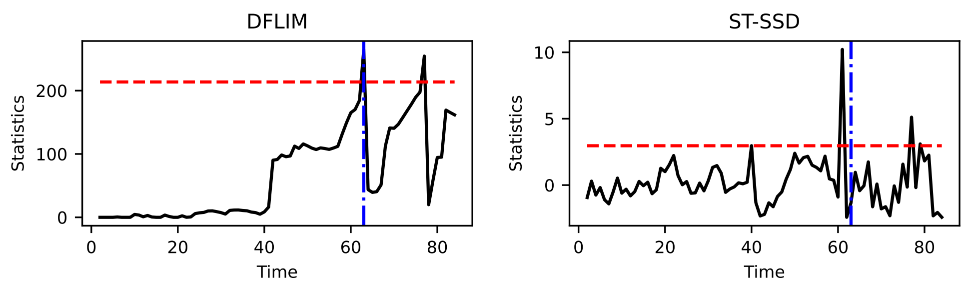


Figure 2.6: The monitoring process of DFLIM and ST-SSD on the 3D printing textured surface images with bead orientation at $45°$. A build-to-build transition happens at $t = 42$, while a defect is introduced at $t = 63$, marked by vertical blue dashed lines in both figures. In each procedure, the control limit is indicated by a red dashed line, with the monitoring statistics shown in black.

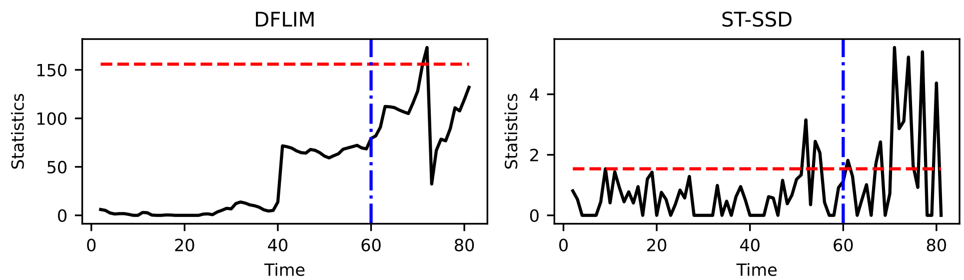


Figure 2.7: The monitoring process of DFLIM and ST-SSD on the 3D printing textured surface images with bead orientation at $135°$. A build-to-build transition happens at $t = 41$, while a defect is introduced at $t = 60$, marked by vertical blue dashed lines in both figures. In each procedure, the control limit is indicated by a red dashed line, with the monitoring statistics shown in black.

2.6.2 Stochastic textured surface monitoring

The online monitoring of additive manufacturing processes, commonly referred to as 3D printing, has drawn increasing attention due to its potential to reduce material waste. This example involves monitoring the production process of a parallelepiped ($20 \times 20 \times 20$ mm) using fused filament fabrication.

[1] use data from a sequential process to print two parallelepipeds. We use the same dataset and, refer to them as Build 1 (representing the process of building the first parallelepiped) and Build 2 (representing the process of building the second parallelepiped) hereafter. Build 1 is in control, while defects are intentionally introduced into Build 2 in the middle of the printing process.

During the printing process, layerwise images are captured by a video-imaging system installed above the printing area. To optimize bonding between consecutive layers, it is recommended that the material extrusion direction be rotated iteratively. Consequently, the captured images are categorized into two types based on bead orientations: $45°$ and $135°$. Each build consists of two sequences labeled as $45°$ and $135°$, respectively, and these sequences are treated separately. Despite both sequences originating from the same build process, some images are occasionally skipped and not captured. Therefore, the index t in this example corresponds to the index of captured images and does not directly translate into time. The sequence with bead orientation $45°$ ($135°$) consists of 84 (81) images. For Build 1 (Build 2), the sequence ends at $t = 42$ ($t = 41$), after which defects are introduced at $t = 63$ ($t = 60$). Each image is represented as a 250-by-250 matrix. During monitoring, we utilize images from Build 1 to set up control limits and implementation parameters, then apply DFLIM and ST-SSD to the all the images.

Unlike the solar flare images, we do not need the patch technique for this dataset because the in-control data naturally exhibits low-rank properties due to the aligned paths of material extrusions. However, this dataset presents challenges similar to those in the solar flare images, including high dimensionality and evolving backgrounds. Additionally, two more challenges arise with this dataset. First, the training sample size is small, consisting of only about 40 images. Second, the dataset exhibits build-to-build variability, which stems from dynamic factors in the printing area, such as changing illumination conditions during transitions between builds. Technically, the shift between builds could be considered a change point, but it is undesirable to detect this inter-build shift. Instead, we aim to detect a shift caused by actual defects, but the build-to-build variability increases the risk of false alarms.

For monitoring, we still take consecutive differences of images and restart the process upon detecting a change point. We set $\mathrm{ARL}_0 = 4000$ for both bead orientations, suggesting approximately half a month between consecutive printer overhauls. Figure 2.6 displays the monitoring process for bead orientation $45°$ while Figure 2.7 shows the case for bead orientation $135°$. In both cases, ST-SSD detects the change at the true change point but raises false alarms. DFLIM immediately detects the defect without raising a false alarm at the build-to-build transition in Figure 2.6. DFLIM still detects the defect in Figure 2.7 but has a delay of 12 images, roughly equivalent to one minute in real time (without considering skipped images).

2.7 Conclusion

In this chapter, we propose a distribution-free monitoring procedure named DFLIM that can address the challenges posed by modern image data, including complex spatial-temporal dependence, non-normality, and high dimensionality with low-dimensional structure. We provide a comprehensive theoretical discussion on the detection ability and the behavior of ARL_0 and ARL_1 for the proposed procedure under reasonable assumptions. Extensive simulations are conducted using various distributions, ranks, and spatial-temporal correlation structures to validate the generality of DFLIM. Additionally, we apply DFLIM to two real datasets, solar flare datasets, and additive manufacturing datasets, to demonstrate its applicability to real-world scenarios.

A future direction is to extend the method to approximately low-rank matrices. While Assumption 2.2 assumes a low-rank in-control mean matrix M_0, in practice, its estimate $\widehat{M_0}$ often deviates from this structure. Significant noise in $\widehat{M_0}$ may lead to overestimating the rank r, introducing extra projected components in y_t and increasing false negatives or false alarms. Analyzing the extent of this deviation could guide a more appropriate selection of r, enhancing DFLIM's performance.

CHAPTER 3
NEURAL NETWORK-BASED CUSUM FOR ONLINE CHANGE-POINT DETECTION

3.1 Overview

Quick detection of unknown changes in the distribution of sequential data, known as change-point detection, is a crucial issue in statistics and machine learning [44, 53, 54, 55]. It is often triggered by an event that may lead to significant consequences, making timely detection critical in minimizing potential losses. For instance, a change-point could arise when production lines become unmanageable. Early detection and remediation by practitioners would result in lesser loss. With the rapidly growing amount of streaming data, the change-point detection applies to practical problems across diverse fields such as gene mapping [56], sensor networks [54], brain imaging [57], and social network analysis [58, 59].

In the modern high-volume and complex data era, across numerous applications, sequential data is collected over networks with higher dimensionality and more complex distributions than before. Therefore, the conventional setting of the change-point detection problem has been surpassed in scope by its modern version. The assumptions in the classical methodologies are possibly too restrictive to tackle the newly occurring challenges, including the high-dimensionality and complex spatial and temporal dependence of sequential data. Such challenges motivate us to introduce neural networks for online change-point detection.

In this chapter, we leverage neural networks to efficiently detect the change-point in the streaming data with high-dimensionality and complex spatial-temporal correlations. The starting point is to convert online change-point detection into a classification problem, of which neural networks are highly capable. Now that the explicit form of the distribution and the log-likelihood ratio are unknown, neural networks benefit us in that the training loss of the neural networks will approximate the underlying true log-likelihood ratio between pre- and post-change distributions. Further, we couple the loss from the neural networks with CUSUM recursion. We summarize the idea as an efficient online training procedure and design extensive simulations and empirical experiments to thoroughly discuss the proposed method against both conventional and state-of-the-art methods.

3.2 Related Works

Various neural network-based methods have been proposed for change-point detection to address the challenges posed by the high dimensionality and complex distributions of modern data. A line of research leverages deep neural network architectures, such as recurrent neural networks [60], autoencoders [61], and convolutional neural networks [62], to capture temporal dynamics and patterns in time series data for change-point detection. However, training models with such deep neural network architectures requires a large amount of data, making them less suitable for online change-point detection.

Likelihood ratio estimation is one of the widely used approaches for change-point detection. Specifically, [63] aim to develop a suitable optimization framework for training neural network-based estimates of the likelihood ratio or its transformations, which can possibly be used for change-point detection and hypothesis testing. Several works utilize neural networks to estimate the likelihood ratio for change-point detection [64, 65].

Detecting changes using statistics generated by online training of neural networks has been explored in several recent studies. [64] introduce online neural network classification (ONNC) and online neural network regression (ONNR). ONNC and ONNR sequentially apply neural networks for classification or regression by comparing the current batch of data with the immediate past batch, which generates a detection statistic with separation of training and testing. ONNC and ONNR can be viewed as Shewhart chart-type methods, as they do not accumulate historical information with limited memory. In contrast, CUSUM-type methods can accumulate all historical information through recursion. It is well known that Shewhart charts may be good at detecting larger changes but can struggle with small changes. In contrast, CUSUM procedures are more sensitive to small changes and enjoy asymptotic optimality properties [66]. Although both ONNC and our proposed method consider the online change-point detection problem as a neural network-based classification problem, they differ in the data utilization, definition of detection statistics, and types of detection procedures.

[67] recently presented a novel *offline* change-point detection method based on training neural networks and a theory that quantifies the error rate for the method. Their work demonstrates competitive empirical performance compared to the standard CUSUM-based classifier for detecting a change in the mean. In contrast, our proposed method aims to detect an abrupt change quickly online from sequential data using the CUSUM procedure, which differs from [67]. Additionally, our numerical experiments aim to study the detection of general types of changes.

3.3 Preliminaries

Consider a sequence of observations $\{x_t \in \mathbb{R}^d : t = 1, 2, \ldots\}$. At some point, there may be a change-point k such that the distribution of the data shifted from f_0 to f_1, which can be formulated as the following composite sequential hypothesis test

$$\begin{aligned} H_0 : &\quad x_1, x_2, \ldots, x_t, \ldots \overset{\text{iid}}{\sim} f_0, \\ H_1 : &\quad x_1, x_2, \ldots, x_k \overset{\text{iid}}{\sim} f_0, \quad x_{k+1}, \ldots x_t, \ldots \overset{\text{iid}}{\sim} f_1, \end{aligned} \tag{3.1}$$

where f_0 and f_1 represent the pre- and post-change distributions, respectively. The expectations under f_0 or f_1 are denoted by $\mathbb{E}_0[\cdot]$ and $\mathbb{E}_1[\cdot]$, respectively. We assume that pre-change data under f_0 are adequate to converge any interested pre-change statistics. We call the pool of pre-change data in the training process as *reference data*, and then the corresponding sequences and samples are called reference sequences and reference samples. This assumption commonly holds since, in practice, the process is mostly pre-change, which allows the sufficient collection of reference data to represent the "normal" state. Another assumption, which can be considered equivalent to sufficient pre-change data up to some estimation error, is that the pre-change distribution f_0 is known.

The classic CUSUM procedure is popular, possibly due to its computationally efficient recursive procedure, and enjoys various optimality properties (e.g., a survey by [55]). CUSUM detects the change using the exact likelihood ratio through the recursion $S_t^{\mathrm{C}} = \left(S_{t-1}^{\mathrm{C}} + \log[f_1(x_t)/f_0(x_t)]\right)^+$, where $S_0^{\mathrm{C}} = 0$, and $(x)^+ = \max\{x, 0\}$. CUSUM procedure raises an alarm at a stopping time $\tau_{\mathrm{C}} = \inf\{t : S_t^{\mathrm{C}} > b\}$, where $b > 0$ is the pre-specified threshold. However, the post-change distribution f_1 is usually unknown in practice. CUSUM may suffer performance degradation when the assumed distributions deviate from the true distributions.

3.4 Neural Network CUSUM

This section proposes the Neural Network (NN)-CUSUM procedure. We consider a neural network function $\varphi(x, \theta) : \mathbb{R}^d \to \mathbb{R}$, parameterized by θ. The network sequentially takes a pre-change distributed reference sequence and an unknown distributed online sequence as inputs.

During training, we convert the change-point detection problem into a binary classification problem. At time index i, we label data from the reference and online sequences with $y_i = 0$ and $y_i = 1$, respectively. Note that the labels are irrelevant to the fact whether the data is from pre-change or post-change distribution. In other words, we do not raise stringent requirements on the knowing of true labels of data as common supervised learning problems. The reference sequences provide information on pre-change distributions so that the classification problem will become non-trivial once the online sequence passes the change point. We feed the labeled data into the neural network and train it using a carefully

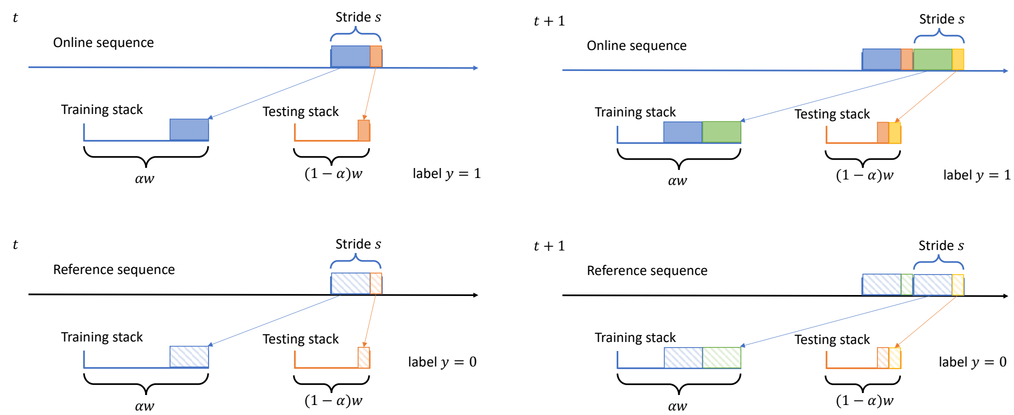


Figure 3.1: Illustration of the NN-CUSUM procedure, running from t (left) to $t+1$ (right). When the newest batch from the streaming data (treated with label $y_i = 1$) is received, it is divided to put into the training stack and the testing stack, respectively (the oldest samples are purged from the stacks to keep a constant stack size). The neural network inherited from the previous t will be updated using a stochastic gradient descent algorithm for *one-pass* through the freshened training stack data. The updated neural network is then used to compute the test statistic for data in the freshened *testing stack* data. The label $y_i = 0$ (from reference data) is constructed completely in parallel. The training/testing stack sizes are hyper-parameters to be tuned in practice to achieve a good *detection performance*.

crafted logistic loss function (such as [68]). Subsequently, the neural network computes the logistic loss for each data point in the reference and online sequences for every time step. We employ the difference between the average logistic function values of the reference and online sequences as the sequential statistics for detecting change-points. Figure 3.1 visualizes the proposed framework.

The stopping time τ is crucial in the sequential hypothesis testing approach for detecting changes in a streaming dataset. It indicates the time to terminate the data stream and declares the occurrence of a distribution shift before time τ. We adopt the CUSUM recursion in our proposed scheme to construct the stopping time but replace the exact log-likelihood ratio with an approximation from a neural network.

3.4.1 Training

The neural network training is conducted via stochastic gradient descent (SGD) updates of the neural network parameter θ on the training stack. As is shown in Figure 3.1, the training stack contains multiple mini-batches of data in $[t-w, t]$ window from the arriving data stream. Within the window, we use a fraction of α samples for training and a fraction of $1-\alpha$ samples for testing. This gives the online training data set $\widetilde{X}_t^{\mathrm{tr}}$ and the online test data set $\widetilde{X}_t^{\mathrm{te}}$, with a size of αw and $(1-\alpha)w$, $\alpha \in (0,1)$, respectively. The window length w and training split fraction α are algorithm parameters.

The sliding window moves forward with a stride s. Half of the stride is put into the training stack, and the other half is put into the testing stack, and the data in the online sequence are labeled 1, and the data in the reference sequence are labeled 0. We update the reference sequence by drawing randomly from reference data.

Recall that $\varphi(x,\theta)$ (to be related to $g_\theta(x)$ later) is the neural network function that maps from $\mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$ and θ is the network parameter. Consider training the neural network using the following general loss, which is computed from the combined training split of arrival and reference samples on the window $D_t^{\mathrm{tr}} := X_t^{\mathrm{tr}} \cup \widetilde{X}_t^{\mathrm{tr}} = \{x_i\}_{i=1}^{2m}$, $m = \alpha w$, labeled by $y_i = 1$ if $x_i \in X_t^{\mathrm{tr}}$ and $y_i = 0$ if $x_i \in \widetilde{X}_t^{\mathrm{tr}}$,

$$\ell(\theta; D_t^{\mathrm{tr}}) = \frac{1}{m}\sum_{i=1}^{2m} l(\varphi(x_i,\theta), y_i), \tag{3.2}$$

The per-sample loss function $l(u,y)$ can be chosen to be different losses:

(1) For the logistic loss, we have

$$l_{\mathrm{logit}}(u,y) = y\log(1+e^{-u}) + (1-y)\log(1+e^{u}). \tag{3.3}$$

The choice of the loss function is motivated by the fact that the functional global minimizer gives the log density ratio: For f_0 and f_1, which are probability densities on $\mathbb{R}^d$, let

$$\ell[\varphi] = \int \log(1+e^{\varphi(x)}) f_0(x)dx + \int \log(1+e^{-\varphi(x)}) f_1(x)dx,$$

which is the population loss, then $\ell[\varphi]$ is minimized at $\varphi^* = \log(f_1/f_0)$. One can verify that the functional $\ell[\varphi]$ is convex with respect to the perturbation in φ. By that

$$\frac{\delta \ell}{\delta \varphi} = \frac{f_0 e^{\varphi} - f_1}{1 + e^{\varphi}},$$

we know $\delta \ell / \delta \varphi$ vanishes when $e^{\varphi} = f_1 / f_0$, that is $\varphi = \varphi^*$, and this is a global minimum of $\ell[\varphi]$. A similar observation has also been made by [63], which shows that training neural networks according to a certain loss lead leads to a log-likelihood function in theory. (2) Apart from the logistic loss, we also consider the ℓ^2-loss, defined as

$$l_2(u, y) = \frac{1}{2} \left[y(u-1)^2 + (1-y)(u+1)^2 \right]. \tag{3.4}$$

(3) The linearized version of the logit loss, motivated by the MMD test statistic [69], as

$$l_{\mathrm{MMD}}(u, y) = -yu + (1-y)u. \tag{3.5}$$

Here, we assume that the SGD training to minimize (3.2) is conducted on D_t^{tr} for each window indexed by t separately from scratch. At time t, the trained network parameter is denoted as $\hat{\theta}_t$, and the trained neural network is $\varphi(\cdot, \hat{\theta}_t)$.

In practical implementation, the neural network parameter θ is initialized at time zero and then updated by SGD (e.g., Adam [70]) through batches of training samples in a sliding training window. Newly arrived data stay in the stack for w/s steps. We trained the network on the training stack in every k step. Thus, each data sample has been used $(w/s)/k$ times for training θ. The number $(w/s)/k$ can be viewed as a hyperparameter indicating an effective number of epochs.

3.4.2 Testing

At each time t, the trained neural network provides a test function $g_{\hat{\theta}_t}(x)$: (1) For logistic and ℓ_2 loss, we set the function $g_{\hat{\theta}_t} = \varphi(\cdot, \hat{\theta}_t)$. (2) For MMD loss, $g_{\hat{\theta}_t}$ is defined as:

$$g_{\hat{\theta}_t}(x) = \frac{1}{\gamma}(\varphi(x, \hat{\theta}_t) - \varphi(x, \theta_0)). \tag{3.6}$$

where the $\gamma > 0$ is the learning rate [69].

To perform online detection, we compute the sample average on the testing windows X_t^{te} and $\widetilde{X}_t^{\text{te}}$ (recall that $\tilde{X}_t \sim f_0$ are drawn sequentially from the reference data):

$$\eta_t := \frac{1}{(1-\alpha)w}\left[\sum_{x \in X_t^{\text{te}}} g_{\hat{\theta}_t}(x) - \sum_{x \in \widetilde{X}_t^{\text{te}}} g_{\hat{\theta}_t}(x)\right]. \tag{3.7}$$

Recall that $(1-\alpha) \in (0,1)$ describes the proportion of the data in the window used for testing, so αw data are used for updating the training stack.

3.4.3 NN-CUSUM Detection Procedure

Starting with $S_0 = 0$, the recursive CUSUM is computed as

$$S_t = \max\{S_{t-1} + \eta_t - D, 0\}, t = 1, 2 \dots \tag{3.8}$$

where $D \geq 0$ is a small constant called the "drift" to ensure that the expected value of the drift term is negative before the change and positive after the change; we will give the condition of the test function that can satisfy this in the following section. For some pre-specified threshold $b > 0$, the detection procedure is specified by the stopping time

$$\tau = \inf\{t : S_t > b\}. \tag{3.9}$$

In practical implementations, for simplicity, our algorithm computes η_t at intervals of s (for "stride"), which is the size of mini-batches of loading the stream data, e.g., $s = 10$; in the extreme case, $s = 2$, since we need half-mini-batch split into stacks. When $s = w$, the sliding window contains completely non-overlapping data. The test statistic S_t is computed with the stride s.

3.5 Experiments

In this section, we perform the comparisons on simulated datasets in Section 3.5.6 and on real datasets in Section 3.5.7. Besides the comparisons, we verify the exponential distribution of τ under f_0 in Section 3.5.2 and introduce some details of the algorithm, including how to calibrate the threshold to control the false alarms (detect a change when there is none) in Section 3.5.3 and numerical studies to investigate the properties of the increments in Section 3.5.4 and the existence of an optimal window in Section 3.5.5. Time complexity analysis among methods is available in Appendix B.1.3.

3.5.1 Simulation Setup

For simulations, including justifications on the proposed method and comparisons with baselines, we will mainly consider two types of shift. One is the Gaussian mean shift, and the other is the Gaussian Mixture Model (GMM) component shift. The concrete settings are

(1) *Gaussian sparse mean shift:* We set $f_0 \sim \mathcal{N}(0, I_d)$ and $f_1 \sim \mathcal{N}(\mu_q, I_d)$, where the shift $\mu_q = (\delta, \delta/2, \delta/3, 0, \cdots, 0)^\top$ with a magnitude $\delta \neq 0$;

(2) *GMM component shift:* We set $f_0 = 1/2\mathcal{N}(\mu\mathbf{1}_d, I_d) + 1/2\mathcal{N}(-\mu\mathbf{1}_d, I_d)$ and $f_1 = p\mathcal{N}(\mu\mathbf{1}_d, I_d) + p\mathcal{N}(-\mu\mathbf{1}_d, I_d) + (1 - 2p)\mathcal{N}(\delta\mathbf{1}_d, \Sigma_q)$ for $\mu > 0$, $|\delta| \neq \mu$ and $p \in (0, 1/2)$, where $\mathbf{1}_d$ is a d-dimensional all-one vector.

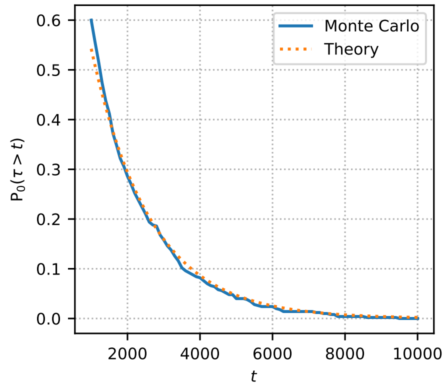


Figure 3.2: The tail probability $\mathbb{P}_0\{\tau > t\}$ versus t. The data consists of 500 sequences with lengths of 40000, generated from i.i.d. Gaussian distribution with a dimension of 100. On this data, we perform NN-CUSUM to get η_ts. The threshold $b = 1$. The solid curve shows numerical values from 500 Monte Carlo repetitions. The dashed curve represents the theoretical values, $e^{-\lambda t}$ with $\lambda = 6 \times 10^{-4}$, which is fitted by non-linear regression on the numerical values.

3.5.2 Verifying Exponential Distribution of τ under f_0

Now we verify that the asymptotic distribution of τ under f_0 is exponential.

Setup In this example, the neural network architecture of NN-CUSUM consists of one fully connected hidden layer of width 64 with a rectified linear unit (ReLU) activation. The last layer is a one-dimensional, fully-connected layer without activation. The training batch size for fixed time t is 100. The stride of the sliding window is 10. We use Adam [70] for training under the logistic loss function with a learning rate of 10^{-3} without scheduling. We set window size $w = 200$ with a split ratio $\alpha = 0.5$. We generate 500 sequences with a total length 40000 consisting of i.i.d. standard Gaussian random variables in $\mathbb{R}^{100}$. This example will have no change point since we study the tail probability of τ under f_0. We set the threshold $b = 1$ in (3.9).

Results In Figure 3.2, the solid curve shows the empirical tail probability of τ from Monte Carlo trials, which we fit with the dashed curve $\mathbb{P}\{\tau > t\} = e^{-\lambda t}$ using non-linear regression to find $\lambda = 6 \times 10^{-4}$. We spot a good match between $e^{-\lambda t}$ and the empirical tail

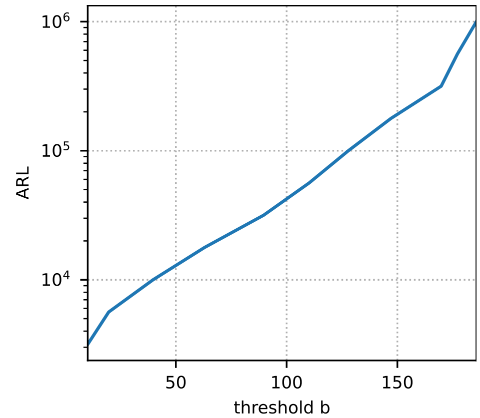


Figure 3.3: Numerically estimate ARL versus threshold b; data is i.i.d. following a Gaussian mixture model $x_i \sim 1/2\mathcal{N}(2\mathbf{1}_d, I_d) + 1/2\mathcal{N}(-2\mathbf{1}_d, I_d)$ where $\mathbf{1}_d$ denotes an all-one vector, and I_d is a d-by-d identity matrix. The experiment consists of 400 sequences of length 500.

probabilities.

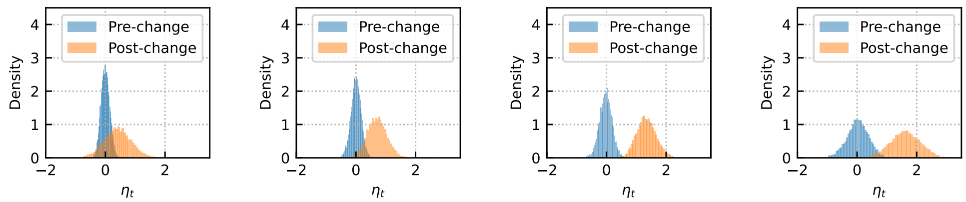
3.5.3 Calibration of Threshold b

It is computationally expensive to use direct simulation of τ to estimate its mean $\mathbb{E}_0[\tau]$. With each fixed threshold b, a large number of repetitions is required, and each repetition takes a long time because we are typically interested in tuning the threshold b to obtain a large ARL when there is no change. Thus, we use a similar technique as in [71] to speed up the simulation to estimate ARL. When the data are i.i.d. f_0 (i.e., before the change-point), the stopping time τ is asymptotically exponentially distributed for some $\lambda > 0$. Then we can estimate for $\mathbb{P}_0\{\tau \geq t\} \approx e^{-\lambda t}$, to further yield an estimate of the rate $\hat{\lambda}$. On the other hand, for the exponential random variable, its mean is equal to the inverse of the rate, and then ARL can be approximated by $\mathbb{E}_0[\tau] \approx 1/\hat{\lambda}$. This way, we can convert simulating τ (which usually takes long runs because we are interested in large ARL when there is no change) into simulating fixed length-t sequences and estimating the probability of such sequences exceeding a threshold b after t.

Figure 3.3 shows ARLs versus the thresholds b on an example consisting of 400 pre-change sequences from GMM and verifies that the logarithm of ARL is approximately

Table 3.1: Mean of pre- and post-change η_t in the Gaussian sparse mean shift example. The standard deviations are in parentheses.

w	Pre-change	Post-change
20	0.001 (0.1441)	0.411 (0.4607)
60	0.003 (0.1685)	0.711 (0.3322)
100	-0.026 (0.2092)	1.337 (0.3315)
200	0.022 (0.3676)	1.621 (0.5112)



(a) $w = 20$ (b) $w = 60$ (c) $w = 100$ (d) $w = 200$

Figure 3.4: Pre- and post-change distributions of η_t in the Gaussian sparse mean shift example.

proportional to threshold b, which is a common property for CUSUM procedures [55].

3.5.4 Properties of Increment η_t

We empirically verify that the training of neural networks will produce nearly zero pre-change η_t and significantly positive post-change η_t.

Setup The neural network setup and the stride of NN-CUSUM are the same with Section 3.5.2, despite of the width of the hidden layer 512 and the batch size of 10. The neural network is trained on the Gaussian sparse mean shift example, namely the first example in Section 3.5.1, where we set $\delta = 0.8$.

We fix the split ratio to be $\alpha = 0.5$, whereas the window sizes slide in $\{20, 60, 100, 200\}$. The data is generated as a sufficiently long sequence with a length 2×10^5 to ensure the stability of the sample average means on η_t. The mean shift happens in the middle, namely at 1×10^5.

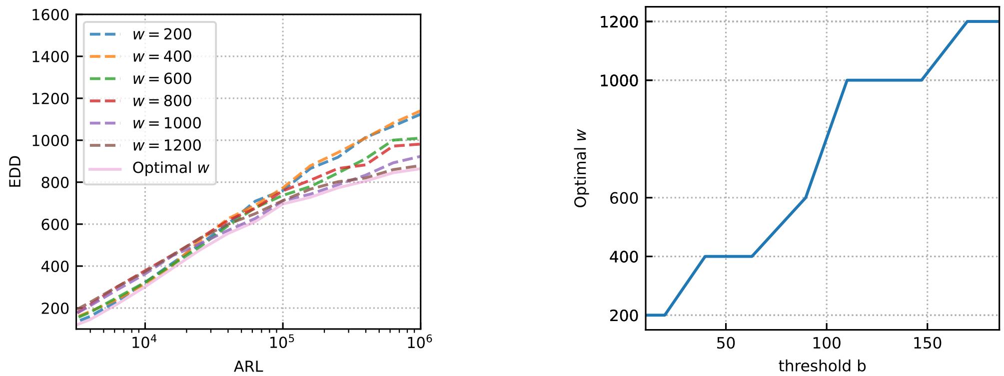
Results Table 3.1 shows the temporal means of pre- and post-change η_t. The post-change expected η_t becomes positive with $w = 20$, and the magnitude is more significant as the window size increases. Figure 3.4 visualizes the empirical distribution of the pre- and post-change η_t under different w, showing that as the sizes of training and testing windows both increase, the mean shift in η_t becomes more evident, representing a possibly faster detection (consistent with Table 3.1). In this example, the NN-CUSUM procedure obtains a positive expected η_t at w as small as 20, which suggests that the method can detect the change point. There is a trade-off in choosing window size w because too large w may delay the detection by sacrificing too many samples before the detection, though it results in an evident drift in η_t which helps with the quick detection.

3.5.5 Optimal Window Size

We empirically investigate the trade-off in choosing the window size w.

Setup The neural network structure, batch size for training at a fixed time, and stride are the same as in Section 3.5.4. The split ratio α takes value in $\{0.1, 0.2, 0.4, 0.8\}$, and the window size w varies in $\{200, 400, \cdots, 1200\}$. The example here is the GMM with a component shift, defined in the second example in Section 3.5.1. To be concrete, we let $\mu = 2$, $p = 1/3$, $\delta = 2.6$ and $\Sigma_q = 0.8I_d + 0.2E$, where E is a d-by-d all-one matrix. 400 sequences are generated, with each length of 10000 and the change point at 5000.

Results In Figure 3.5(a), the dotted curves are the lowest EDD over all α's for every fixed w. The optimal EDD achieves the lower envelope of EDDs over every pair of w and α. The optimal w, to which the lower envelope overlaps with the corresponding curve, increases with the growth of ARLs. Figure 3.5(b) indicates that the optimal w monotonously increases versus a growing threshold b. Figure 3.3 characterizes the monotone relationship between ARLs and bs. The combination of Figure 3.5(b) and 3.3 supports Figure 3.5(a) in terms of the transitions of optimal w against ARLs.



(a) Minimal EDD

(b) Optimal w versus b

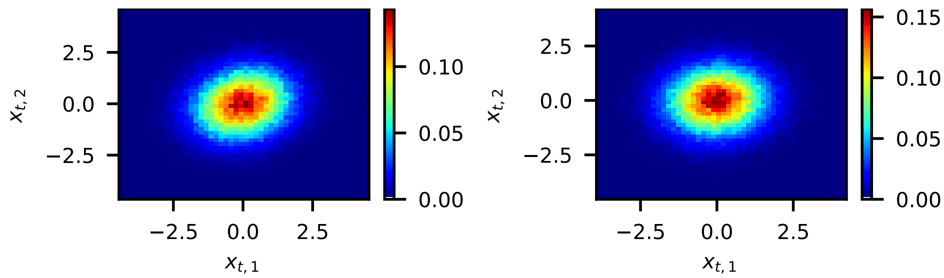
Figure 3.5: Effect of window length w: (a) Minimal ARL versus EDD over window sizes from 200 to 1200 by 200. (b) Optimal window size w versus threshold b. The experiment is conducted on the GMM component shift example in Section 3.5.5.

3.5.6 Comparison on Simulated Data

We perform a numerical comparison of the proposed NN-CUSUM with baselines on a GMM component shift example, in which we set $p = 1/6$, $\mu = 0.4$ and $\delta = 0$ in the second example in Section 3.5.1. The baselines have been introduced in Appendix B.1.1.

Setup In the simulation comparison, the neural network architecture of NN-CUSUM is the same as the example in Section 3.5.2. The selection on the stride and window keeps consistent with Section 3.5. We generate the sequences with a total length 2500 and the change-point located at $k = 500$, namely the pre-change length 500. We calibrate the threshold b to achieve a constant ARL using the ARL calibration in Section 3.5.3. The fixed $\mathrm{ARL} = 5000$, which is adopted later in Table 3.2 will control the Type-I error, which is the probability that the procedure detects a change before the true change point, under 10%.

We generate 400 sequences to perform the detection procedures. Meanwhile, all methods share reference data containing 400 sequences from f_0 with lengths 3500. The detection procedures allow us to learn the statistical behavior of the pre-change process from the reference data. For all NN-based methods, namely NN-CUSUM, ONNC, and ONNR, the



(a) Pre-change density (b) Post-change density

Figure 3.6: Pre- and post-change marginal 2-dimensional distribution densities of x_t in GMM component shift in Section 3.5.1 with dimension $d = 100$, $p = 1/6$, $\mu = 0.4$ and $\delta = 0$.

Table 3.2: For $\mathrm{ARL} = 5000$, EDDs and failure rates on various methods are reported with data generated from for GMM component shift example in Section 3.5.1 with dimension $d = 100$, $p = 1/6$, $\mu = 0.4$ and $\delta = 0$. The smallest EDD and failure rate among the methods except for Exact CUSUM are bold. EDD of Exact CUSUM is lower bound for all methods. The standard deviations are in parentheses. In the table line for EDDs, "–" denotes "failure to detect": the procedure did not raise an alarm by the end of the time horizon 2500 for all sequences, which is equivalent with a failure rate at 100%.

	NN-CUSUM	ONNC	ONNR	WL-GLR	WL-CUSUM	MEWMA	H-CUSUM	Exact-CUSUM
EDD	207.08 (2.02)	1267.4 (38.937)	1523.75 (34.84)	–	–	1638.07 (30.626)	1839.32 (20.387)	2.69 (0.082)
failure rate	0.00%	37.50%	61.50%	100.00%	100.00%	66.75%	81.75%	0.00%

model will train on a burn-in sequence consisting of pre-change data with a length of 1000 to stabilize the neural networks before starting detection.

Tune drift D We tune the drifts D in (3.8) by training neural networks on 500 reference sequences sequentially and computing each sequence's temporal sample means of η_t. In this way, we can have a total of 500 empirical drifts. In the online change-point detection, we deploy the finite sample mean over 500 drifts as D in (3.8).

Data visualization The detection on this GMM component shift is challenging due to the high-dimensionality and subtle change. By setting $\delta = 0$, both of the means of pre- and post-change data are zero. In other words, there is no mean shift. Meanwhile, by setting $\mu = 0.4$, the two modes in the pre-change GMM and the three modes in the post-change GMM will be close to each other, leading to a subtle change in distribution. In Figure 3.6,

Table 3.3: Time complexity for NN-CUSUM and baselines. Assume NN-CUSUM, ONNC and ONNR use one hidden layer neural network and h is the width of the hidden layer. For WL-CUSUM, we consider the worst case under the change in Gaussian distribution, namely MLEs of both mean and variance matrix of online sequence will be computed.

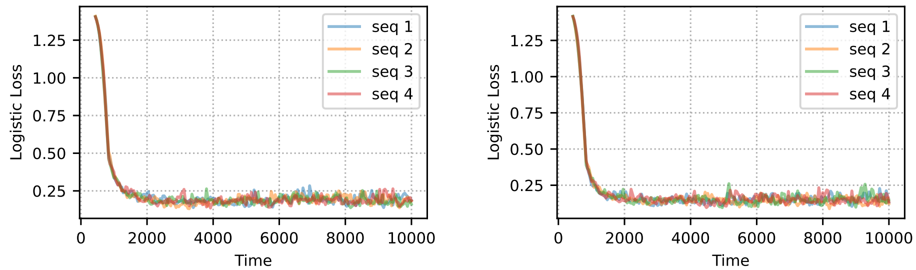
Method	Time complexity
NN-CUSUM	$O(w^2dh/s)$
ONNC/ONNR	$O(w^2dh/s)$
WL-GLR	$O(\max(w,d)wd)$
WL-CUSUM	$O(\max(w,d)d^2)$
MEWMA	$O(d^3)$
H-CUSUM	$O(d^2)$

we visualize the pre- and post-change marginal 2-dimensional distribution densities of x_t in this example, in which the pre- and post-density are similar, but the pre-change density shows inclined pattern. This inclined pattern stems from the two modes in the pre-change distribution, locating at $(0.4, 0.4)$ and $(-0.4, -0.4)$ in the marginal 2-dimensional subspace. When the shift happens, more probability weight is distributed to $(0, 0)$, leading to a more centered post-change density.

Comparisons Table 3.2 shows the comparison between NN-CUSUM and other baselines in the GMM component shift example. The exact-CUSUM provides the theoretical EDD lower bound for methods since it knows both the pre- and post-change distributions. The fact that exact-CUSUM cannot immediately detect, namely an EDD at $2.69 > 1$, numerically indicates that the change is subtle. Despite the subtle change and the high dimensionality, NN-CUSUM can achieve a significantly smaller EDD than other methods, with an advantage over 1000 time indices. Meanwhile, NN-CUSUM can consistently detect the change before the end of time horizon, leading to a failure rate at 0%, whereas other baselines have positive failure rates.

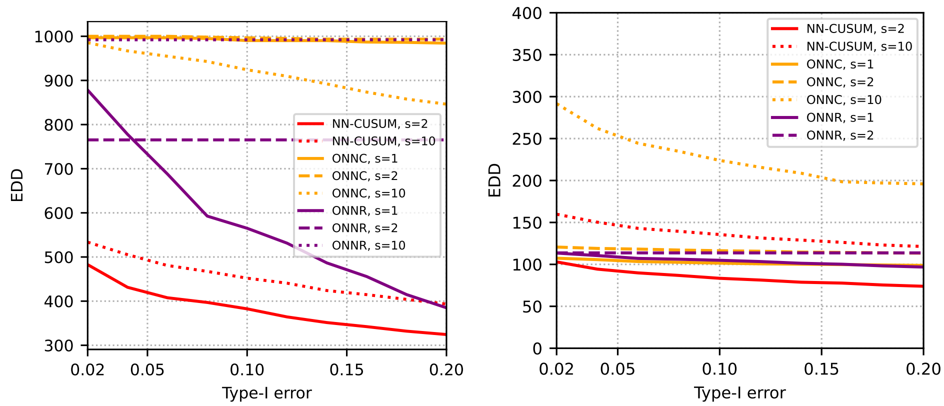
3.5.7 Real data experiments

To examine the performance of NN-CUSUM, we compare the proposed method with various baselines on two real-world datasets: the Higgs boson dataset and the MiniBooNE



(a) Pre-change training logistic loss (b) Post-change training logistic loss

Figure 3.7: (a) Convergence of pre-change training logistic loss over time. (b) Convergence of post-change training logistic loss over time.



(a) (b)

Figure 3.8: EDD versus multiple Type-I errors for NN-CUSUM, ONNC, and ONNR with different choices of the stride of sliding window on (Left) Higgs boson dataset and (Right) MiniBooNE dataset. ONNR with $s = 10$ on MiniBooNE dataset fails to detect in all Type-I errors and is excluded for readability.

dataset. We note that widely used benchmark datasets for change-point detection, such as Beedance [72], Occupancy [73] and HASC [74], are often low-dimensional. Therefore, we specifically select these two datasets to focus on high-dimensional signals.

Dataset Collisions at high-energy particle colliders are useful for discovering rare particles such as Higgs boson. Machine learning approaches often detect such rare particles, which requires solving challenging signal-versus-background classification problems [75]. In our experiment, we convert the Higgs boson detection task into a change-point detection task using the Higgs boson dataset to evaluate NN-CUSUM and baselines. The Higgs boson dataset contains data samples obtained from two types of signals: Higgs boson signal and background signal that does not produce Higgs bosons. The dataset contains about ten million samples produced from Monte Carlo simulations. Each sample comprises 28 numeric features, where 21 features are kinematic properties measured by the particle detectors, and 7 features are functions of the first 21 features, which are high-level features derived from the 21 kinematic features. In our experiment, we use all 28 features.

In another experiment, we consider detecting neutrinos, which is difficult due to their low mass and lack of electrical charge. Mini Booster Neutrino Experiment (MiniBooNE) is designed to search for neutrino oscillations, which provides the MiniBooNE dataset. Similar to the Higgs boson dataset, we establish a change-point detection task using the MiniBooNE dataset. The dataset contains about 130 thousand data samples obtained from background signals or electron neutrinos. Each sample initially contains 50 numeric features. We used 47 features for the experiment after excluding features with excessively high occurrences of particular values. For illustration, we refer to the non-background signal in each dataset as the target signal throughout this section. Both datasets are available at the University of California Irvine machine learning repository [76].

Baselines We compare with baselines: ONNC, ONNR, WL-GLR, WL-CUSUM, MEWMA, and H-CUSUM, whose implementation details are introduced in Appendix B.1.1. Exact CUSUM is excluded since we do not know the true distribution of the real data.

Setup We construct reference and online sequences for each dataset where only online sequences have a change point. We create 500 reference and online sequences using both datasets: one reference sequence contains 1500 data samples from the background signal; one online sequence contains 500 data samples from the background signal and 1000 data samples from the target signal so that change point occurs at time index 500 in online sequence. For the baselines that do not require reference sequences, online sequences are used for experiments.

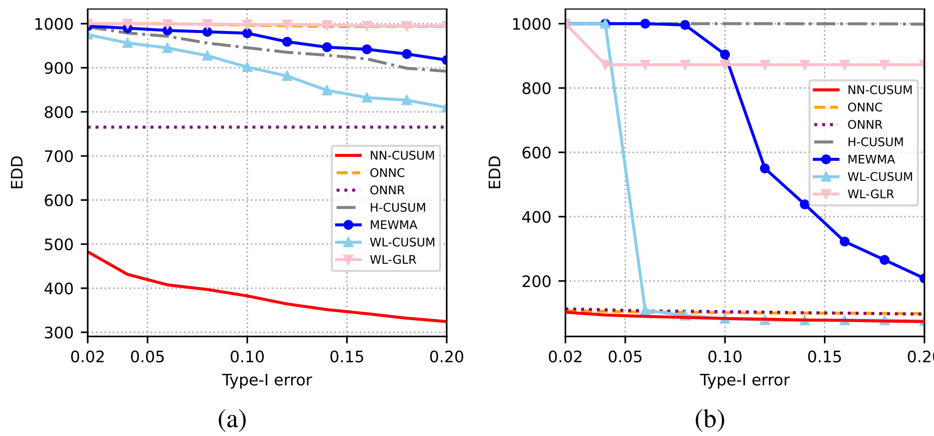


Figure 3.9: EDD versus Type-I errors in the range of $\{0.02, 0.04, \ldots, 0.20\}$ on (Left) Higgs boson dataset and (Right) MiniBooNE dataset. Besides EDDs, detection failure rates are reported in Table 3.4.

For NN-CUSUM, we use fully connected neural networks with one hidden layer of width 64 and ReLU activation. ONNC and ONNR use fully connected neural networks with one hidden layer of width 20 and sigmoid activation following the original publication [64]. The stride is set to 2 for NN-CUSUM, ONNC and ONNR. In the three methods, the stride is the length that the sliding window moving forward at each update of the neural network. The sensitivity analysis regarding the selection of stride on the two datasets is available in Appendix B.1.4. The window size is set to 100 for NN-CUSUM and other baselines that require a window because, in the online setting, the window length has the same meaning over different methods, namely the memory requirement of historical data. Adam is used to train the NN-CUSUM, ONNC, and ONNR neural networks. The learning rate is set to 10^{-3} without scheduling.

Type-I error ARL may not be a good metric for false-alarm control in real data since there is no guarantee on the i.i.d. property of the real data, and thus the calibration of the threshold b in Section 3.5.3 might not be accurate enough. Instead, we use a Type-I error as an alternative to the ARL, which is defined as $\mathbb{P}_0[\tau \leq k]$ representing the probability that a detection procedure raises an alarm before the change point k under f_0. With an

Table 3.4: Detection failure rate with 0.02, 0.10, and 0.20 Type-I error on Higgs boson and MiniBooNE dataset. The lowest detection failure rate (Type-II error) for each fixed Type-I error is in bold.

	Higgs boson			MiniBooNE		
	Type-I Error			Type-I Error		
	0.02	0.10	0.20	0.02	0.10	0.20
NN-CUSUM	**0.04**	**0.01**	**0.01**	**0.00**	**0.00**	**0.00**
ONNC	0.99	0.99	0.98	**0.00**	**0.00**	**0.00**
ONNR	0.84	0.42	0.20	**0.00**	**0.00**	**0.00**
WL-GLR	1.00	0.99	0.99	1.00	0.74	0.74
WL-CUSUM	0.94	0.80	0.62	1.00	**0.00**	**0.00**
MEWMA	0.99	0.85	0.83	1.00	0.82	0.02
H-CUSUM	0.96	0.84	0.73	1.00	1.00	0.99

arbitrarily fixed b, the detection procedure stops before $k+1$ is equivalent with the event $\{\max_{t\leq k} S_t > b\}$. In other words, Type-I errors can be approximated by the empirical frequency of $\{\max_{t\leq k} S_t > b\}$ over sequences.

Results The left panel in Figure 3.9 shows EDD versus multiple Type-I errors for NN-CUSUM and baselines on the Higgs boson dataset. Type-I error takes value in $\{0.02, 0.04, \ldots, 0.18, 0.20\}$. We observe that NN-CUSUM consistently outperforms all other baselines. We also evaluate how often a detection procedure fails to detect the change point by calculating the detection failure rate. The detection failure rate is defined by the frequency of the event $\{\max_{t>k} S_t \leq b\}$ over sequences. Table 3.4 shows the detection failure rate of NN-CUSUM and baselines with respect to Type-I errors in $\{0.02, 0.10, 0.20\}$ on both datasets. NN-CUSUM achieves the lowest detection failure rate.

The right panel in Figure 3.9 shows EDD versus Type-I errors in $\{0.02, 0.04, \ldots, 0.18, 0.20\}$ on the MiniBooNE data. NN-CUSUM has the smallest EDDs over all Type-I errors. WL-CUSUM shows comparable EDD with NN-CUSUM in Type-I errors higher than 0.06. In terms of the failure rate, NN-CUSUM, ONNC, and ONNR consistently achieve the zero detection failure rate as shown in Table 3.4. WL-CUSUM totally fails to detect with low Type-I errors. Sampled trajectories of the detection statistics for NN-CUSUM, ONNC, and ONNR on both datasets are provided in Appendix B.1.5.

3.6 Conclusion

We present a novel neural network-based procedure for online change-point detection using CUSUM recursion with increments obtained from a trained neural network binary classifier applied on test batches. The motivation is that the training of a logistic loss will converge in population to a log-likelihood ratio between two samples, which motivates us to construct CUSUM statistics using the sequentially learned neural networks to test samples. We demonstrate the competitive performance of the proposed method compared to other classical statistical and neural network-based methods with extensive numerical experiments.

CHAPTER 4
HIGHER-CRITICISM FOR SPARSE MULTI-STREAM CHANGE-POINT DETECTION

4.1 Overview

Sequential change-point detection, which aims to identify changes in the distribution of streaming data as quickly as possible, is a fundamental challenge in statistics [55]. Recently, there has been much interest in the settings where a large number of data streams are being monitored and the potential change may affect only a small subset of variables, a scenario referred to as *sparse* multi-stream change-point detection [77, 71, 78, 79, 80]. The challenges in this situation follow from the unknown identity of the affected streams if they exist, their relatively small number, and the standard sequential change-point trade-offs of minimizing *detection delay* while maintaining the longest possible run without false positives under the null.

In offline hypothesis testing, Higher Criticism (HC) [81] has proven to be an effective tool for global testing against sparse alternatives [82, 83, 84, 85]. The relative simplicity of HC, its well-understood properties, and its inherent feature selection mechanism [86] have made HC a popular choice for sparse signal detection among practitioners [83, 87]. Therefore, we aim to develop an adaptation of HC for sparse sequential change-point detection.

In this chapter, we present a new procedure for detecting a change point affecting a small and unknown subset of data streams by combining their detection statistics using HC. Our procedure can be viewed as a sequential adaptation of offline sparse signal detection using HC: in every time instance, we test the significance of a change in every stream individually using one of the known tests, such as the cumulative sum (CUSUM) test, then use HC to combine P-values from many individual change-point tests. The resulting statistic is sensitive to sparse changes. Namely, changes affecting only a small fraction of the streams when the locations of the affected streams are unknown. Additionally, HC provides a mechanism to identify streams suspected to be affected, offering optimality properties compared to other fault localization methods based on feature selection methods such as those based on false discovery rate controlling [86, 82]. Since the procedure wraps around P-values, it can be naturally applied to data streams of heterogeneous pre- and post-change distributions in individual streams with general change-point detection statistics. We demonstrate the effectiveness of the HC-based procedure through extensive numerical evaluations and provide a theoretical analysis under a sparse multi-stream model with normal data, where individual streams may experience changes in their means and variances, referred to as the heteroscedastic change. We show that the expected detection delay of HC of P-values obtained from the likelihood ratio (LR) based CUSUM [88], or from the generalized likelihood ratio (GLR) test [89], attains the information-theoretic limits of detection when only the mean is affected (i.e., the homoscedastic change) derived in [78]. Our results show that the HC of P-values from individual streams is also optimally adaptive to the sparsity level. Our key contributions are summarized as follows.

- We propose a *sequential* change-point detection method based on HC, effectively detecting a global change point with sensitivity to a change affecting a few streams simultaneously out of many monitored streams.

- We consider a multi-stream normal sparse heteroscedastic change point model and characterize the information-theoretic detection delay in this model for individual tests based on the LR and the GLR statistics. We deliver such results concerning the detection delay rather than the expected detection delay studied in [78].

- We show that the HC-based procedure attains the asymptotic minimal detection delay in the model mentioned above, even when the sparsity and post-change mean and variance are unknown.

- We use extensive simulations to compare the detection procedure based on HC to other procedures proposed in the literature for the same setting in finite sample sizes. Our evaluations show that the HC-based procedure typically outperforms other methods.

Our procedure may also benefit from other well-known properties of HC, such as robustness to mild dependency among the P-values [90, 91] and a known procedure for adapting to stronger dependencies [92].

To the best of our knowledge, this chapter is the first to study the use of higher criticism statistics for combining P-values to detect sparse changes sequentially. Related recent works focusing on offline change point and anomaly detection in multi-stream data include [93, 94, 95, 96]. Methods for sparse sequential change-point detection include tests based on the stream providing the strongest indication [97, 79], combining local tests (typically, CUSUM-based) via averaging or other methods [98, 77, 99, 100], kernel-based scan statistics [99], and a mixture of likelihood ratios known to be useful when the number of affected streams is known [71]. Methods with proven optimal adaptivity to the sparsity level include the mixture likelihood ratio with adaptive window size of [78], and combining P-values from individual streams using likelihood scores [80].

The rest of the paper is organized as follows. Section 4.2.1 sets up the data analysis problem. Section 4.2 presents the HC procedure for sequential change-point detection. Section 4.3 contains the main theoretical results. Section 4.4 contains numerical examples that validate the theoretical results and compare HC's performance with other methods. Section 4.6 summarizes the paper. Section 4.5 contains proofs. Some proof details are delegated to the appendix.

4.2 Method

4.2.1 Setup

We observe N data streams over time. Let $X_{n,t}$ represent the observation of the n-th stream at time $t = 1, 2, \ldots$. The null hypothesis states no change in the distribution of any stream. Under the alternative hypothesis, there exists an *unknown* time τ, $0 < \tau < \infty$, and an *unknown* subset of streams with index set $I \subset \{1, \ldots, N\}$ affected by the change at time $t = \tau$ and thus their distribution is different at times $t = \tau, \tau + 1, \ldots$. In contrast, the distribution of each unaffected stream remains unchanged. Formally, given sequences of pre-change distributions $\{F_n^0\}_{n=1}^N$ and post-change distributions $\{F_n^1\}_{n=1}^N$, we consider the following sequential hypothesis testing problem:

$$\begin{aligned} H_0: &\quad X_{n,t} \sim F_n^0, \quad t = 1, 2, \ldots \\ H_1: &\quad X_{n,t} \sim \begin{cases} F_n^0, & n \notin I, \\ F_n^0, & n \in I, \quad t = 1, \ldots, \tau - 1, \\ F_n^1, & n \in I, \quad t = \tau, \tau + 1, \ldots. \end{cases} \end{aligned} \tag{4.1}$$

Our goal is to detect the change as quickly as possible after it occurs while allowing a sufficiently long run without a false alarm when there is no change. We focus on the case when the fraction of affected streams $p := |I|/N$ is small. Our approach involves testing for a change in every stream individually and combining multiple tests using *higher-criticism* [81, 83].

4.2.2 Change-point testing in individual streams

Given the observations at time t, suppose we use certain detection statistics $Y_{n,t}$ that is a function of $\{X_{n,s}\}_{s \le t}$ for sequential change-point detection at the n-th stream. We summarize the significance of the change by a P-value with respect to the no-change distribution

F_n^0. Let $\{\pi_{n,t}\}$, $n = 1, \ldots, N$ be a sequence of such P-values at times $t = 1, 2 \ldots$.

4.2.3 Detection using Higher Criticism

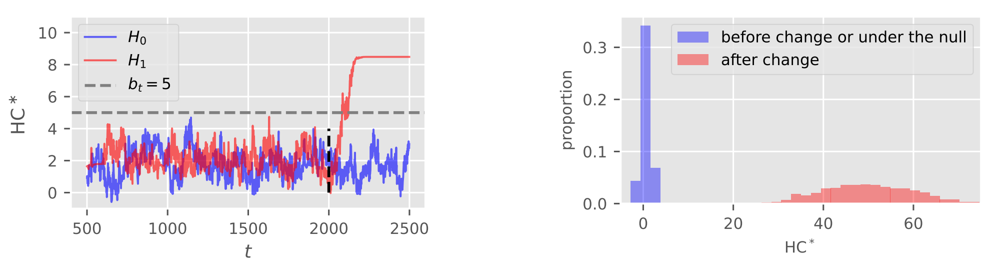
Before presenting the change point detection procedure, let us review the HC statistic [81, 83]. For a given collection of P-values $\pi_1, \ldots, \pi_N$, let $\pi_{(1)} \leq \cdots \leq \pi_{(N)}$ be their ordered version. The HC statistic of a collection of P-values is defined as

$$\mathrm{HC}^*(\pi_1, \ldots, \pi_N) := \max_{1 \leq n \leq \alpha_0 N} \sqrt{N} \frac{\frac{n}{N} - \pi_{(n)}}{\sqrt{\frac{n}{N}\left(1 - \frac{n}{N}\right)}}, \tag{4.2}$$

where $\alpha_0 \in (0, 1)$ is a tunable parameter known to have little effect on the large sample behavior of HC^* [83].

To apply HC in the sequential setting, we form the detection statistic at time t using individual P-values at each data stream:

$$\mathrm{HC}_t^* = \mathrm{HC}^*(\pi_{1,t}, \ldots, \pi_{N,t}), \qquad t = 1, 2 \ldots. \tag{4.3}$$



(a) (b)

Figure 4.1: HC detection statistic HC_t^* of (4.3) before and after the change that occurs at time $t = 2000$. Each of $N = 500$ streams is distributed as $\mathcal{N}(0, 1)$ before the change, after the change, streams in I are distributed as $\mathcal{N}(\mu_r(N), 1)$. P-values of each stream at each time are computed using the CUSUM statistic (4.7) under the null distribution (before the change). (a) Trajectories of HC_t^* over t, when there is no change, and when there is a change at 2000, post-change distribution being normally distributed with $\mu_r(N) = 0.35$ and $|I| = 4$. (b) Histograms of 1000 simulated values of HC_t^* at time $t = 1,000$ (before the change) and $t = 2,100$ (after the change), with $\mu_r(N) = 0.79$ and $|I| = 22$.

We declare global change when the sequence $\{\mathrm{HC}_t^*\}_{t=1,2,\ldots}$ exceeds a prescribed threshold sequence $\{b_t\}_{t=1,2,\ldots}$ for the first time. This detection procedure corresponds to the following stopping rule called the *HC procedure*:

$$T_{\mathrm{HC}} = \inf\{t > 0 : \mathrm{HC}_t^* > b_t\}, \tag{4.4}$$

where $b_t > 0$ is a user-specified (usually time-invariant) threshold chosen to control the false alarm when there is no change, and thus control the run length (RL) and the average run length (ARL). Figure 4.1(a) presents an example of the detection statistic and its empirical distributions before and after the change.

The HC procedure can perform change localization conveniently, once a change has been detected. When the detection statistic has exceeded the threshold, thus, a global change is declared, we may select streams suspected to experience a change, by setting

$$n^* = \underset{1 \le n \le \alpha_0 N}{\arg\max} \frac{\frac{n}{N} - \pi_{(n)}}{\sqrt{\frac{n}{N}\left(1 - \frac{n}{N}\right)}}, \tag{4.5}$$

and select the n-th stream if $\pi_i \le \pi_{(n^*)}$. Prior works [86, 82, 83, 101] discuss this selection mechanism and its optimality properties in the context of feature selection for classification.

4.3 Asymptotic detection performance in normal data

In this section, we analyze the asymptotic performance of the HC procedure when detecting a sparse change in the mean across independent normal data streams. The asymptotic setting assumes that the number of streams N goes to infinity while the other problem parameters, such as the average proportion of affected streams and the magnitude of an individual mean change, are calibrated to N.

Our calibration follows that of (Chan, 2017) [78], which has been shown to yield a challenging and interesting statistical setup: There exists an “information-theoretic” lower bound in the sense that any detection procedure with a controllable false alarm rate has an expected delay exceeding this lower bound in the sparse change setting. The main results of this section state that the higher criticism procedure attains this information-theoretic bound. Additionally, we extend the setting of [78] to include potential change in the variance of affected streams, detection based on the generalized likelihood ratio statistics [89], and convergence in distribution results rather than convergence of the means as studied in [78].

4.3.1 Problem formulation

While our procedure works for general distributions, for theoretical analysis, we make some additional distributional assumptions in addition to (4.1). Let $I \subset \{1, \ldots, N\}$ be a random set of indices such that $\Pr[n \in I] = p$ independently across $n = 1, \ldots, N$. The observations obey $X_{n,t} \sim \mathcal{N}(0, 1)$ for all t and $n \notin I$, and for $n \in I$,

$$X_{n,t} \sim \begin{cases} \mathcal{N}(0,1), & t = 1 \ldots, \tau - 1, \\ \mathcal{N}(\mu, \sigma^2), & t = \tau, \ldots, \mathcal{T}, \end{cases} \tag{4.6a}$$

where the paths $\{X_{n,t}\}_{t \geq 1}$ are independent for different $n = 1, \ldots, N$, and $\mathcal{T}$ denotes the time horizon. The global change point hypothesis testing can be cast as:

$$\begin{aligned} H_0^{(N)} &: \ \tau > \mathcal{T}, \\ H_1^{(N)} &: \ \tau \leq \mathcal{T}, \end{aligned} \tag{4.6b}$$

i.e. $\tau > \mathcal{T}$ indicates no change. Here, for simplicity, we assume that in (4.6a), $\mu > 0$ is the common shift in the means of all affected streams and $\sigma > 0$ is associated with a common change in their variances. The homoscedastic case ($\sigma = 1$) corresponds to the setting of [93, 78, 80].

4.3.2 Testing individual streams

We consider two useful cases of test statistics in individual streams.

Case I: CUSUM for mean-shift (see, e.g., (Lorden 1971) [88]). Consider the CUSUM statistic formed on each stream assuming the known post-change mean:

$$Y_{n,t}^{\mathsf{LR}} := \max_{k \le t} V_{n,t,k}, \qquad V_{n,t,k} := \left(S_{n,t} - S_{n,k} - \frac{\mu}{2}(t-k)\right)\mu, \tag{4.7}$$

where $S_{n,t} := \sum_{j=1}^{t} X_{n,j}$, and μ is the assumed post-change mean. Note that $Y_{n,t}^{\mathsf{LR}}$ is associated with the likelihood ratio statistic for change in the mean of a normal model from zero to $\mu_N(r)$ while the variance is unchanged. This statistic arises in testing for a mean shift by μ under a local asymptotically normal family; see the discussions in [102, 55] and references therein. The CUSUM $Y_{n,t}^{\mathsf{LR}}$ can be computed recursively in time t, facilitating its online implementation and thus a popular choice for online change-point detection. A P-value $\pi_{n,t}^{\mathsf{LR}}(x)$ for the n-th stream corresponds to the survival function of $Y_{n,t}^{\mathsf{LR}}$, as in

$$\pi_{n,t}^{\mathsf{LR}}(x) := \Pr\left[Y_{n,t}^{\mathsf{LR}} \ge x \mid H_0\right]. \tag{4.8}$$

Case II: Generalized Likelihood Ratio (GLR). The GLR statistic to detect some change in the mean (see, e.g., (Siegmund and Venkatraman 1995) [89]) is defined by

$$Y_{n,t}^{\mathsf{GLR}} := \max_{t-w<k\le t} W_{n,t,k}, \qquad W_{n,t,k} := \frac{|S_{n,t} - S_{n,k}|}{\sqrt{t-k}}. \tag{4.9}$$

where $w > 0$ is the window-length; this corresponds to the window-limited GLR [103]. The GLR statistic cannot be computed recursively, unlike the CUSUM statistic in (4.7). On the other hand, GLR does not require assuming the post-change parameters, which are typically hard to specify a priori, and the true change magnitude can change from instance to instance. The window length is typically chosen to be large enough so that it does not affect the detection delay. This statistic is particularly useful when the post-change mean is unknown. The P-value at time t is given by $Y_{n,t}^{\mathsf{GLR}}$ by

$$\pi_{n,t}^{\mathsf{GLR}}(x) := \Pr\left[Y_{n,t}^{\mathsf{GLR}} \geq x \mid H_0\right]. \tag{4.10}$$

We have a few remarks:

- Statistics based on the cumulative sum $S_{n,t}$ such as $Y_{n,t}^{\mathsf{LR}}$ and $Y_{n,t}^{\mathsf{GLR}}$ are widely used in the literature for detecting a change in the mean. They are generally less effective for detecting a particularly small or large change in the variance, hence it is useful to think about σ^2 in our model (4.6a) as a nuisance parameter. As it turns out, larger values of σ^2 are asymptotically *favorable* for detecting using $Y_{n,t}^{\mathsf{LR}}$ and $Y_{n,t}^{\mathsf{GLR}}$.

- It is also possible and sometimes helpful to define the test statistics $V_{n,t}$ and $W_{n,t}$ by replacing $S_{n,t} - S_{n,k}$ with $\max\{S_{n,t} - S_{n,k}, 0\}$. Such replacement does not appear to change the main conclusions concerning asymptotic detection delays.

- In either case $\square \in \{\mathsf{LR}, \mathsf{GLR}\}$ and under either hypothesis, $Y_{n,t}^{\square}$ and thus $\pi_t^{\square}(Y_{n,t})$ have continuous distributions, hence $\pi_{n,t}^{\square}(Y_{n,t})$ is uniformly distributed over $(0,1)$ under H_0.

4.3.3 Asymptotic detection delay

We study the detection procedure (4.4) under an asymptotic setting of $N \to \infty$, while p and μ are calibrated to N. Specifically, we calibrate p as

$$p = N^{-\beta}, \tag{4.11a}$$

where $\beta \in (0,1)$ is a parameter controlling sparsity, and

$$\mu := \mu_r(N) := \sqrt{2r\log(N)}, \tag{4.11b}$$

$r > 0$ is a fixed parameter controlling the magnitude of the mean shift in affected streams. Under this calibration, (4.6a) defines a sequence of hypothesis testing problems indexed by N.

We characterize the asymptotic detection delay of procedures based on P-values obtained either from Y_t^{LR} or Y_t^{GLR}. Define the function

$$\Delta^*(r, \beta, \sigma) := \lceil \rho^*(\beta, \sigma)/r \rceil, \tag{4.12}$$

(the smallest integer larger than $\rho^*(\beta, \sigma)/r$) where

$$\rho^*(\beta, \sigma) := \begin{cases} (2-\sigma^2)(\beta - 1/2), & \frac{1}{2} < \beta < 1 - \frac{\sigma^2}{4}, \quad 0 < \sigma^2 < 2, \\ \left(1 - \sigma\sqrt{1-\beta}\right)^2, & 1 - \frac{\sigma^2}{4} \leq \beta < 1, \quad 0 < \sigma^2 < 2, \\ 0, & \frac{1}{2} < \beta < 1 - \frac{1}{\sigma^2}, \quad \sigma^2 \geq 2, \\ \left(1 - \sigma\sqrt{1-\beta}\right)^2, & 1 - \frac{1}{\sigma^2} \leq \beta < 1, \quad \sigma^2 \geq 2. \end{cases}$$

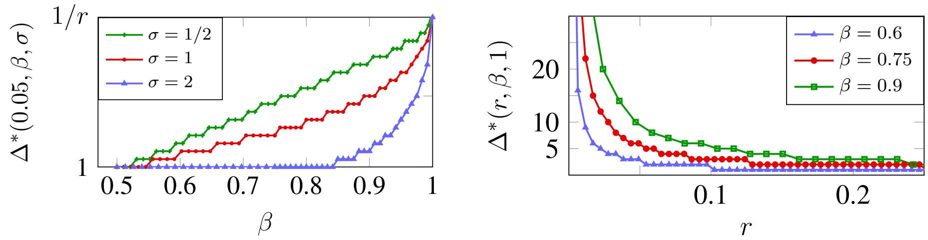
The function $\rho^*(\beta, \sigma)$ was first introduced in [104] for describing the detection boundary in heteroscedastic sparse signal detection. Here, we will show that $\Delta^*(r, \beta, \sigma)$ characterize the asymptotic detection delay; Figure 4.2 illustrates $\Delta^*(r, \beta, \sigma)$ for several varying parameter values.

Our first main result says that for detection procedures based on LR or GLR tests, the probability of detecting at times $t < \tau + \Delta^*(r, \beta, \sigma)$ is asymptotically identical under the null and the alternative. This result implies that the detection delay allowing for a meaningful change-point monitoring procedure is bounded below by $\Delta^*(r, \beta, \sigma)$.

Theorem 4.3.1. *Consider the change-point detection problem* (4.6) *and P-values*

$$\pi_{n,t} := \pi_t^{\square}(Y_{n,t}^{\square}), \quad n = 1, \ldots, N, \quad t = 1, \ldots, \mathcal{T},$$

where $\square \in \{\mathsf{LR}, \mathsf{GLR}\}$ *as defined by* (4.7)-(4.8) *or* (4.9)-(4.10). *For a given test statistic*



(a) (b)

Figure 4.2: Curves describing the asymptotic theoretical detection delay $\Delta^*(r,\beta,\sigma)$ of (4.12) in multistream normal data with a heteroscedastic sparse change for detection based on LR or GLR statistics (Theorems 4.3.1 and 4.3.2). (a): $\Delta^*(r,\beta,\sigma)$ versus β for several values of σ. (b): $\Delta^*(r,\beta,\sigma)$ versus r for several values of β. The detection delay increases with larger β (higher sparsity) and decreases with larger r (stronger change magnitude) and larger σ^2 (larger post-change variance).

U_t based on $\pi_{1,t},\ldots,\pi_{N,t}$, consider a detection procedure that stops at time T_U as soon as U_t exceeds $b_t^{(N)}$. Consider an asymptotic setting $N\to\infty$, with p and μ calibrated to N as in (4.11) *and $\mathcal{T}/\log(N)\to 0$. If $t<\tau+\Delta^*(r,\beta,\sigma)$, for any array of thresholds $\{b_t^{(N)},\ t=1,\ldots,\mathcal{T},\ N=1,2,\ldots\}$ we have*

$$\Pr\left[T_U \le t \mid H_1^{(N)}\right] - \Pr\left[T_U \le t \mid H_0^{(N)}\right] \to 0.$$

In other words, Theorem 4.3.1 says that whenever there exists a procedure for detecting a change based on the LR or GLR statistics with a probability of true detection at time t asymptotically exceeding the probability of a false alarm at time t, then $t\ge\Delta^*(r,\beta,\sigma)$.

The following result shows that it is possible to detect using higher criticism with a delay converging to $\Delta^*(r,\beta,\sigma)$.

Theorem 4.3.2. *Consider the change-point detection problem* (4.6a) *and P-values*

$$\pi_{n,t} := \pi_t^{\square}(Y_{n,t}^{\square}),\quad n=1,\ldots,N,\quad t=1,\ldots,\mathcal{T},$$

where $\square\in\{\mathsf{LR},\mathsf{GLR}\}$ as defined by (4.7)-(4.8) *or* (4.9)-(4.10). *Let the detection procedure* *stop at time T_{HC} as soon as $\mathrm{HC}^*_t = \mathrm{HC}^*(\{\pi^{\square}_{n,t}\}_{n=1}^N)$ exceeds b_t:*

$$T_{\mathrm{HC}} = \inf\{t : \mathrm{HC}^*_t > b_t\}.$$

In an asymptotic setting $N \to \infty$, with p and μ calibrated to N as in (4.11) *and $\mathcal{T}/\log(N) \to 0$, for any integer $\Delta \geq \Delta^*(r, \beta, \sigma)$, there exists an array of thresholds $\{b_t^{(N)},\ t = 1, \ldots, \mathcal{T},\ N = 1, 2, \ldots\}$ such that,*

$$\Pr\left[T_{\mathrm{HC}} - \tau = \Delta \mid H_1^{(N)}\right] \to 1,$$

while

$$\Pr\left[T_{\mathrm{HC}} \leq \mathcal{T} \mid H_0^{(N)}\right] \to 0.$$

To state the following corollary, we first define a false alarm rate α for a finite time horizon $\mathcal{T}$, as $\Pr\left[T_{\mathrm{HC}} \leq \mathcal{T} \mid H_0^{(N)}\right] \leq \alpha$.

Corollary 4.3.3. *Under the setting and conditions of Theorem 4.3.2, for any (β, σ, r) configuration such that $\rho^*(\beta, \sigma)/r$ is not an integer, and for any false alarm rate $\alpha \in (0, 1)$ in the horizon $\mathcal{T}$, there exists an array of threshold $b_t^{(N)}$, which controls the false alarm at level α, such that the detection delay $T_{\mathrm{HC}} - \tau$ converges to $\Delta^*(r, \beta, \sigma)$ in distribution uniformly in τ. In particular,*

$$\max_{1 \leq \tau \leq \mathcal{T}} \mathbb{E}_\tau \left[T_{\mathrm{HC}} - \tau \mid T_{\mathrm{HC}} \geq \tau\right] = \Delta^*(r, \beta, \sigma). \tag{4.13}$$

4.3.4 Discussions

Information-theoretic delay and phase transition

Theorem 4.3.1 establishes a fundamental limit on the ability to construct a meaningful detection procedure for delays $\Delta < \Delta^*(r, \beta, \sigma)$, since the probability of a false alarm before such Δ converges to the probability of true detection before Δ. This renders the detection delay comparable in magnitude to the in-control run length, effectively reducing detection to a random guess and, therefore, making it uninformative. On the other hand, Theorem 4.3.2 shows that there exist detection procedures with detection delays converging in distribution (from above) to $\Delta^*(r, \beta, \sigma)$ while maintaining a stretch of no false alarms $\mathcal{T}$ going to infinity. One such procedure is the higher criticism-based procedure of (4.4). This describes a phase transition in the ability to provide meaningful change-point monitoring: asymptotically impossible for delays smaller than $\Delta^*(r, \beta, \sigma)$ and asymptotically possible for larger delays. Previous results addressing the information-theoretic detection delay focused on the *expected* detection delay (EDD) subject to a large ARL [78, 80], in which case the aforementioned phase transition when applied to a well-calibrated HC procedure takes the following form. An "undetectable" region $t < \Delta^*(r, \beta, \sigma)$ of linear increase of the EDD with the ARL, and a "detectable" region $t > \Delta^*(r, \beta, \sigma)$ of a very moderate increase of the EDD with a significant increase in ARL. Numerical evaluations in Figure 4.7 below illustrate the two regimes and suggest that the increase in the EDD is logarithmic in the ARL in the detectable region. The logarithmic increase is typical in sequential change-point analysis [105].

The restriction to monotonic statistics in Theorem 4.3.1 is sufficient from a practitioner's perspective when seeking a detection procedure that is monotonic in the magnitude of a change gathered across individual streams. However, we would like to remark that a simple extension of similar results in [104] and [106] appear to yield a lower bound not restricted to such statistics, and we leave such extensions to future work.

Relation to previous results

Typical results in change point detection literature focus on expectations of the stopping time, such as the ARL, $\mathbb{E}\left[T \mid H_0\right]$, and the worst-case EDD, defined by [107] as $\max_{\tau \geq 1} \mathbb{E}_\tau[T - \tau \mid T \geq \tau]$ when a change occurs at time τ. We address the worst-case EDD in Corollary 4.3.3.

The work of (Chan, 2017) [78] provided the asymptotic EDD under a setting similar to (4.6) with $\sigma = 1$ and subject to an ARL lower bound γ that increases in N. The EDD therein is identical to the one provided by T_{HC} in Theorem 4.3.2, once we adjust the calibration of the change magnitude and the scaling in the ARL lower bound γ of [78] to our setup. Consequently, in this case, T_{HC} attains the optimal detection delay under the change detection model (4.6).

Our analysis limits the growth rate of the horizon $\mathcal{T}$ in N and, therefore, our false alarm run length guarantee and its expected value, i.e., the ARL. This is a restriction compared to [78] that characterized the tradeoff among β, r, and the ARL lower bound γ under the scaling $\log(\gamma) \sim N^{\zeta}$ for $\zeta > 0$. Note that our setup corresponds to the case $\zeta \to 0$ in [78]; the minimal asymptotic detection delay $\Delta^*(r, \beta, \sigma)$ coincides with that provided in [78] for $\zeta \to 0$.

Under special cases, the work of (Liu et al. 2021) [108] also delivers asymptotic interplays among the number of streams N, the control duration $\mathcal{T}$, the magnitude of a change, and the sparsity level. However, their asymptotic scaling is different than ours and therefore incomparable. Specifically, in the case of severe sparsity $\beta \in (1/2, 1)$, the setting of [108] does not lead to a phase transition. On the other hand, it provides interesting interplays for a relatively dense change $\beta \in (0, 1/2)$, whereas detection is trivial in this regime in our setting.

Heteroscedastic change

An increase in the post-change variance σ^2 reduces $\Delta^*(r, \beta, \sigma)$ and thus leads to a quicker detection based on LR and GLR statistics. This situation is well-understood in the context of offline signal detection in [104] and [109]. When σ^2 is dramatically smaller or larger than 1, statistics based on the cumulative sum such as $Y_{n,t}^{\mathsf{LR}}$ and $Y_{n,t}^{\mathsf{GLR}}$ may be much less effective compared to alternatives statistics. For this reason, it is useful to consider σ as a nuisance parameter in our setting. In particular, the analysis and its conclusions may change dramatically when σ is permitted to increase or vanish with N.

4.4 Numerical experiments

This section presents numerical experiments to validate theoretical results and compare the proposed procedure with existing methods. Throughout, we consider data sampled from the model (4.6a) with $\mu = \sqrt{2r \log N}$, $\sigma^2 = 1$, and $p = \Pr[n \in I] = N^{-\beta}$. Namely, normal pre- and post-change distributions (4.6a) with $\sigma = 1$, μ and p are parametrized using N, r, and β as in (4.11). The HC hyperparameter is fixed at $\alpha_0 = 0.2$ based on the recommendations in (Donoho and Jin, 2015) [83].

4.4.1 Trajectory illustration

We start by illustrating the detection procedure when there is no change versus when there is a change. Figure 4.1(a) shows the trajectory $\{\mathrm{HC}_t^*\}$ of (4.3) over t under H_0 (no change) and another one under H_1 (when the change happens at 2000). Clearly, in this setting, by setting a threshold $b_t = 5$, one can successfully detect the change. Figure 4.1(b) shows the empirical distribution of simulated values of HC_t^* before and after the change. It is worth mentioning that the empirical distribution appears to be stable after a "burn-in" period of about $t = 200$ instances. We only consider in our experiments the detection statistics after the burn-in period.

4.4.2 Calibration of threshold

To estimate the threshold $b_t = b$ for controlling ARL under H_0, we employ an indirect Monte Carlo simulation technique used in (Xie and Siegmund 2013) [71], to reduce computational cost when calibrating the threshold for large target ARL values. This approach is based on the observation that the stopping time under the null hypothesis is approximately exponentially distributed, i.e., $\Pr[T_{\mathrm{HC}} \geq t \mid H_0] \approx e^{-\lambda t}$, which can be argued by large-deviation theoretically. Thus, we first estimate $\Pr[T_{\mathrm{HC}} \geq t \mid H_0]$ to obtain $\hat{\lambda}$ by simulating fixed-length sequences, then use the relationship between the probability of hitting

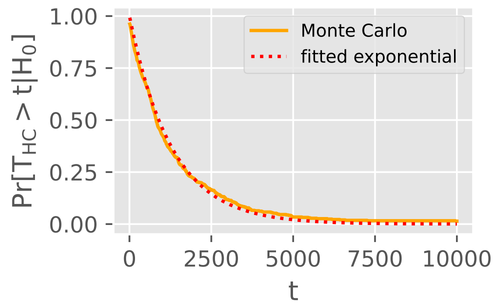


Figure 4.3: Empirical and fitted survival functions of T_{HC} under the null, illustrated for the threshold $b_t = 5$. Data is generated under the null hypothesis with $N = 20,000$ across 500 repetitions.

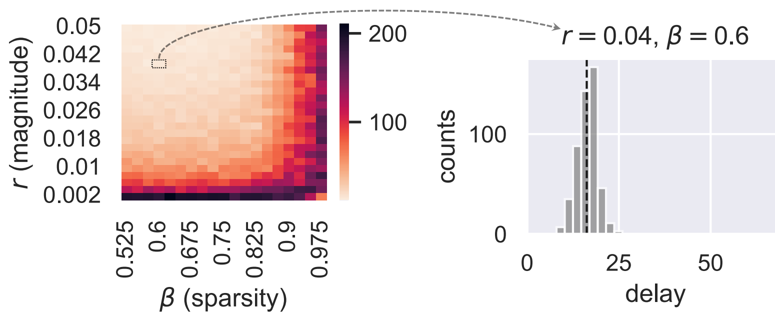
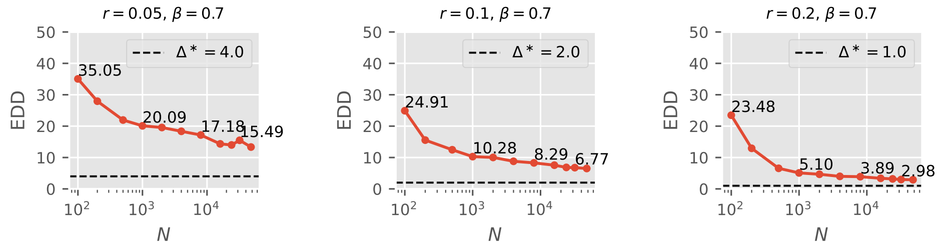


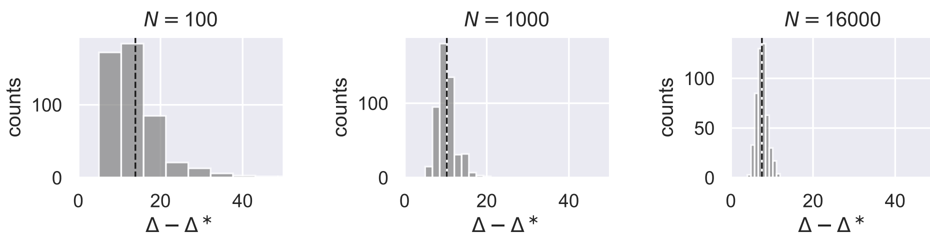
Figure 4.4: Detection delays in sparse normal mean-shift using the HC procedure. Approximately $N^{1-\beta}$ streams undergo a change from $\mathcal{N}(0,1)$ to $\mathcal{N}(\sqrt{2r\log(N)},1)$, with $N = 5,000$ and a grid of (r,β) for change magnitude and sparsity. The target ARL is set to be $5,000$. P-values for each stream are computed using the CUSUM statistic. Left: Average detection delay across 500 Monte Carlo trials for each (r,β) configuration. Right: Histogram of detection delay for a single (r,β) configuration, with the dashed vertical line indicating the corresponding EDD.

the threshold and its expected value to estimate ARL as $\mathbb{E}\left[T_{\mathrm{HC}} \mid H_0\right] \approx 1/\hat{\lambda}$.

To validate the exponential distribution approximation for the stopping time, we fit the exponential function using least squares; the coefficient of determination exceeds 0.99. Figure 4.3 shows that the empirical tail aligns closely with the fitted exponential function. We observed similar fits in similar experiments over a range of values besides $b = 5$.



(a) (b) (c)



(d) (e) (f)

Figure 4.5: Convergence of detection delay $\Delta = T_{\text{HC}} - \tau$ to its theoretical asymptotic value as the number of streams N increases. Data is generated according to (4.6a), with P-values for each stream computed using the CUSUM statistic. The thresholds are tuned to meet a target ARL 5000, and results are based on 500 Monte Carlo repetitions. (a)–(c): Expected detection delay $\mathbb{E}\left[\Delta \mid \Delta \geq T_{\text{HC}} \geq \tau\right]$ versus N for $\beta = 0.7$ and $r \in \{0.05, 0.1, 0.2\}$. Dashed horizontal lines indicate the theoretical asymptotic delay $\Delta^*(r, \beta, \sigma)$. (d)–(f): Histograms of $\Delta - \Delta^*(r, \beta, 1)$ for fixed $(r, \beta) = (0.1, 0.7)$ and $N \in \{100, 1000, 16000\}$; dashed vertical lines represent EDD.

4.4.3 Validation of theoretical results

EDD dependence on sparsity and change magnitude

We begin by illustrating the performance of the HC procedure as the sparsity and magnitude of the change vary. Figure 4.4 shows EDD over a grid of sparsity levels and change magnitudes, with ARL calibrated to be 5000, along with the histogram of detection delays for a single configuration. The EDD and ARL are estimated using 500 Monte Carlo repetitions. The results confirm that the detection delay increases as the change's magnitude becomes weaker and sparser.

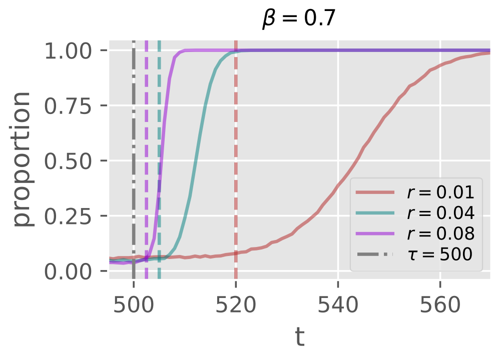


Figure 4.6: Rolling empirical probability of detection for HC-procedure over time t, computed over 4000 Monte Carlo trials, for various change magnitudes controlled by r via (4.11b) and $N = 20,000$. The P-value is calculated using the CUSUM statistic at each stream. At each time t, the percentage of trajectories with detection statistics exceeding the 95 percentile of HC_t^* under the null hypothesis is shown. Dashed lines indicate $\tau + \Delta^*(r, \beta, 1)$, corresponding to the theoretical asymptotic time, exceeding which detection is possible.

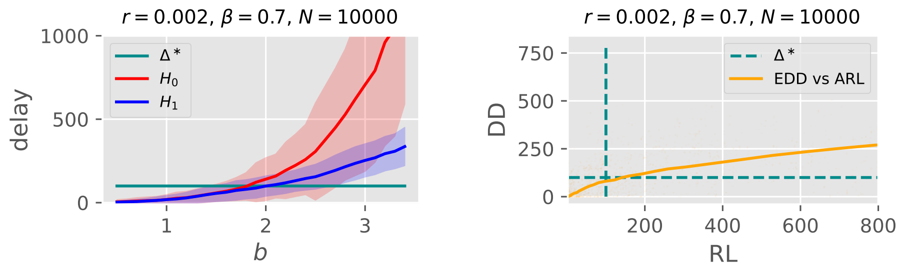
Convergence of detection delay

For a finite N, there is a gap between the empirical detection delay and the asymptotic detection delay $\Delta^*(r, \beta, \sigma)$. As predicted by our asymptotic theory, this gap may shrink with larger N and proper calibration of the threshold b_t. The example in this section verified this numerically.

The first row of Figure 4.5 illustrates the convergence of the EDD to $\Delta^*(r, \beta, \sigma)$: the gap decreases with N for various change magnitude r. The second row of Figure 4.5 shows that the distribution of the detection delay concentrates around $\Delta^*(r, \beta, \sigma)$ as N grows. Figure 4.6 illustrates the probability of detection, over time, subject to an ARL control. This probability increases to one quickly when t exceeds the asymptotic theoretical delay $\Delta^*(r, \beta, \sigma)$.

Detection delay and run length tradeoff

This example illustrates the phase-transition phenomenon discussed in Section 4.3.4: an "undetectable" regime for $t < \tau + \Delta^*(r, \beta, \sigma)$, where the expected detection delay (EDD)



(a) (b)

Figure 4.7: Detection delay (DD) and average no-false-alarm run length (RL) of sparse change point detection using the HC procedure. (a) Empirical expected DD (i.e., EDD) and RL (i.e., ARL) versus the HC threshold b. Colored regions represent standard errors. (b) The curve $(\mathrm{ARL}(b), \mathrm{EDD}(b))$ is based on empirical measurements of many HC threshold b values. As $N \to \infty$, Theorems 4.3.1 and 4.3.2 predicts that the point of significant distinction between the curves in (a) converges to $\Delta^*(r, \beta, \sigma)$, and the bending point in (b) at which the relation between EDD and ARL is no longer linear is at $\Delta^*(r, \beta, 1), \Delta^*(r, \beta, 1)$. The data involves 100 independent experiments with a change at time $t = 0$. Each experiment uses normal data following (4.6a) with $N = 10,000$ streams, roughly $N^{1-\beta} \approx 16$ affected streams $(\beta = 0.7)$ by a mean shift of $\sqrt{2r\log(N)} \approx 0.2$ $(r = 0.002)$, and the CUSUM statistic (4.7) in each stream.

increases linearly with the ARL, and a "detectable" regime for $t \geq \tau + \Delta^*(r, \beta, \sigma)$, where the EDD increases only moderately despite a significant increase in ARL. Figure 4.7 illustrates the mean and standard deviation of detection delay (DD) and the RL across multiple trials for varying HC detection thresholds. As predicted in Theorem 4.3.1, DD and RL are indistinguishable for delays smaller than $\Delta^*(r, \beta, \sigma)$. As predicted in Theorem 4.3.2, for larger delays, it is feasible to calibrate a procedure based on HC that yields a significantly larger RL relative to DD. Namely, a meaningful detection is possible.

4.4.4 Comparisons with other detection procedures

We compare the HC procedure with various existing change point detection procedures, some of which are designed for detecting sparse changes. We examine the procedures' EDD while calibrating each procedure's threshold to reach a target ARL of 5000. For simulating ARL, we set $\mathcal{T} = 20,000$, and the corresponding thresholds are listed in Table 4.1.

The ARLs for each procedure were estimated via Monte Carlo simulations based on 500 trials.

(i) Xie and Siegmund's (XS) [71]:

$$T_{\mathrm{XS}} = \inf\left\{ t : \max_{t-w\le k<t} \sum_{n=1}^{N} \log(1 - p_0 + p_0 e^{(W_{t,k,n}^{+})^2/2}) > b \right\},$$

where $W_{t,k,n}$ is computed for each stream n via (4.9), and $z^+ = \max\{z, 0\}$; here $p_0 = 1/\sqrt{N}$, and $w = 200$.

(ii) Chan's procedure [78]:

$$T_{\mathrm{Chan}} = \inf\left\{ t : \max_{t-w\le k<t} \sum_{n=1}^{N} g\left(W_{t,k,n}^{+}\right) \ge b \right\},$$

where $g(z) = \log\left[1 + p_0\left(Ce^{z^2/4} - 1\right)\right]$ and $C = 2(\sqrt{2} - 1)$, and we use $p_0 = 1/\sqrt{N}$, $w = 200$.

(iii) Chen, Wang, and Samworth (CWS) [110]: The work proposes a likelihood ratio-based detection procedure for Gaussian mean-shift detection by testing against simple alternatives of varying signal magnitude scales in each coordinate and aggregates the test statistics across scales and coordinates. Here, for comparison, we employ the algorithm version from [110, Sec. 3.3], which adapts to both dense and sparse changes. Following the same method as in [110, Sec. 4.1], we tune the three detection thresholds to meet the ARL constraint. The key hyperparameter in this procedure is the lower bound of the ℓ_2 norm of the vector mean change across all streams; we set this to be 1, matching the lower bound of norms for all possible changes considered in our experiments.

(iv) Chen and Chan's (Chen+Chan) [80]: The work proposes a detection statistic for offline change-point detection, based on a score statistic of P-value distribution departure from the uniform distribution. Here, we modify the statistic for online change-point detection, as follows

$$T_{\text{CC}}\left(\lambda_1, \lambda_2\right) = \inf\left\{t : \sum_{n=1}^{N} \log\left(1 + \frac{\lambda_1 \log N}{N} g_1(\pi_{n,t}) + \frac{\lambda_2}{\sqrt{N \log N}} g_2(\pi_{n,t})\right) \geq b\right\},$$

where for $\lambda_1 \geq 0$, $\lambda_2 > 0$, $g_1(z) = \frac{1}{z(2-\log z)^2} - \frac{1}{2}$, and $g_2(z) = \frac{1}{\sqrt{z}} - 2$; in the experiment we choose $\lambda_1 = 1$ and $\lambda_2 = \sqrt{\log \mathcal{T} / \log\log \mathcal{T}}$, following the same choices as in [80].

(v) logp sum: the procedure combines P-values according to Fisher's method as

$$T_{\text{sum}} = \inf\left\{t : -\sum_{n=1}^{N} \log(\pi_{n,t}) \geq b\right\}.$$

The rationale is to aggregate the P-values across all data streams uniformly; this can work well when all streams are affected by the change, but can be less efficient when the change only affects a few coordinates, as seen in our experiments.

(vi) logp min: the procedure uses the smallest P-value as in Bonferroni-type analysis:

$$T_{\min} = \inf\left\{t : -\min_{n=1,\ldots,N} \log(\pi_{n,t}) \geq b\right\}.$$

The procedure works well when only one data stream is affected by the change; however, it may lack combining statistical power when more than one stream is affected by the change.

(vii) orderp SSBH: A test inspired by a statistic associated with the Seeger-Simes family-wise error controlling and the Benjamini-Hochberg false discovery rate controlling, [111], which combines the P-values with weights:

$$T_{\text{SSBH}} = \inf\left\{t : -\min_{n=1,\ldots,N} \frac{\pi_{(n),t}}{n/N} \geq b\right\}.$$

To ensure a fair comparison, we tune the thresholds b for each procedure to achieve the same target ARL of $5,000$. The thresholds, determined through simulation over 500 repetitions, are provided in Table 4.1. We evaluate the methods under the following settings for moderate and high dimensions: (i) $N = 100$, $r = 1$, with varying $|I|$; (ii) $N = 100$, $|I| = 5$, with varying r; (iii) $N = 10^4$, $r = 1$, with varying $|I|$; and (iv) $N = 10^4$, $|I| = 10$, with varying r.

For $N = 100$, the results are shown in Tables 4.2 (for varying $|I|$ with fixed r) and 4.3 (for varying r with fixed $|I|$). In setting (i), the HC-based procedure achieves the smallest EDDs for relatively sparse shifts ($|I| = 1, 3, 5$), while the XS and CWS procedures perform better in denser cases. In setting (ii), where the sparsity is fixed at $|I| = 5$ and the shift magnitude r varies, the HC-based procedure consistently outperforms other methods across different values of r for sparse shifts.

For larger-scale experiments with $N = 10^4$, the results are presented in Tables 4.4 (for varying $|I|$ with fixed r) and 4.5 (for varying r with fixed $|I|$). The HC-based procedure achieves the smallest EDDs for relatively sparse shifts ($|I| = 1, 5, 10$). In Table 4.5, with fixed sparsity $|I| = 10$ and varying r, the HC-based procedure consistently performs well. As a sanity check, Tables 4.2 and 4.4 also include results for the XS and Chan methods from [78], which align closely with our reproduced results.

Computational complexity with respect to dimension N. Here, the complexity corresponds

Table 4.1: Threshold b for stopping rules corresponding to $\mathrm{ARL} = 5000$.

	$N = 100$		$N = 10^4$	
	b	ARL	b	ARL
XS [71]	19.50	4968	113.59	5000
Chan [78]	4.25	5066	4.52	5005
Chen+Chan [80]	2.89	4992	-0.66	5012
logp sum	119.60	4997	9696.15	4999
logp min	10.39	4968	14.21	5021
orderp SSBH	-3.06×10^{-3}	5042	-6.77×10^{-3}	5017
HC-WLCUSUM	9.93	4966	12.11	4990

Table 4.2: EDD of several detection procedures and values of $|I|$ and $N = 100$; data is normal with $r = 1$. EDDs are calculated from 500 repetitions, and the standard errors are in parentheses. The smallest EDD (up to standard errors) is bolded.

	$\|I\|$						
	1	3	5	10	30	50	100
XS [71]	30.1 (0.55)	13.2 (0.20)	9.3 (0.14)	**5.7** (0.08)	2.5 (0.03)	1.8 (0.02)	**1.0** (0.01)
XS presented in [78]	31.6	14.2	10.4	6.7	3.5	2.8	2.0
Chan [78]	25.7 (0.49)	13.3 (0.2)	9.7 (0.14)	6.3 (0.08)	2.9 (0.04)	2.0 (0.02)	1.1 (0.01)
Chan presented in [78]	26.8	13.4	9.6	6.4	2.8	2.0	1.1
CWS [110]	37.2 (0.42)	15.8 (0.21)	10.6 (0.14)	6.5 (0.09)	**2.2** (0.04)	**1.2** (0.02)	**1.0** (0.00)
Chen+Chan [80]	28.6 (0.50)	15.8 (0.21)	11.5 (0.14)	7.3 (0.08)	3.6 (0.03)	2.5 (0.03)	1.8 (0.02)
logp sum	613.0 (11.19)	33.1 (0.48)	20.0 (0.20)	10.6 (0.10)	4.1 (0.03)	2.7 (0.02)	1.8 (0.02)
logp min	18.2 (0.38)	12.2 (0.22)	9.8 (0.17)	8.0 (0.13)	5.6 (0.08)	4.7 (0.07)	4.0 (0.05)
orderp SSBH	18.2 (0.38)	12.2 (0.22)	9.7 (0.17)	7.9 (0.12)	5.5 (0.08)	4.7 (0.07)	3.9 (0.05)
HC-WLCUSUM	**16.3** (0.35)	**10.3** (0.19)	**8.2** (0.14)	6.4 (0.10)	4.0 (0.05)	3.2 (0.04)	2.3 (0.02)

Table 4.3: EDD of several detection procedures and values of r, and $N = 100$; the data is normal as in (4.6a) with a fixed $|I| = 5$. EDDs are calculated from 500 repetitions and the standard errors are in parentheses. The smallest EDD (up to standard errors) is bolded.

	r							
	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8
XS [71]	50.2 (0.81)	23.7 (0.33)	13.8 (0.20)	9.3 (0.12)	6.4 (0.09)	5.0 (0.07)	4.0 (0.05)	3.3 (0.04)
Chan [78]	53.5 (0.86)	25.0 (0.36)	14.4 (0.21)	9.9 (0.13)	6.8 (0.09)	5.3 (0.07)	4.2 (0.06)	3.5 (0.05)
CWS [110]	62.9 (0.91)	28.9 (0.40)	16.4 (0.22)	10.7 (0.15)	7.1 (0.10)	5.4 (0.07)	4.1 (0.06)	3.1 (0.05)
Chen+Chan [80]	63.5 (0.85)	29.5 (0.37)	17.2 (0.23)	11.7 (0.13)	8.1 (0.10)	6.2 (0.07)	5.0 (0.06)	4.2 (0.05)
logp sum	105.3 (1.23)	49.5 (0.56)	29.3 (0.34)	20.1 (0.20)	14.4 (0.14)	11.1 (0.1)	8.8 (0.08)	7.4 (0.07)
logp min	56.4 (1.08)	25.5 (0.44)	15.2 (0.26)	10.2 (0.17)	7.1 (0.12)	5.5 (0.09)	4.4 (0.07)	3.6 (0.05)
orderp SSBH	56.2 (1.07)	25.3 (0.44)	15.1 (0.26)	10.1 (0.16)	7.0 (0.12)	5.5 (0.08)	4.4 (0.07)	3.6 (0.05)
HC-WLCUSUM	**44.9** (0.89)	**20.9** (0.37)	**12.6** (0.22)	**8.3** (0.13)	**5.8** (0.10)	**4.6** (0.07)	**3.7** (0.06)	**3.0** (0.04)

to the computation of the detection statistics at each time t. Due to the requirement to sort the P-values, the HC procedure and procedures (vi) and (vii) have a time complexity of $O(N \log N)$. The CWS procedure has a complexity of $O(N^2 \log N)$ because it aggregates statistics across every pair of streams and considers up to $\log(N)$ different change magnitudes. All other baseline methods have a complexity of $O(N)$.

4.5 Proofs

We now present the proof of the main result. In Section 4.5.1, we provide the necessary background on offline hypothesis testing for rare and weak moderately departure models

Table 4.4: Large N comparsion: EDD for varying $|I|$, while fixing $r = 1$ and $N = 10^4$. EDDs are calculated from 500 repetitions, and the standard errors are in parentheses. In each setting, the smallest EDD (up to standard errors) is bolded. "–" denotes "failure to detect": the procedure did not raise an alarm by the end of the detection time horizon 1000 for all sequences. "N/A" means that a specific result is not available.

	$\|I\|$					
	1	5	10	10^2	10^3	10^4
XS [71]	64.5 (0.9)	20.8 (0.22)	13.9 (0.12)	**4.0** (0.03)	**1.0** (0.0)	**1.0** (0.0)
Chan [78]	37.3 (0.54)	18.0 (0.2)	13.3 (0.13)	4.5 (0.04)	**1.0** (0.0)	**1.0** (0.0)
Chan presented in [78]	37.7	N/A	13.3	4.5	**1.0**	**1.0**
CWS [110]	64.4 (0.61)	20.0 (0.20)	14.4 (0.13)	5.9 (0.05)	**1.0** (0.0)	**1.0** (0.0)
Chen+Chan [80]	870.9 (5.67)	35.2 (0.31)	23.5 (0.16)	9.6 (0.05)	3.7 (0.02)	**1.0** (0.0)
logp sum	–	999.9 (0.07)	998.1 (0.59)	263.4 (0.79)	7.2 (0.02)	**1.0** (0.0)
logp min	**25.8** (0.47)	**15.5** (0.24)	12.7 (0.18)	7.2 (0.09)	4.2 (0.05)	2.8 (0.03)
orderp SSBH	**25.8** (0.47)	**15.5** (0.24)	12.6 (0.18)	7.1 (0.09)	4.2 (0.05)	2.7 (0.03)
HC-WLCUSUM	**25.8** (0.47)	**15.3** (0.23)	**12.4** (0.17)	6.7 (0.08)	3.5 (0.03)	**1.0** (0.0)

Table 4.5: Large N comparison: EDD for varying r, while fixing $|I| = 10$, and $N = 10^4$. EDDs are calculated from 500 repetitions, and the standard errors are in parentheses. The smallest EDD (up to standard errors) is bolded in each setting.

	r							
	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8
XS [71]	79.7 (0.88)	36.7 (0.4)	21.2 (0.21)	13.8 (0.13)	9.9 (0.09)	7.4 (0.07)	5.8 (0.05)	4.7 (0.04)
Chan [78]	76.9 (0.92)	35.3 (0.41)	20.5 (0.21)	13.4 (0.13)	9.4 (0.09)	7.1 (0.07)	5.6 (0.05)	4.6 (0.04)
CWS [110]	87.2 (0.84)	39.5 (0.35)	22.5 (0.20)	14.3 (0.14)	9.9 (0.09)	7.3 (0.07)	5.5 (0.05)	**4.3** (0.04)
Chen+Chan [80]	135.2 (1.05)	61.5 (0.49)	36.0 (0.28)	23.5 (0.17)	16.8 (0.11)	12.7 (0.09)	9.9 (0.06)	8.1 (0.06)
logp sum	997.9 (0.54)	997.2 (0.73)	997.4 (0.83)	997.6 (0.66)	997.9 (0.68)	997.4 (0.62)	998.0 (0.58)	998.3 (0.49)
logp min	**70.7** (1.23)	32.4 (0.55)	19.3 (0.29)	12.6 (0.19)	9.0 (0.13)	6.9 (0.09)	5.4 (0.07)	4.4 (0.06)
orderp SSBH	**70.5** (1.22)	**32.3** (0.55)	19.3 (0.29)	12.6 (0.19)	9.0 (0.13)	6.8 (0.09)	5.4 (0.07)	4.4 (0.06)
HC-WLCUSUM	**69.7** (1.19)	**31.8** (0.53)	**18.9** (0.28)	**12.3** (0.18)	**8.8** (0.12)	**6.7** (0.09)	**5.3** (0.07)	**4.3** (0.05)

and asymptotically log-chi-squared P-values. In Section 4.5.2 we apply these previous results to sparse sequential change point detection with asymptotically log-chi-squared P-values. Section 4.5.3 shows that the P-values relevant to this chapter satisfy these properties. Section 4.5.4 proves the main result.

4.5.1 Rare moderately departing P-values

Rare (sparse) and weak models for signal detection involve detection or classification challenges in which most of the features are useless (pure noise), except perhaps very few features. The locations of the useful features, if there are any, are unknown to us in advance [81, 101]. As explained in [106], previously studied rare and weak models in which departures of non-null features are on the moderate deviation scale can be carried out under the following testing problem, involving independent P-values $\{\pi_n\}_{n=1}^N$ from individual features.

$$\begin{aligned} H_0^{(N)} &: \pi_n \sim \mathrm{Uniform}(0,1), \quad \forall \mathrm{n} = 1,\ldots,\mathrm{N}, \\ H_1^{(N)} &: \pi_n \sim (1-p)\mathrm{Uniform}(0,1) + \mathrm{pQ_n^{(N)}}, \quad \forall \mathrm{n} = 1,\ldots,\mathrm{N}, \end{aligned} \tag{4.14}$$

where the sequence of distributions $\{Q_n^{(N)}\}_{n=1}^N$ obey:

$$X_n \overset{\text{iid}}{\sim} Q_n^{(N)} \Leftrightarrow \lim_{N\to\infty} \max_{n=1,\ldots,N} \left| \frac{-\log \Pr\left[-2\log X_n \geq 2q\log(N)\right]}{\log(N)} - \left(\frac{\sqrt{q}-\sqrt{\rho}}{\sigma}\right)^2 \right| = 0, \tag{4.15}$$

for all $q > \rho > 0$. A sufficient condition for the validity of (4.15) is that

$$-2\log Q_n^{(N)} \overset{D}{=} \left(\sqrt{2\rho\log(N)} + \sigma Z\right)^2 (1 + o_p(1)), \quad Z \sim \mathcal{N}(0,1),$$

where $o_p(1)$ indicates a sequence tending to zero in probability as $N \to \infty$ uniformly in n. Indeed, if $q > \rho$, for N large enough

$$\begin{aligned} &\Pr\left[\left(\sqrt{2\rho\log(N)} + \sigma Z\right)^2 (1 + o_p(1)) \geq 2q\log(N)\right] \\ &= 2\Pr\left[\frac{(\sqrt{2q}-\sqrt{2\rho})\sqrt{\log(N)}}{\sigma}(1+o(1)) \leq Z\right] \\ &= N^{-\left(\frac{\sqrt{q}-\sqrt{\rho}}{\sigma}\right)^2 + o(1)}, \end{aligned}$$

the last transition by Mill's ratio [112].

The name "asymptotically log-chisquared" is because (4.15) holds in particular when

$$X_n \overset{D}{=} \exp\left\{-\frac{1}{2}\left(\sqrt{2\rho\log(N)} + \sigma Z\right)^2\right\}.$$

Namely, when the X_n's follow a non-central log-chisquared distribution over one degree of freedom, as explained in [106], models involving moderate departures in individual features falling under the formulation (4.15) share many common properties concerning the asymptotic behavior of testing procedures. Two properties relevant to our situation are the asymptotic powerlessness of any global test based on $\pi_1 \leq \cdots \leq \pi_N$ when $\rho < \rho^*(\beta, \sigma)$ and the asymptotic powerfulness of HC^* of (4.2) for $\rho > \rho^*(\beta, \sigma)$.

Theorem 4.5.1. *[106, Thms. 1 & 2] Fix $\rho > 0$, $\sigma > 0$, and $\beta > 0$. Assume that $p = N^{-\beta}$ and that $\{\pi_n\}_{n=1}^N$ follow* (4.14). *If $\rho > \rho^*(\beta, \sigma)$, there exists a sequence $\{b^{(N)},\ N = 1, 2, \ldots\}$ such that*

$$\lim_{N \to \infty} \left(\Pr\left[\mathrm{HC}^* < b^{(N)} \mid H_1^{(N)} \right] + \Pr\left[\mathrm{HC}^* \geq b^{(N)} \mid H_0^{(N)} \right] \right) = 0. \tag{4.16}$$

If $\rho < \rho^*(\beta, \sigma)$, for any sequences of test statistics $U = U(\pi_1, \ldots, \pi_N)$ and thresholds $\{b^{(N)},\ N = 1, 2, \ldots\}$,

$$\liminf_{N \to \infty} \left(\Pr\left[U < b^{(N)} \mid H_1^{(N)} \right] + \Pr\left[U \geq b^{(N)} \mid H_0^{(N)} \right] \right) = 1. \tag{4.17}$$

4.5.2 Multistream change-point detection with rare, moderately departed streams

We now provide two corollaries of Theorem 4.5.1 when applied to detect a global change point sequentially across multiple streams of moderately departed P-values. Later, we consider the case where each stream is obtained by individually testing each data stream for a change. The proofs of both corollaries are based on triangular array arguments and are provided in Appendix C.1.

The setting below supposes a random set I such that $\Pr[n \in I] = p = N^{-\beta}$, $\beta \in (0, 1)$, independently for all $n = 1, \ldots, N$, and a stopping rule based on P-values

$$\pi_{n,t} \sim \begin{cases} \mathrm{Uniform}(0,1), & n \notin I \text{ or } t = 1, \ldots, \tau - 1, \\ Q_{n,t}^{(N)}, & n \in I \text{ and } t = \tau, \ldots, \mathcal{T}. \end{cases} \tag{4.18a}$$

Here the array of distributions $\{Q_{n,t}^{(N)},\ n = 1, \ldots, N,\ t = 1, \ldots \mathcal{T}\}$ has the property that for every fixed t, if $X_{n,t} \sim Q_{n,t}^{(N)}$ independently in $n = 1, \ldots, N$, then for some $\rho(\Delta) \geq 0$ we have

$$\lim_{N \to \infty} \max_{n=1,\ldots,N} \left| \frac{-\log \Pr\left[-2\log X_{n,t} \geq 2q\log(N)\right]}{\log(N)} - \left(\frac{\sqrt{q} - \sqrt{\rho(\Delta)}}{\sigma}\right)^2 \right| = 0, \tag{4.18b}$$

for all $q > \rho(\Delta)$. Later on, we show that P-values of LR and GLR tests in the normal heteroscedastic change-point setting (4.6) with $\mu = \mu_N(r) = \sqrt{2r\log(N)}$ satisfy (4.18b) with $\rho(\Delta) = \max\{r(\Delta - \tau + 1), 0\}$.

The first corollary says that when $\rho(\Delta) < \rho^*(\beta, \sigma)$, it is impossible to provide a non-trivial detection procedure using any statistics based on $\{\pi_{n,t}\}$ of (4.18a) that rejects the global null for large values, i.e., small values of $\{\pi_{n,t}\}$.

Corollary 4.5.2. *Define the set* $\Theta_0 := \{\Delta : \rho(\Delta) \leq \rho^-\}$*, for some* $\rho^- < \rho^*(\beta, \sigma)$*. Namely,* Θ_0 *is the set of "insufficient" time delays. Let* T_U *be a stopping rule based on* $U_t := U(\pi_{1,t}, \ldots, \pi_{N,t})$ *and an array of threshold* $\{b_t^{(N)},\ t = 1, 2, \ldots,\ N = 1, 2, \ldots\}$*. Suppose that* $U(x_1, \ldots, x_N) \in \mathbb{R}$ *is non-increasing in either argument. Then*

$$\lim_{N \to \infty} \max_{\Delta \in \Theta_0} \left| \Pr\left[T_U \leq \Delta \mid H_1^{(N)}\right] - \Pr\left[T_U \leq \Delta \mid H_0^{(N)}\right] \right| = 0.$$

The next corollary says that when $\rho(\Delta) > \rho^*(\beta, \sigma)$, there exists a detection procedure based on higher criticism of P-values obeying (4.18) that asymptotically has full power.

Corollary 4.5.3. *Define the set* $\Theta_1 := \{\Delta : \rho(\Delta) \geq \rho^+\}$ *for some* $\rho^+ > \rho^*(\beta, \sigma)$*. Let*

HC_t^* *be higher criticism of* $\{\pi_{1,t}, \ldots, \pi_{N,t}\}$ *of* (4.18). *Suppose that* $\mathcal{T} = o(\log(N))$. *There exists an array of thresholds* $\{b_t^{(N)},\ t = 1, \ldots, \mathcal{T},\ N = 1, 2 \ldots\}$ *such that the stopping time* $T_{\mathrm{HC}} = \inf\{t \,:\, \mathrm{HC}_t^* > b_t^{(N)}\}$ *satisfies*

$$\max_{\Delta \in \Theta_1} \Pr\left[T_{\mathrm{HC}} > \Delta \mid H_1^{(N)}\right] \to 0,$$

while

$$\Pr\left[T_{\mathrm{HC}} \leq \mathcal{T} \mid H_0^{(N)}\right] \to 0.$$

4.5.3 Log-chisquared properties of change-point detection tests

We provide conditions under which the P-values of the likelihood and generalized likelihood ratios are asymptotically log-chisquared uniformly in t. These conditions coincide with the asymptotic calibration of the intensity of individual change points and their sparsity to N in [78, 80]. One notable difference from [78] is the lower bound on the ARL γ, which we assume to go to infinity but at a rate arbitrarily small. This is a weaker result than the rate of $\log(\gamma)N^{-\zeta} \to 0$ provided in [78].

Recall that for the CUSUM and GLR procedures, monitoring for a change in the data's distribution based on Y_t^{LR} or Y_t^{GLR} involves the stopping times

$$T_x^{\mathsf{LR}} := \inf\{t \,:\, Y_t^{\mathsf{LR}} \geq x\},$$

and

$$T_x^{\mathsf{GLR}} := \inf\{t \,:\, Y_t^{\mathsf{GLR}} \geq x\}.$$

Recall that the P-values $\pi_t^{\mathsf{LR}}(x)$ and $\pi_t^{\mathsf{GLR}}(x)$ the survival functions of the statistics Y_t^{LR} and Y_t^{GLR} under the null. If we plug in the corresponding observed statistics value to replace x, we get P-values under the model (4.14).

The distribution of $Y_t^{\square}$, $\square \in \{\mathsf{LR}, \mathsf{GLR}\}$ is continuous under either H_0 or H_1. Therefore, under H_0, we have

$$\pi_t^{\square}(Y_t^{\square}) \sim \mathrm{Uniform}(0,1), \qquad \mathrm{t} = 1, \ldots, \mathcal{T}. \tag{4.19}$$

Under the alternative, the following theorem states that P-values obtained from individual LR or GLR tests in streams experiencing a change under (4.6) are asymptotically log-chisquared with scaling parameter σ and non-centrality parameter $\rho = \rho(\Delta) = r[t-\tau+1]_+$, where $[x]_+ = \max\{0, x\}$.

Theorem 4.5.4. *Assume that $\mu_N(r) = \sqrt{2r\log(N)}$ for some $r > 0$, and for some $\tau \geq 1$ we have $X_1, \ldots, X_{\tau-1} \overset{\mathsf{iid}}{\sim} \mathcal{N}(0,1)$, $X_\tau, \ldots, X_{\mathcal{T}} \overset{\mathsf{iid}}{\sim} \mathcal{N}(\mu_N(r), \sigma^2)$, and $\mathcal{T} = o(\log(N))$. Set $\rho(\Delta) := r[t-\tau+1]_+$. For $\square \in \{\mathsf{LR}, \mathsf{GLR}\}$ we have*

$$-2\log \pi_t^{\square}(Y_t^{\square}) = \left(\sigma Z + \mu_N(\rho(\Delta))\right)^2 (1 + o_p(1)), \tag{4.20}$$

where $o_p(1)$ represents a random variable that goes to 0 in probability as $N \to \infty$, uniformly in t and τ.

4.5.4 Proof of main results

Now we will put everything together to prove the main result for the detection delay in multi-stream with sparse, moderately large change points.

Recall that we consider the change-point detection problem (4.6) with P-values $\pi_{n,t}^{\square}$ of (4.8) for $\square = \mathsf{LR}$ and (4.10) for $\square = \mathsf{GLR}$. These P-values obey (4.18a) and, by Theorem 4.5.4, every $Q_{n,t}^{(N)}$ satisfies (4.18b) with $\rho(\Delta) = r[\tau - \Delta + 1]_+$, hence Corollaries 4.5.2 and 4.5.3 apply.

Proof of Theorem 4.3.1. Let $\Delta_0 := \lceil \rho^*(\beta,\sigma)/r \rceil - 1 < \rho^*(\beta,\sigma)/r$, hence $r\Delta_0 < \rho^*(\beta,\sigma)$. We use Corollary 4.5.2 with $\rho^- = r[\tau - \Delta_0 + 1]_+$ and $\rho(\Delta) = r[\tau - \Delta + 1]_+$. Thus, any

$\Delta \leq \Delta_0 < \rho^*(\beta,\sigma)/r$ is in the set $\Theta_0 = \{\Delta \,:\, \rho(\Delta) \leq \rho^-\}$. Theorem 4.3.1 follows. $\square$

Proof of Theorem 4.3.2. Let Δ^* be the minimal Δ such that $\rho(\Delta) = r[\Delta - \tau + 1]$ (strictly) exceeds $\rho^*(\beta,\sigma)$. Because $\rho^*(\beta,\sigma)/r$ is not an integer, $\Delta^* - \tau + 1 = \lceil \rho^*(\beta,\sigma)/r \rceil$, and thus $\rho(\Delta) \geq \rho(\Delta^*) > \rho^*(\beta,\sigma)/r$ for all $t \geq t^*$. Use Corollary 4.5.3 with $\rho^+ = r\Delta^*$, thus $\Theta_1 = \{\Delta \geq \Delta^*\}$. It follows that there exists an array of thresholds such that

$$\Pr\left[T_{\mathrm{HC}} \leq \Delta^* \mid H_1^{(N)}\right] \to 1, \tag{4.21}$$

while

$$\Pr\left[T_{\mathrm{HC}} \leq \mathcal{T} \mid H_0^{(N)}\right] \to 0.$$

On the other hand, because HC_t^* is non-decreasing as any of $\pi_{1,t},\ldots,\pi_{N,t}$ is decreasing, when $\Delta^* - 1 < \rho^*(\beta,\sigma)/r$, Corollary 4.5.2 implies that

$$\Pr\left[T_{\mathrm{HC}} \leq \Delta^* - 1 \mid H_1^{(N)}\right] \to 0. \tag{4.22}$$

Equations (4.21) and (4.22) implies

$$\Pr\left[T_{\mathrm{HC}} = \Delta^* \mid H_1^{(N)}\right] \to 1.$$

Theorem 4.3.2 follows. $\square$

4.6 Conclusions and open challenges

This chapter introduced a change-point detection procedure that combines information from individual streams using higher criticism (HC). The method is advantageous for practitioners because higher criticism is a well-understood and popular tool for sparse signal detection, having, among other advantages, a built-in selection mechanism to identify a useful set of suspected departing streams [83]. Extensive numerical evaluations show that the HC-based procedure typically outperforms other methods in simulation studies. Finally, the detection delay of the HC procedure has interesting optimality properties: in a sparse heteroscedastic normal change-point setting with CUSUM or general likelihood ratio tests, this delay converges to the theoretical minimal delay, allowing useful change-point monitoring.

Below, we discuss several limitations of our work that invite further exploration.

Finer asymptotic delay characterization. Our main results show that there exists a detection procedure based on HC whose asymptotic detection delay converges in distribution to $\Delta^*(r, \beta, \sigma)$ as the number of streams N goes to infinity. However, Figure 4.5 suggests that this convergence is relatively slow, hence the limiting detection delay may not provide enough information on the performance for finite N. Such information can be provided by characterizing the limiting distribution of the delay, or at least the behavior of higher-order asymptotic terms of its mean.

Better ARL guarantee. Our theoretical analysis allows the ARL to diverge with N but limits it to $o(\log(N))$. Therefore, it provides a lower ARL guarantee than studies such as [78] and [80] which consider ARL scaling as $\sim e^{N^\zeta}$, $\zeta \in (0, 1)$. Despite this limitation of our theoretical analysis, numerical experiments show that detection based on HC performs well compared to other methods under arbitrarily large ARL requirements. Therefore, extending our results to ARL scaling better than $o(\log(N))$ appears to be possible and, thus, an interesting avenue for future work.

Information theoretic delay. We did not fully characterize the information-theoretic detection delay, as alternative tests applied to individual streams might perform better. For instance, the likelihood ratio (LR) test assumes a known post-change mean but not the post-change variance.

Other sparse models. Another direction for future work is the consideration of other sparse data models. For example, the work of [106] suggests that the most interesting tradeoff between change intensity and sparsity occurs when changes follow a moderate deviation scale in N. This insight suggests that sparse sequential change-point detection based on HC has optimality properties under various data models, with affected streams experiencing a moderate change in the distribution. It is also interesting to explore sparse sequential change-point detection under non-moderate distribution changes in analogy to the offline change detection considered in [113, 101, 114, 115, 85].

CHAPTER 5

POINT PROCESS WITH EVENT TIME UNCERTAINTY

5.1 Overview

Point processes, particularly self-exciting point processes (widely known as Hawkes processes; see, e.g., [116, 117]), are extensively used for modeling sequential discrete event data across various fields, including seismology [118, 119, 120], social network analysis [121], high-frequency finance [122], and genomics [123]. Most existing models assume exact knowledge of event times. However, in practice, event times are often uncertain. For example, in medical data, such as COVID-19 cases or ICU monitoring for sepsis detection, the exact onset of a condition may not be observed immediately. Symptoms may appear, but a confirmed diagnosis is only possible after lab results. Similarly, in crime data, the precise timing of an incident, such as a burglary, may be unknown, as the event is often discovered after the fact when the suspect has already left the scene. This uncertainty in event timing presents a significant modeling challenge (illustrated in Figure 5.1).

Addressing timing uncertainty in point processes is a critical research question, however, it has not been extended studied in the existing literature. Early attempts, such as Ogata's Bayesian approach [124], model event timing uncertainty by assuming a prior distribution (e.g., uniform) over a predefined time window. However, selecting an appropriate prior can be subjective, and these early works did not fully address the computational challenges involved. Recent studies have explored a related problem: Hawkes process estimation with time-interval-censored observations, where events are recorded as counts within non-overlapping time intervals. For instance, Schneider et al. [125] developed an expectation-maximization (EM) algorithm for parameter estimation in time-censored Hawkes processes. Rizoiu et al. [126] proposed an approach that approximates the Hawkes process using a Mean Behavior Poisson Process (MBPP), which assumes independent increments to simplify model fitting. However, this approximation deviates from the exact dynamics of the original Hawkes process.

In this chapter, we propose a new approach to address timing uncertainty in discrete event data without relying on prior distributions for event times, thereby adopting a non-Bayesian framework. Our method offers a principled formulation that directly models the underlying continuous-time Hawkes process under time uncertainty, avoiding approximations. We employ maximum likelihood estimation and develop computational algorithms for parameter inference, resulting in a convex optimization problem.

More precisely, our formulation starts from deriving the likelihood model of a sequence of event-windows, under a continuous-time setting, by defining the filtration to be induced by the history of event-window data having time-uncertainty. Because time uncertainty windows often fall on evenly spaced windows in applications, e.g., the medical measurements are recorded every hour, it is natural to assume that the event windows are also aligned with a time grid. This setting reduces our continuous-time model into a discrete-time one, where the learnable model parameters are entries in an influence kernel matrix that is a quantized version of the continuous-time influence kernel function (and a baseline intensity), where the size of the matrix is determined by the time grid and also the influence history time lag. Interestingly, the discrete model obtained in this way has an equivalent formulation in the form of a Bernoulli process, and is also closely connected to the GLM in predicting discrete-time point process data. Our model is then ready to be inferred using either the likelihood, i.e., by Maximum Likelihood Estimation (MLE) using Gradient Descent (GD), or by GLM approaches, particularly, the monotone Variational Inequality (VI) approach. The numerical optimization can be implemented based on mini-batches, namely using Stochastic Gradient Descent (SGD), which allows scalability to large number of observations.

While we provide the general formulation to handle (with time uncertainty) the non-

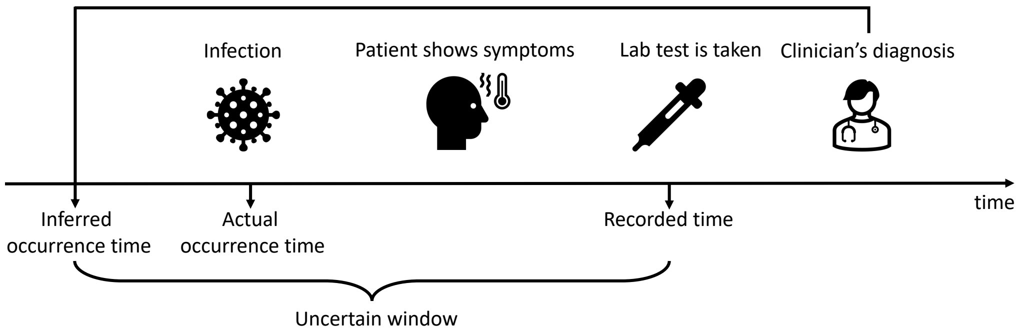


Figure 5.1: Illustration of a real-world medical application scenario leading to discrete event data with time uncertainty.

stationary processes [127], namely the influence kernel is allowed to be time-varying, our model also addresses the case of stationary process as a special case where the influence kernel is time-invariant. In this simplified scenario, instead of having an influence kernel matrix, we will have a finite "influence vector" to infer in the discrete-time model. The proposed model and algorithms all extend to the special case of stationary processes.

Furthermore, we extend our time-uncertainty model to point processes on a network, where the event data have both (uncertain) event time and event location, the latter modeled by indices on a graph. Similarly to the time-only case, we derive the time-uncertainty model firstly in continuous time, and then obtain a discrete-time model after imposing a time grid where the uncertainty event time windows fall onto. Due to the extra spatial dimension, the model parameters include a four-way influence kernel tensor that is jointly indexed by time and graph nodes (and a baseline intensity vector over graph). To infer the model parameters from data, we extend our first-order algorithms by GD and VI using SGD, and we also propose a novel scheme to infer the baseline vector along the SGD iterations.

The effectiveness of our approach are experimentally demonstrated on simulated point process data with time uncertainty, where we show the numerical convergence of the algorithms as well as model recovery accuracy. In addition, under both time-only and on-network settings, our model allows to predict the chance of having an event happening in a future time interval. This is empirically verified by prediction accuracy on simulated data.

Going beyond simulated data, we also apply our model to the prediction of Sepsis using the Sepsis-Associated Derangements (SADs) data [128] from real application, which can be viewed as a point process over a network consisting of medical states. The proposed model shows competitive prediction accuracy and also reveals causal relations among the medical states that are interpretable from a medical viewpoint. On both simulated and real data, the proposed model gives better performance than alternative methods based on GLMs or classical Hawkes process.

In summary, our contributions are:

- We propose a time-uncertainty model for event data, possibly lying on a network, derived rigorously from a series of assumptions motivated by applications. After adopting a time grid, the formulation naturally leads to a discrete-time model that allows inference based on likelihood and prediction. The discrete model takes an equivalent form as a Bernoulli process and is closely connected to the General Linear Model (GLM).

- We develop inference algorithms based on Gradient Descent (GD) and also monotone Variational Inequality (VI), and our algorithm is based on stochastic gradient descent (SGD) using mini-batches.

- Our model covers the non-stationary process where the time-varying influence kernel is reduced to a matrix to infer (and a tensor for the on-network case), and it also includes the stationary process as a special case where the kernel is time-invariant and reduced to a vector to infer.

- The accuracy and efficiency of the proposed approach are demonstrated in experiments on simulated data and also a real dataset of Sepsis-Associated Derangements (SADs). The proposed method outperforms previous GLM and Hawkes process baselines and also reveals interpretable causal relations on the SADs data.

Our data model differs from those considering time-interval-censored observations in [125, 126] in that we assume each bin contains exactly one event, representing a scenario where events are rare, and we can localize them to specific time intervals. This setup contrasts with [125, 126], where multiple events can occur within each non-overlapping bin—a more suitable model for denser observations. While our approach currently focuses on sparse event settings, it can be extended to accommodate multiple events per bin, which we leave for future work.

Another line of work, including Juditsky et al. [129] introduced a discrete-time Bernoulli process to model multivariate event data and the formulation gives rise to a General Linear Model (GLM) which can be efficiently computed by first-order optimization. Later, Wei and Xie [130] implement this model to learn the causal network in time series data. These discrete-time models can be viewed as incorporating the time uncertainty of event data up to the discrete time index adopted, however, the exact modeling of time uncertainty is not explicit. In comparison, our model of event time uncertainty is derived from the probabilistic principle of point processes and is applicable to more general point process data. We will see that our model also gives to a GLM and we followed the optimization techniques in [129].

Notations. The notations used in this work are standard. For a vector $x \in \mathbb{R}^n$, we use both bracket $x(u)$ or subscript x_u to denote the u-th entry of the vector. When to emphasize that x is a vector, we use boldface symbol $\boldsymbol{x}$. $\otimes$ means vector outer-product, i.e., for $u, v \in \mathbb{R}^n$, $u \otimes n$ is an n-by-n matrix. $a \vee b := \max\{a, b\}$, and $a \wedge b := \min\{a, b\}$. $\mathbb{R}_+ = (0, \infty)$ denotes the set of strictly positive real numbers.

5.1.1 Related works

Hawkes process. The Hawkes process, originally studied in [116], is a self-exciting continuous-time point process model, which can capture the excitation effects among events. The Hawkes process [131, 116, 132] with parametric kernel function gives rise to one thread of literature to model the complex dynamics of the heterogeneous time event data, for example, the earthquake occurrences [118, 133]. Due to its good interpretability and well-studied estimations, many applied works are keen on using parametric kernels, including the exponential kernels [134, 121, 135, 136] or the power-law kernels [137, 138]. However, the expressiveness of the Hawkes process seems insufficient when the data is temporally non-stationary in its distribution or not self-exciting. This is due to not only the parametric structure of the kernel in the Hawkes process, but also the limited number of trainable parameters. Later, non-parametric Hawkes models [139], including neural Hawkes models [140], are constructed to accommodate the increasing complexity of real-world applications.

Point process beyond Hawkes. For the events recorded at certain times, more expressive models are developed in two directions. One is to model the intensity function, namely the rate of the arrivals of the events. Hansen et al. [139] aim to find the best linear approximation of the intensity function given a fixed dictionary. Mei and Eisner [141] model the intensity functions through a continuous-time Long Short-Term Memory neural network. As indicated by [127], the modeling of intensity functions has the potential concern of overfitting when the data dynamics are indeed simple. To circumvent this issue, one choice could be the modeling of the kernel function.

For the learning of the kernel function, Zhuang et al. [142] learn a space-time branching process model (the Epidemic Type Aftershock-Sequences model) with a parametric kernel relying on the ancestors' magnitudes, the time lags from offspring events to the current and the location of events. Lewis and Mohler [143] propose a non-parametric EM algorithm to learn the coefficients in the kernel density estimation (KDE) of the triggering kernel function. Zhou et al. [144] improve the work in [143] by expressing the triggering kernel as a linear combination of a series of base kernels instead of a single kernel in KDE. Bacry et al. [145] estimate the shape of the triggering kernel through the empirical auto-covariance of the counting process. Among them, few works consider the non-stationarity of the data in the kernel design.

Uncertainty quantification in point process. Rich literature exists for the uncertainty quantification on the model selection or the model parameter estimation. Wang et al. [146] quantify the uncertainty of the MLE of the multivariate Hawkes process. Dubey et al. [147] use Bayesian neural networks to model the events uncertainty by the posterior distribution of the model parameters. Osama et al. [148] infer the confidence intervals for the intensity functions of the spatial point process. Yang et al. [149] assess the model selection uncertainty using the spatial point process with an application to wildfire occurrences. Theses works study the model parameter uncertainty instead of the event time uncertainty, which is the focus of our work.

Uncertainty of observations in time series. The time series literature also study the uncertainty of the response values, either by probability distributions [150], or by interval-valued observations [151]. Another related branch is fuzzy time series, firstly introduced in [152] and applied to forecast university enrollments in [153, 154]. In fuzzy time series, uncertain observations are modeled as fuzzy sets. Concretely, the observations are recorded by linguistic variables instead of numbers. More recently, Song [155] applies auto-correlation function to measure the dependency between the fuzzy data and Cai et al. [156] perform Ant Colony Optimization (ACO) algorithm to search for the best partition of the universe of discourse. For more details on uncertain time series literature, one can see this survey paper [157]. These models are discrete in time and consider the uncertainty of the observations, while our focus is on the event time uncertainty, and our discrete-time model is obtained from the time uncertainty inherited in the event data.

5.2 Background

In this section, we review some preliminaries of the classical point process (without time uncertainty) to facilitate the comparison with the time-uncertainty model to be introduced and make this work self-contained; more details of the statistical model can be found in [158, 117].

5.2.1 Continuous-time Hawkes process

Given a sequence of n event times $\{t_1, t_2, \ldots, t_n\}$ on $(0, T)$,

$$0 < t_1 < t_2 < \cdots < t_n < T,$$

the (conditional) *intensity function* $\lambda(t)$ is defined as the probability of having the next event in $[t, t + dt)$ given the history, where we use dt to denote the infinitesimal time. Formally,

$$\lambda(t) := \lim_{\Delta t \to 0} \frac{\mathbb{E}[\mathbb{N}([t, t + \Delta t)) \mid \mathcal{H}_t]}{\Delta t}, \tag{5.1}$$

where $\mathcal{H}_t$ is the history of all events up to t, and $\mathbb{N}$ is the counting measure of the events. Define the filtration $\mathcal{F}_k$ to be the σ-algebra generated by observations up to the k-th event, i.e.,

$$\mathcal{F}_k := \sigma\{t_1, \cdots, t_k\}. \tag{5.2}$$

The *survival function* $S_k(t)$ is defined as

$$S_k(t) := \Pr[t_k > t | \mathcal{F}_{k-1}], \quad t \geq t_{k-1}, \tag{5.3}$$

and by definition $S_k(t_{k-1}) = 1$ and $S_k(t) > 0$.

We define the *hazard function*, $h_k(t)$ for the k-th event, in the standard fashion

$$h_k(t) := -\frac{S_k'(t)}{S_k(t)}, \quad t > t_{k-1}, \tag{5.4}$$

and equivalently,

$$S_k(t) = \exp\left\{-\int_{t_{k-1}}^{t} h_k(s) ds\right\}, \quad t \geq t_{k-1}.$$

Thus, by definition, this gives that for each k,

$$\Pr[t_k \in [t, t + dt)|\mathcal{F}_{k-1}] = -dS_k(t) = h_k(t) \exp\left\{-\int_{t_{k-1}}^{t} h_k(s)ds\right\} dt. \qquad (5.5)$$

From the above derivation, it can be shown that the intensity function $\lambda(t)$ defined in (5.1) actually equals the hazard function $h_k(t)$ on the interval $(t_{k-1}, t_k]$. Evaluating (5.5) at $t = t_k$ and applying the argument consecutively for $k = 1, \ldots, n$, we have the expression of the log-likelihood as

$$\ell := \log \Pr[t_1, \cdots, t_n, t_{n+1} > T] = \sum_{k=1}^{n} \log \lambda(t_k) - \int_0^T \lambda(s)ds.$$

The self-exciting mechanism is captured by the parametrization of the intensity $\lambda(t)$ through the *influence kernel* function $k(t', t)$ in the form as

$$\lambda[k](t) = \mu + \sum_{i,\, t_i < t} k(t_i, t), \qquad (5.6)$$

where we have assumed that $\mu > 0$ is the constant base intensity of events throughout time. It is possible to make μ time-dependent in general cases.

In the classical Hawkes process [131, 132], $k(t', t)$ is set to be a time-invariant, exponential function, namely $k(t', t) = \alpha\beta e^{-\beta(t-t')}$ with scalars $\alpha > 0$, $\beta > 0$. Time-invariant kernel also leads to a stationary process whose distribution does not change under time shift. Here, in our framework, we can extend and consider general influence kernel functions, such as time-varying (i.e., not time-invariant), and possibly taking negative values. The general influence kernel can capture how events happening in the past at time t' will influence events happening in the future at time $t > t'$. When the influence is time-invariant, we can write $k(t', t) = k(t - t')$, i.e., it only depends on the time difference. Otherwise, it means that events happening at different past times t' will have a time-varying influence on the future. Because of *temporal causality*, i.e., only the past can influence the future and not the other way around, the influence kernel is asymmetrical.

5.2.2 Point process on networks

Hawkes already discussed the multivariate case in [116]. In a point process on a network $\mathcal{V} = \{1, \ldots, V\}$ having V nodes, we observe the event data

$$(t_1, u_1), (t_2, u_2), \ldots, (t_n, u_n), \tag{5.7}$$

over the time horizon $[0, T]$, $0 < t_1 < \cdots < t_n < T$; t_i is the occurrence time of i-th event, and $u_i \in \mathcal{V}$ denotes the location of the event on the network. We can similarly define the filtration $\mathcal{F}_k$ of precise timing and node information of events as

$$\mathcal{F}_k := \sigma\{(t_1, u_1), \cdots, (t_k, u_k)\}. \tag{5.8}$$

The event history $\mathcal{H}_t$ and the counting measure $\mathbb{N}$ contain both event time and location. The conditional intensity function at time t and location u is defined as

$$\lambda(t, u) := \lim_{\Delta t \to 0} \frac{\mathbb{E}[\mathbb{N}([t, t + \Delta t), u) \mid \mathcal{H}_t]}{\Delta t}, \quad t > 0,\, u \in \mathcal{V},$$

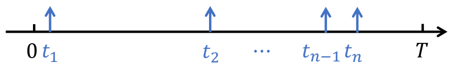
where $\mathcal{H}_t$ is the history of all events up to t. The log-likelihood of n events has the expression as [117]

$$\ell = \sum_{k=1}^{n} \log \lambda(t_k, u_k) - \int_0^T \sum_{v \in \mathcal{V}} \lambda(s, v) ds.$$

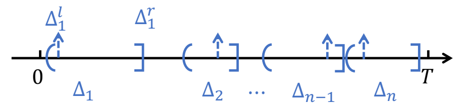
On a network, the influence kernel function $k(t', t, u', u)$ depends on both time and location. The conditional intensity function $\lambda(t, u)$ is parameterized as

$$\lambda[k](t, u) = \mu(u) + \sum_{i,\, t_i < t} k(t_i, t, u_i, u), \tag{5.9}$$

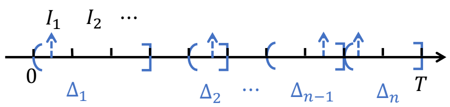
where $\mu(u) > 0$ is the base intensity at location u. The kernel function $k(t', t, u', u)$, $t > t'$,



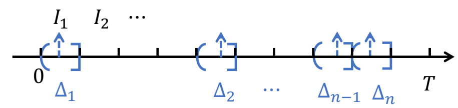
(a) Continuous-time: exact event times



(b) Continuous-time: arbitrary event uncertainty



(c) Discrete-time: arbitrary event uncertainty



(d) Discrete-time: unit-time event uncertainty

Figure 5.2: Illustration of model: (a) Event times without uncertainty in the standard point process model. (b) Continuous-time model with arbitrary-length event window, see Section 5.3.1. (c) Discrete-time model with arbitrary-sized event uncertainty, see Section 5.3.2. (d) Discrete-time model with unit-sized event uncertainty, see Section 5.3.3.

$u, u' \in \mathcal{V}$ represents the influence, i.e., triggering (or inhibiting) effect from the past event at time and location (t', u') on the probability of a future event at (t, u).

Our proposed model to be introduced in Section 5.5 is more expressive than those previously considered in the literature (see, e.g., [117]), including the spatial-temporal factorized model $k(t', t, u', u) = k_1(t', t)k_2(u', u)$. As a result, our model can represent more complex spatial-temporal influence patterns of point processes on graphs.

5.3 Events with time uncertainty

In this section, we establish the time uncertainty model using only event time data, effectively treating it as a one-dimensional point process since there is no location involved. All the proofs are given in Appendix D.1.

We first detail how to express the likelihood of a sequence of events subject to time uncertainty under a continuous-time point process model governed by a general influence kernel function. We then explore a specific scenario where the uncertainty windows align with a uniform grid on $[0, T]$. Furthermore, it is highlighted that if the uncertainty window spans just a single unit of time, it can be equated to a Bernoulli process governed by a specific probability function. This model will be extended to encompass events on a network, considering both time and location, in Section 5.5.

5.3.1 Continuous-time model with arbitrary-length event window

Uncertainty window. The classical Hawkes model assumes that we observe exactly when the events happen; however, in practice, oftentime we do not know the exact time of the events. Instead, we may know when the i-th event happens up to certain *uncertainty window* Δ_i, namely,

$$t_i \in \Delta_i := (\Delta_i^l, \Delta_i^r], \quad i = 1, 2, \ldots, \tag{5.10}$$

where $\Delta_i^l < \Delta_i^r$ are the end-points of the interval for each event. In other words, we do not know exactly when the event happens but only know that it happens within the interval.

We also make the following assumption that each interval only contains one event and the intervals do not overlap. In other words, the observation has enough time resolution to identify an individual event up to its time uncertainty.

Assumption 5.1 (Non-overlapping windows)**.** *(A1) The consecutive intervals Δ_i do not overlap, i.e., $\Delta_{i-1}^r \leq \Delta_i^l$ for all i.*

We would like to remark that the extension to allow multiple events in one uncertainty window may lead to a Poisson process which can be handled by other mathematical tools and is not the focus of this work. The time uncertainty window is illustrated in Figure 5.2(b).

Likelihood model. We are ready to derive the likelihood of the observation

$$\mathcal{L} := \Pr[t_1 \in \Delta_1, \cdots, t_n \in \Delta_n, t_{n+1} > T], \tag{5.11}$$

which will enable model estimation. Our derivation will be based on the first principle and defining the survival and hazard functions under the new setting that the history contains time uncertainty. Specifically, instead of using the filtration $\mathcal{F}_k$ as in (5.2) by precise timing information, we define the filtration to be generated by historical events with time uncertainty, i.e.,

$$\tilde{\mathcal{F}}_k := \sigma\{t_i \in \Delta_i,\, i = 1, \cdots, k\}. \tag{5.12}$$

For notation brevity, below we use Δ_i to stand for the probability event of $t_i \in \Delta_i$ when there is no confusion. The survival function $S_k(t)$ is defined similarly as in the classical case in Section 5.2.1, c.f. (5.3), but conditioning on the uncertain history, that is

$$S_k(t) := \Pr[t_k > t | \tilde{\mathcal{F}}_{k-1}], \quad t \geq \Delta_{k-1}^r. \tag{5.13}$$

Because t_k must happen after Δ_{k-1}^r by Assumption (A1), we have $S_k(\Delta_{k-1}^r) = 1$. The hazard function h_k is defined from S_k same as before, c.f. (5.4),

$$h_k(t) := -\frac{S_k'(t)}{S_k(t)}, \quad t > \Delta_{k-1}^r, \tag{5.14}$$

and note that the time is from $t > \Delta_{k-1}^r$. As a result, $S_k(t)$ can be represented by integrating h_k as

$$S_k(t) = \exp\left\{ -\int_{\Delta_{k-1}^r}^{t} h_k(s) ds \right\}, \quad t \geq \Delta_{k-1}^r. \tag{5.15}$$

We further define the intensity function $\lambda(t)$ by *piecing together* the hazard function h_k from each interval as

$$\lambda(t) := h_k(t), \quad t \in (\Delta_{k-1}^r, \Delta_k^r], \quad k = 1, 2, \cdots, \tag{5.16}$$

and for $k = n + 1$ it will from the interval $(\Delta_n^r, T]$.

Lemma 5.3.1. *The log-likelihood of $\mathcal{L}$ as in* (5.11) *has the expression*

$$\ell := \log \mathcal{L} = \sum_{k=1}^{n} \log\left(\exp\left\{ \int_{\Delta_k^l}^{\Delta_k^r} \lambda(s) ds \right\} - 1 \right) - \int_0^T \lambda(s) ds. \tag{5.17}$$

Influence kernel. In the classical Hawkes model, the intensity function $\lambda(t)$ is further parametrized by the influence kernel function $k(t', t)$, see (5.6). Here, we also parametrize the intensity function (5.16) by a linear superposition of $k(t', t)$, and we adopt the following expression to incorporate time uncertainty

$$\lambda[k](t) = \mu + \sum_{i, \Delta_i^r < t} \frac{1}{|\Delta_i|} \int_{\Delta_i^l}^{\Delta_i^r} k(t', t) dt'. \tag{5.18}$$

The notation has emphasized the dependence of $\lambda(t)$ on the continuous time kernel function k.

Remark 5.1 (Consistency)**.** *We choose the form of conditional intensity function in* (5.18) *because it is consistent with the earlier Hawkes conditional intensity model specified in* (5.6)*. Specifically, if the uncertainty window Δ_i shrinks to a single point t_k, i.e., $|\Delta_i| \to 0$ for all i, then, by that $t_i \in (\Delta_i^l, \Delta_i^r]$ according to* (5.10) *(and assuming the continuity of the function k), the expression* (5.18) *is reduced to* (5.6)*. In this limit, the time-uncertain model for the conditional intensity function $\lambda(t)$ recovers the classic (time-certain) Hawkes model.*

Remark 5.2 (Time-invariant kernel)**.** *In* (5.18)*, we have assumed the most general form of kernel $k(t', t)$, which is allowed to be non-stationary. The time-invariant kernel is a special case when $k(t', t) = \psi(t - t')$ for some function ψ. We will detail the inference methods to learn kernel from data in Section 5.4, under the framework of the discrete-time model to be introduced below. Later in Section 5.4, we will first present the general case of a time-varying kernel (corresponding to the recovery of a matrix), and then discuss the special case of the time-invariant kernel (corresponding to the recovery of a vector) in Section 5.4.4.*

5.3.2 Discrete time model: Arbitrary event uncertainty

In many applications, the observed uncertainty windows are some regular time intervals, e.g., a window having a length of one or more hours that starts at integer hours. For example, in the physionet challenge data [128], the event (patient test results) record are kept in an hourly manner. We show that under this setting, the continuous-time time uncertainty model above can be simplified into a special case of a discrete-time model, where the uncertainty are up to a certain time unit.

Time grid. We evenly divide the time horizon $(0, T)$ into N intervals, each having length h,

$$h := T/N, \quad I_j := ((j-1)h, jh], \quad j \in [N] := \{1, \cdots, N\}. \tag{5.19}$$

In the discrete-time models, we use h to stand for the size of the time grid, not to confuse the notation with the hazard function h_k, which was denoted by $\lambda(t)$ from (5.16) on.

To proceed, we assume that the event time uncertainty intervals Δ_i are "aligned" with the discrete-time grids, as illustrated in Figure 5.2(c). We index the N intervals by the set $[N] := \{1, \cdots, N\}$.

Assumption 5.2 (Alignment to time grid)**.** *(A2) The uncertain event time window Δ_i can only have the end points Δ_i^l and Δ_i^r taking value on the evenly spaced discrete time grid points $\{jh : j \in [N]\}$.*

Under (A2), the event-time window Δ_i for each event i occupies some consecutive intervals of I_j. When Δ_k occupies from the l-th interval till the r-th interval, we denote the index l as t_k^l, the index r as t_k^r, that is,

$$\Delta_k = \cup_{t_k^l \le j \le t_k^r} I_j, \quad t_k^l, t_k^r \in [N], \quad t_k^l \le t_k^r, \quad t_k^r < t_{k+1}^l. \tag{5.20}$$

The last inequality $t_k^r < t_{k+1}^l$ is by that the uncertainty windows Δ_k do not overlap as assumed in (A1). The assumption (A2) provides a discrete-time formulation that will allow us to simplify the model (5.17) and (5.18) by writing everything in discrete time indexed by $j \in [N]$.

Likelihood and influence model. Under (A2), the log-likelihood (5.17) only depends on the N quantities which are integral of $\lambda(t)$ on the subintervals I_j. To be specific, we define

$$\Lambda_j := \frac{1}{h}\int_{I_j} \lambda(s)ds, \quad j \in [N], \tag{5.21}$$

which can be interpreted as the "average intensity" over a discrete time interval I_j. With the definition of Λ_j, (5.17) is reduced to

$$\ell = \sum_{k=1}^{n} \log\left(e^{h\sum_{j'=t_k^l}^{t_k^r} \Lambda_{j'}} - 1\right) - h\sum_{j=1}^{N} \Lambda_j. \tag{5.22}$$

Next, we derive the representation of intensity Λ_j by the influence kernel. Define the matrix

$$K_{i,j} := \frac{1}{h^2}\int_{I_i}\int_{I_j} k(t', t)dt'dt, \quad i < j, \quad i, j \in [N], \tag{5.23}$$

which can be interpreted as the "average influence" casted on the interval I_j from an earlier interval I_i. We will show that only the $i < j$ entry of $K_{i,j}$ is used. We call K the *influence kernel matrix*, which can be viewed as a quantization of the kernel function $k(t', t)$.

The following lemma provides the representation of Λ_j by the kernel matrix $K_{i,j}$.

Lemma 5.3.2. *Under Assumption (A2), for $j \in [N]$,*

$$\Lambda_j[K] = \mu + \sum_{\{k,\, t_k^r < j\}} \frac{1}{t_k^r - t_k^l + 1} \sum_{i=t_k^l}^{t_k^r} K_{i,j}. \tag{5.24}$$

In the notation we have emphasized the dependence of Λ_j on the kernel matrix K. The lemma shows that the dependence of the discrete-time intensity function Λ_j from the influence kernel function $k(t', t)$ is all encoded into the matrix $K_{i,j}$.

The equations (5.22) and (5.24) provide a discrete time model which only involves discrete-time objects Λ_j and $K_{i,j}$. This will facilitate computation, as we can thus reduce the functional optimization problem with respect to $k(t', t)$ to a matrix-valued optimization problem with respect to $K_{i,j}$.

5.3.3 Discrete time model: Unit-time event uncertainty

The discrete-time formulation in Section 5.3.2 has a simple and important special case, which we call "unit uncertainty,"as follows.

Assumption 5.3 (Unit-time uncertainty)**.** *(A3) The uncertainty window Δ_i falls on exact one of the N intervals, i.e., $\Delta_i = I_{j(i)}$ for some $j(i) \in [N]$. In other words, $t_i^l = t_i^r = j(i)$.*

Likelihood and influence model. Under Assumption (A3), the discrete-time model (5.22) and (5.24) is further simplified into

$$\ell = \sum_{k=1}^{n} \log \left(e^{h\Lambda_{j(k)}} - 1 \right) - h \sum_{j=1}^{N} \Lambda_j, \quad \Lambda_j = \mu + \sum_{\{k:\, j(k)<j\}} K_{j(k),j}. \tag{5.25}$$

Like before, $\Lambda_j = \Lambda_j[K]$ which depends on the kernel matrix K.

Meanwhile, when the baseline intensity μ is unknown, in principle it can also be inferred from data. Thus one can treat the scalar μ as a learnable parameter. We write the learnable parameters $\{\mu, K\}$ together as θ, and write Λ_j as $\Lambda_j(\theta)$ to emphasize the dependence on the parametrization. In this work, our inference algorithms and experiments will focus on the unit-time uncertainty case under (A3).

In the rest of this subsection, we first show that the model (5.25) has an equivalent formulation in the form of a Bernoulli process, which will give a close connection to the general linear model for discrete-time point process [129] and also facilitate notations in future sections. In addition, we show that, once the parameters are learned, our discrete-time model allows to predict event chance on a future interval.

Bernoulli process formulation. Under the unit uncertainty setting (A3), our discrete-time model then has an equivalent form as a Bernoulli process. Below, we will use i, j and also $t \in [N]$ to denote the time-grid index, not to confuse with continuous time t.

Specifically, each trajectory of the event time uncertainty window data can be written as a binary sequence $y = (y_1, \cdots, y_N)$, where $y_t = 1$ if $I_t = I_{j(k)}$ for some k-th event (i.e., there is an event happening inside I_t), and $y_t = 0$ otherwise. Using the notation of y_j, (5.25) can be equivalently written as

$$\ell = \sum_{j=1}^{N} \left(-(1-y_j)h\Lambda_j + y_j \log(1-e^{-h\Lambda_j})\right), \quad \Lambda_j(\theta) = \mu + \sum_{i<j} y_i K_{i,j}. \tag{5.26}$$

Furthermore, the following lemma characterizes the conditional expectation of y_t given the time-uncertain history. In this lemma, Λ_t refers to the true intensity Λ_t^*, namely induced by the true parameter under which the sequence $\{y_t\}_t$ is generated. The lemma provides a general linear model to predict y_t using Λ_t. We note that the link function ϕ here takes a particular form under the framework of [129], and it is not a logistic function.

Lemma 5.3.3. *Under the unit-uncertainty setting Assumption 5.3(A3) and using the formulation of the Bernoulli process y_t, for any $t \in [N]$, $\Lambda_t \in \sigma\{y_i, i \leq t-1\}$ and*

$$\mathbb{E}[y_t | y_i, i \leq t-1] = \phi(h\Lambda_t), \quad \phi(x) := 1 - e^{-x}. \tag{5.27}$$

The function ϕ is monotonically increasing on $\mathbb{R}$. The relationship (5.27) will be used in deriving the estimation procedure.

Prediction on future interval. Once the parameters θ are learned from data, we can compute Λ_j based on (5.25) and equivalently (5.26). This allows to predict the chance of a event happening during a future interval. To derive the expression, we take a step back to recall the notations of the continuous-time setting in Section 5.3.1. Given n historical events with uncertainty, by the definition of S_k in (5.13), the chance of having the next event on a given interval $(l, r]$, $r > l \geq \Delta_n^r$, can be written as

$$\Pr[t_{n+1} \in (l, r] | \tilde{\mathcal{F}}_n] = S_{n+1}(l) - S_{n+1}(r). \tag{5.28}$$

Under the discrete-time model and using the notation in Section 5.3.2, suppose l and r are endpoints of the time grid that is,

$$l = j_l h, \quad r = j_r h, \quad j_r > j_l \geq t_n^r.$$

By (5.28), (5.15) and the definition of Λ_j in (5.21), we have

$$\Pr[t_{n+1} \in (j_l h, j_r h] | \tilde{\mathcal{F}}_n] = e^{-h \sum_{j'=t_n^r+1}^{j_l} \Lambda_{j'}} \left(1 - e^{-h \sum_{j'=j_l+1}^{j_r} \Lambda_{j'}} \right). \tag{5.29}$$

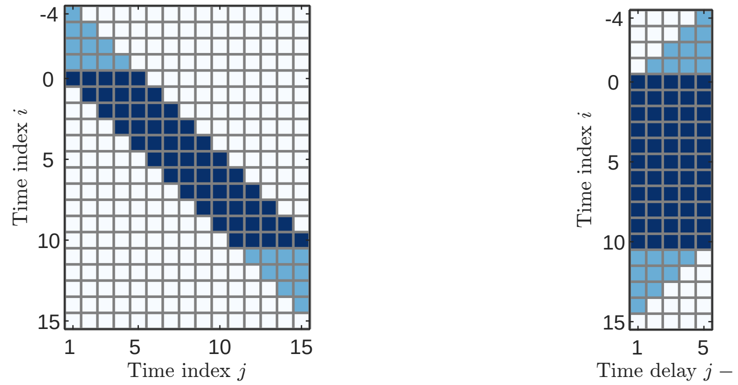
For the probability of having no event happening till time $l = j_l h$, we have

$$\Pr[t_{n+1} > j_l h | \tilde{\mathcal{F}}_n] = e^{-h \sum_{j'=t_n^r+1}^{j_l} \Lambda_{j'}}. \tag{5.30}$$

In particular, under unit-time uncertainty (A3), $\Delta_n = I_{j(n)}$ and $t_n^r = j(n)$. One can compute (5.29) and (5.30) with $t_n^r = j(n)$ in the expressions.

5.4 Estimation of influence kernel

Given an observed sequence of event data with time uncertainty, our goal is to recover the influence kernel up to the time uncertainty. The derivations in Section 5.3 provide a discrete-time model, where the learnable parameters are the influence kernel matrix $K_{i,j}$ (and the scalar base intensity μ). In this section, we develop estimation approaches to recover the kernel matrix from data, focusing on unit uncertainty setting under Assumption



(a) Kernel K (b) Kernel Ψ

Figure 5.3: Kernel matrix K in (5.23) and the reparameterized form Ψ in (5.32) . We set $N' = 5$ and $N = 15$. The blue (including light and dark blue) elements represent the trainable parameters θ_K, see (5.31); The white elements are not involved in the computation of the likelihood and thus excluded from θ_K. We apply the low-rank constraint to the submatrix formed by the dark blue entries in Ψ.

5.3(A3), see Section 5.3.3.

5.4.1 Kernel matrix and log-likelihood

Recall that θ denotes the trainable parameters in the model, which consists of entries in the influence kernel matrix K and μ. In the model (5.25) and equivalently (5.26), only $K_{i,j}$ for $j > i$ are contributing to the likelihood. Thus, only $K_{i,j}$ for $j > i$ are included into θ. We further assume that the influence has a finite memory length, that is,

$$k(t', t) = 0, \quad \text{if } t - t' > \tau_{\max}.$$

Without loss of generality, assume that $\tau_{\max} = N'h$ for some integer N'. By definition (5.23), we have

$$K_{i,j} = 0, \quad \text{if } j - i > N'.$$

Thus, we only include $K_{i,j}$ for $i < j \le i + N'$ into θ, as illustrated in Figure 5.3(a).

The trainable parameters are then

$$\theta = \{\mu\} \cup \{K_{i,j},\, 0 < j - i \le N',\, -N' < i \le N,\, 0 < j \le N\} =: \{\mu\} \cup \theta_K, \quad (5.31)$$

where θ_K has NN' trainable entries and thus is in the vector space of $\mathbb{R}^{NN'}$. Here, for convenience, we have assumed negative time grids indexed from $-N'+1$ to 0 with possible observed event data (that is, the Bernoulli process model in Section 5.3.3 starts at $-N'+1$ instead of 1). These historical data are used to compute Λ_j for $j > 0$, while the likelihood is computed on $1 \le j \le N$. The vector θ_K can be re-arranged into an array indexed by $-N' < i \le N$ and $0 < j - i \le N'$, which we call the kernel matrix Ψ,

$$\Psi_{i,l} = K_{i,i+l}, \quad l = 1, \cdots, N', \quad (5.32)$$

and Ψ is of size $(N+N') \times N'$, see Figure 5.3(b). In many cases, it is reasonable to assume that the re-arranged kernel matrix Ψ is low-rank in a certain way, see more in Section 5.4.4.

Using the notations above, our likelihood model (5.26) can be written as

$$\ell(\theta) = \sum_{t=1}^{N} \left(-(1-y_t)h\Lambda_t(\theta) + y_t \log(1 - e^{-h\Lambda_t(\theta)})\right), \quad \Lambda_t(\theta) = \mu + \langle \eta_t, \theta_K \rangle,$$

and $\eta_t \in \mathbb{R}^{NN'}$ is a vector consisting of binary valued entries determined by the process y_t such that

$$\langle \eta_t, \theta_K \rangle = \sum_{i=t-N'}^{t-1} y_i K_{i,t}. \quad (5.33)$$

In practice, we are given M many independent (training) trajectories $\{y_t^{(m)}\}_{m=1}^{M}$, and thus the log-likelihood is summed over the M trajectories. The trajectories may potentially differ in length, and the likelihood model is still well-defined. For simplicity, we assume the M trajectories are of the same length N, and the trajectories $y^{(m)}$ are i.i.d. across m. We introduce the superscript $^{(m)}$ to stand for the m-th trajectory and apply to the notations

y_t, η_t, Λ_t and log-likelihood ℓ. We denote the (averaged) summed log-likelihood as $L(\theta)$, and the likelihood model is summarized as

$$L(\theta) = \frac{1}{M} \sum_{m=1}^{M} \ell^{(m)}(\theta), \tag{5.34}$$

$$\ell^{(m)}(\theta) = \sum_{t=1}^{N} \left(-(1 - y_t^{(m)}) h \Lambda_t^{(m)}(\theta) + y_t^{(m)} \log(1 - e^{-h\Lambda_t^{(m)}(\theta)}) \right), \tag{5.35}$$

$$\Lambda_t^{(m)}(\theta) = \mu + \langle \eta_t^{(m)}, \theta_K \rangle. \tag{5.36}$$

We propose first-order approaches to estimate the model parameter from data. Given the likelihood model, it is natural to estimate the parameters using MLE by e.g. Gradient Descent (GD). Before going into more detail about GD, we introduce another approach to estimate the model parameters using Variational Inequality (VI) optimization [159].

5.4.2 Monotone Variational Inequality (VI) for parameter recovery

The VI approach leverages the GLM model in (5.27) with link function $\phi : \mathbb{R} \to \mathbb{R}$, and formally minimizes the L^2 loss $L_{\mathrm{VI}}(\theta) = \frac{1}{M} \sum_{m=1}^{M} \ell_{\mathrm{VI}}^{(m)}(\theta)$, where

$$\ell_{\mathrm{VI}}^{(m)}(\theta) = \frac{1}{2h} \sum_{t=1}^{N} (\phi(h\Lambda_t^{(m)}(\theta)) - y_t^{(m)})^2.$$

Recall that $\theta = \{\mu, \theta_K\}$. For simplicity and without loss of generality, in the rest of this section we always assume that μ is a known constant, and focus on estimating the influence kernel $\theta_K \in \mathbb{R}^{NN'}$. We will explain algorithmic detail to recover μ from data in Section 5.6.

VI stochastic scheme. For notation convenience, we denote θ_K as $z \in \mathbb{R}^{NN'}$, and also write $\Lambda_t(z) = \Lambda_t([\mu, z])$, i.e., with $\theta = [\mu, z]$. Since $\Lambda_t^{(m)}$ is linear in $z = \theta_K$ and

$\partial_{\theta_K}\Lambda_t^{(m)}(\theta) = \eta_t^{(m)}$ by (5.36), we have the VI vector field defined as

$$\hat{G}(z) = \frac{1}{M}\sum_{m=1}^{M}\hat{G}^{(m)}(z), \quad \hat{G}^{(m)}(z) = \sum_{t=1}^{N}\left(\phi(h\Lambda_t^{(m)}(z)) - y_t^{(m)}\right)\eta_t^{(m)}. \tag{5.37}$$

While computing VI using $\hat{G}(z)$ is possible, instead, we will consider a stochastic scheme to compute VI , following the stochastic optimization framework in [159]. Our algorithm in Section 5.6 will adopt stochastic optimization looping over batches of training samples.

We use z_k to stand for the parameter at the k-th iteration, $k = 0, 1, \cdots$. The stochastic updating scheme reads

$$z_k = \mathrm{Proj}_{\Theta_K}[z_{k-1} - \gamma_k \hat{G}^{(k)}(z_{k-1})], \tag{5.38}$$

where $\gamma_k > 0$ is the step size, Θ_K is a convex compact domain in $\mathbb{R}^{NN'}$ satisfying certain technical assumptions to be specified below, and Proj_Ω is the metric projection operator onto convex set Ω defined as

$$\mathrm{Proj}_\Omega(z) := \arg\min_{u\in\Omega} \|u - z\|_2^2. \tag{5.39}$$

To proceed, we introduce the following assumption on the domain Θ_K of z.

Assumption 5.4 (Physical intensity)**.** *There exist constants $B, b > 0$ s.t.*

(i) The true conditional intensity Λ_t^* satisfies

$$b \le \Lambda_t^* \le B, \quad \forall t = -N' + 1, \cdots, N, \quad \textit{a.s.} \tag{5.40}$$

(ii) There exists a non-empty, compact, and convex set $\Theta_K \subset \mathbb{R}^{NN'}$ such that the true kernel parameter $z^* \in \Theta_K$, and for any $z \in \Theta_K$,

$$b \le \Lambda_t(z) \le B, \quad \forall t = 1, \cdots, N, \quad \textit{a.s.} \tag{5.41}$$

To clarify the relationship of Λ_t^* and z^*, note that z^* is the true kernel matrix restricted to the NN' entries in (5.31). Thus $\Lambda_t^* = \Lambda_t(z^*)$ when $1 \le t \le N$, and Λ_t^* for $-N' < t \le 0$ also involves entries of the true kernel outside the NN' index set.

We say that the intensity $\Lambda_t(z) > 0$ is "physical". Assumption 5.4(ii) imposes that the model intensity $\Lambda_t(z)$ is upper and lower bounded by positive constants. For example, when $z = 0$, $\Lambda_t \equiv \mu$. Then, as long as $[b, B]$ contains μ, we have (5.41) hold at $z = 0$. Thus, one can initialize the iteration (5.38) from $z_0 = 0$ assuming that $0 \in \Theta_K$.

In practice, we compute the VI using batch-based stochastic updates, and the constraint of Λ_t being physical, corresponding to the lower bound in Assumption 5.4(ii), is enforced by a barrier penalty. The details will be introduced in Section 5.6.

5.4.3 Gradient Descent (GD) for MLE

Given the likelihood model (5.34)(5.35)(5.36), the MLE solves for parameter θ by maximizing $L(\theta)$. Again, we assume that μ is known and focus on solving for the kernel matrix parameters $z = \theta_K \in \mathbb{R}^{NN'}$. Define $L(z) := L([\mu, z])$, the GD dynamic follows the gradient field

$$F(z) = -\partial_z L(z) = \frac{1}{M} \sum_{m=1}^{M} \hat{F}^{(m)}(z), \quad \hat{F}^{(m)}(z) = -\partial_z l^{(m)}([\mu, z]),$$

and the GD update is by $z_k = z_{k-1} - \gamma_k F(z_{k-1})$ at step size $\gamma_k > 0$. One can also implement stochastic optimization similar as in (5.38), replacing $\hat{G}^{(k)}$ with $\hat{F}^{(k)}$.

Direct computation gives the expression of $\hat{F}^{(m)}$ as

$$\hat{F}^{(m)}(z) = \sum_{t=1}^{N} \frac{h}{\phi(h\Lambda_t^{(m)}(z))} (\phi(h\Lambda_t^{(m)}(z)) - y_t^{(m)}) \eta_t^{(m)}, \tag{5.42}$$

as well as the Hessian of $L(z)$.

One may use the i.i.d. or possibly relaxed independence assumptions on trajectories $y^{(m)}$ to derive a concentration argument and show that when there are sufficiently many training trajectories, $-L$ is strongly convex and the MLE has a unique solution.

Comparing the GD gradient field (5.42) to the VI gradient field (5.37), we see that $\hat{F}^{(m)}$ has an extra multiplicative factor $h/\phi(h\Lambda_t^{(m)}(z))$ for each time t. While $\phi(h\Lambda_t^{(m)}(z)) \sim h$ under Assumption 5.4, and thus this factor does not change the order of magnitude of the gradient field, the term $\phi(h\Lambda_t^{(m)}(z))$ on the denominator is a random variable and can introduce additional fluctuation especially when the conditional intensity at t is small. This is consistent with our empirical observation that VI dynamics are more stable than GD, see Appendix D.3.3.

5.4.4 Special structures in influence kernel

Time-invariant kernel. When the influence kernel $k(t', t)$ is time-invariant, see Remark 5.2, we know that the kernel matrix K has the pattern that $K_{i,j} = \Psi_{i,j-i} = \psi_{j-i}$, where $\{\psi_l, l = 1, \cdots, N'\}$ is a vector of length N'. Thus the learnable parameters in the kernel matrix are the vector $\psi \in \mathbb{R}^{N'}$, assuming that μ is known and fixed. The likelihood model (5.34)(5.35) still holds, where the kernel parametrization of $\Lambda_t^{(m)}$ is

$$\Lambda_t^{(m)}(\psi) = \mu + \langle \xi_t^{(m)}, \psi \rangle, \quad \text{where } \langle \xi_t, \psi \rangle = \sum_{i=t-N'}^{t-1} y_i \psi_{t-i}. \tag{5.43}$$

The vector $\xi_t \in \mathbb{R}^{N'}$ consists of binary valued entries determined by the history of y_t from $t - N'$ to $t - 1$. The estimation by VI and GD applies here, and the optimization is reduced to solving for the vector ψ of dimension N' instead of the kernel parameter θ_K of dimension NN' as in (5.31). We give more details of VI recovery in below.

Similar to (5.37), the VI vector field is

$$\hat{G}_s(\psi) = \frac{1}{M} \sum_{m=1}^{M} \hat{G}_s^{(m)}(\psi), \quad \hat{G}_s^{(m)}(\psi) = \sum_{t=1}^{N} \left(\phi(h\Lambda_t^{(m)}(\psi)) - y_t^{(m)} \right) \xi_t^{(m)}, \tag{5.44}$$

and we use the subscript $_s$ to stand for "stationary". Define

$$G_s(\psi) := \mathbb{E}\hat{G}_s^{(m)}(\psi) = \mathbb{E}\sum_{t=1}^{N}\left(\phi(h\Lambda_t(\psi)) - y_t\right)\xi_t, \tag{5.45}$$

and again we have $G_s(\psi^*) = 0$ where ψ^* is the true time-invariant kernel. We assume that the true conditional intensity Λ_t^* satisfies Assumption 5.4(i), and similar to (ii) we assume that

(ii') There exists a non-empty, compact and convex set $\Theta_\psi \subset \mathbb{R}^{N'}$ such that the true kernel parameter $\psi^* \in \Theta_\psi$, and for any $\psi \in \Theta_\psi$, $b \leq \Lambda_t(\psi) \leq B$, $\forall t = 1, \cdots, N$, a.s.

Low-rank structure. There are different possible ways to impose low-rank structures on the influence kernel, and in our discrete-time model, naturally on the kernel matrix. Recalling the representation of the kernel matrix as K and Ψ in Section 5.4.1 and Figure 5.3. We think it is natural to impose the low-rank structure on Ψ, as previously adopted in [127] for influence kernel functions. In particular, the time-invariant kernel corresponds to the case where $\Psi = \mathbf{1}_N\psi^T$ (after filling the entries outside the region of NN' entries to be recovered from data), that is, Ψ is a rank-1 matrix.

Theoretically, the kernel matrix is constructed by a grid average of the kernel function, see (5.23). Thus, the low rankness of the kernel matrix can be a result of the discretization of a continuous kernel function $k(t', t)$, assumed to have certain regularity with respect to the varying times t' and t. In Section 5.7, we experimentally study several simulated examples where the kernel matrix is induced from some continuous function k, and numerically the matrix Ψ can be approximated by a low-rank matrix.

We need some additional tricks when applying the low-rank constraint in practice. As has been explained in Section 5.4.1, the NN' learnable parameters in θ_K form a parallelogram-shaped region in the matrix Ψ which is of size $(N + N') \times N'$, see Figure 5.3(b). Thus even the underlying Ψ has low-rankness (after filling the $(N + N') \times N'$ matrix), the inferred

Ψ by optimization has no values (or zero values) outside the parallelogram region, and this interferes with the low-rankness. To overcome this issue, observe that when Ψ is rank-r, then any $m \times N'$ submatrix Ψ' of Ψ (by retrieving rows) is at most rank-r, and the rows of Ψ' are spanned by the r right singular vectors of Ψ. We choose the submatrix Ψ' to be the "middle chunk" of Ψ that has full rows inside the parallelogram region, indicated by dark blue entries in Figure 5.3(b). In practice, we impose low rankness on Ψ' using a truncated SVD, and use the right singular vectors of Ψ' to apply a projection of the inferred full matrix Ψ, see more in Section 5.6.

5.5 Time-uncertainty point process on network

In this section, we extend the time uncertainty model to the network setting, namely when the event data also contains location information u_i for the i-th event and u_i is a node on a graph. All the proofs can be found in Appendix D.2.

5.5.1 Continuous-time formulation

Filtration and conditional intensity. We consider the event data on a network $\mathcal{V} = \{1, \cdots, V\}$ where the event time t_i has uncertainty, and the event location $u_i \in \mathcal{V}$. The classical scenario with exact event time is provided in (5.7), as a comparison. Following the formulation in Section 5.3, we consider $t_i \in \Delta_i$ same as before. Again, let Δ_i also denote the event that $\{t_i \in \Delta_i\}$, the filtration is defined as

$$\tilde{\mathcal{F}}_k := \sigma\{(\Delta_i, u_i),\, i = 1, \cdots, k\}.$$

We define the survival function $S_k(t)$ and hazard function $h_k(t)$ same as in (5.13) and (5.4) respectively, that is, $S_k(t)$ and $h_k(t)$ only addresses the time of the event regardless of the event location. We then have that (5.14)(5.15) also hold.

To be able to incorporate event location information u_i, we define $\tilde{f}_k(u|t)$ by

$$\tilde{f}_k(u|t) := \Pr[u_k = u | t_k \in [t, t+dt),\, \tilde{\mathcal{F}}_{k-1}], \tag{5.46}$$

and then, for $t > \Delta^r_{k-1}$,

$$\begin{aligned} \Pr[t_k \in [t, t+dt), u_k = u | \tilde{\mathcal{F}}_{k-1}] &= \Pr[t_k \in [t, t+dt) | \tilde{\mathcal{F}}_{k-1}] \tilde{f}_k(u|t) \\ &= -S'_k(t) \tilde{f}_k(u|t) dt \\ &= S_k(t) h_k(t) \tilde{f}_k(u|t) dt \quad \text{(by (5.14))} \\ &= S_k(t) \lambda(t, u) dt, \end{aligned} \tag{5.47}$$

where we define

$$\lambda(t, u) := h_k(t) \tilde{f}_k(u|t). \tag{5.48}$$

By that $\tilde{f}_k(u|t)$ is a conditional distribution of $u \in \mathcal{V}$ and thus $\sum_{u \in \mathcal{V}} \tilde{f}_k(u|t) = 1$, we have

$$\sum_{u \in \mathcal{V}} \lambda(t, u) = h_k(t) =: \bar{\lambda}(t), \tag{5.49}$$

where we define $\bar{\lambda}(t)$ to equal $h_k(t)$ piece-wisely on $(\Delta^r_{k-1}, \Delta^r_k]$, similarly as in (5.16). Back to (5.47), by integrating dt over the interval Δ_k on both sides, we have

$$\begin{aligned} \Pr[t_k \in \Delta_k,\, u_k = u | \tilde{\mathcal{F}}_{k-1}] &= \int_{\Delta^l_k}^{\Delta^r_k} S_k(t) \lambda(t, u) dt \\ &= \int_{\Delta^l_k}^{\Delta^r_k} \lambda(t, u) e^{-\int_{\Delta^r_{k-1}}^{t} h_k(s) ds} dt \quad \text{(by (5.15))} \qquad (5.50) \\ &= e^{-\int_{\Delta^r_{k-1}}^{\Delta^r_k} \bar{\lambda}(s) ds} \int_{\Delta^l_k}^{\Delta^r_k} \lambda(t, u) e^{\int_t^{\Delta^r_k} \bar{\lambda}(s) ds} dt. \qquad (5.51) \end{aligned}$$

Likelihood model. With these notions in hand, in the following lemma, we derive the expression of the likelihood

$$\mathcal{L} := \Pr[(\Delta_1, u_1), \cdots, (\Delta_n, u_n),\, t_{n+1} > T]$$

Lemma 5.5.1. *Recall that* $\bar{\lambda}(t) = \sum_{u \in \mathcal{V}} \lambda(t, u)$*,*

$$\log \mathcal{L} = \sum_{k=1}^{n} \log \left(\int_{\Delta_k^l}^{\Delta_k^r} \lambda(t, u_k) e^{\int_t^{\Delta_k^r} \bar{\lambda}(s) ds} dt \right) - \int_0^T \bar{\lambda}(s) ds. \tag{5.52}$$

Remark 5.3 (Reduction to classical scenario without time uncertainty)**.** *We consider the limit where the length of window* Δ_i *shrinks to zero, that is,* $\Delta_i^l,\, \Delta_i^r \to t_i$ *the certain event time for all* i*. We claim that in this case,*

$$\frac{1}{|\Delta_k|} \Pr[t_k \in \Delta_k,\, u_k = u | \tilde{\mathcal{F}}_{k-1}] \to \lambda(t_k, u) e^{-\int_{t_{k-1}}^{t_k} \bar{\lambda}(s) ds}. \tag{5.53}$$

This is can be derived from (5.50)*, which gives that the l.h.s. can be written as*

$$\frac{1}{|\Delta_k|} \int_{\Delta_k^l}^{\Delta_k^r} \lambda(t, u) e^{-\int_{\Delta_{k-1}^r}^{t} \bar{\lambda}(s) ds} dt,$$

and then using the continuity of $\lambda(t, u)$ *(and* $\bar{\lambda}(t)$*) with respect to* t*. The equation* (5.53) *leads to the expression*

$$\Pr[(t_k, u) | \mathcal{F}_{k-1}] = \lambda(t_k, u) e^{-\int_{t_{k-1}}^{t_k} \bar{\lambda}(s) ds},$$

where we recall that in the classical (time-certain) model, $\mathcal{F}_k := \sigma\{(t_i, u_i),\, i = 1, \cdots, k\}$*. Then, the log-likelihood of the* n *events can be shown to be*

$$\log \mathcal{L} = \sum_{k=1}^{n} \log \lambda(t_k, u_k) - \int_0^T \bar{\lambda}(s) ds,$$

which recovers the classical model without time uncertainty, see e.g. Eqn (8) in [117].

Remark 5.4 (Reduction to time-only model)**.** *Consider the special case where the network $\mathcal{V}$ only has one node u. In this case, $\bar{\lambda}(t) = \lambda(t,u)$ and define it to be $\lambda(t)$. Then, one can verify that*

$$\int_{\Delta_k^l}^{\Delta_k^r} \lambda(t) e^{\int_t^{\Delta_k^r} \lambda(s)ds} dt = e^{\int_{\Delta_k^l}^{\Delta_k^r} \lambda(s)ds} - 1.$$

This equality reduces the conditional probability (5.51) *to* (D.1)*, and the log-likelihood* (5.52) *to* (5.17)*. That is, the model is reduced to the time-only case in Section 5.3.1.*

Influence kernel. We introduce the spatial-temporal influence kernel function $k(t', t; u', u)$ to parametrize the intensity function $\lambda(t,u)$. By incorporating the spatial dependence in (5.18), we have

$$\lambda[k](t,u) = \mu(u) + \sum_{i, \Delta_i^r < t} \frac{1}{|\Delta_i|} \int_{\Delta_i^l}^{\Delta_i^r} k(t', t; u_i, u) dt', \tag{5.54}$$

where $\mu(u) > 0$ for $u \in \mathcal{V}$ is the base intensity on the network $\mathcal{V}$. Based on the continuous-time model (5.52) and (5.54), we will derive discrete-time model in the next subsection under additional assumptions.

5.5.2 Discrete-time model and discrete parametrization

We consider the time grid as in Section 5.3.2 which has N intervals I_j, $|I_j| = h = T/N$. For simplicity, we only consider the unit uncertainty setting as in Assumption 5.3(A3).

Time-quantized influence kernel. Suppose $\Delta_k = I_{j(k)}$, the log-likelihood (5.52) can be written as

$$\log \mathcal{L} = \sum_{k=1}^{n} \log \left(\int_{I_{j(k)}^l}^{I_{j(k)}^r} \lambda(t, u_k) e^{\int_t^{I_{j(k)}^r} \bar{\lambda}(s)ds} dt \right) - \int_0^T \bar{\lambda}(s) ds. \tag{5.55}$$

This, however, still cannot be expressed by the average of $\lambda(t,u)$ on $t \in I_j$ only, due to that $\lambda(t,u)e^{\int_t^{I_{j(k)}^r} \bar{\lambda}(s)ds}$ takes different value over location u and $t \in I_{j(k)}$. To be able to reduce to a discrete problem, we introduce the following assumption on the influence kernel.

Assumption 5.5 (Time-quantized influence kernel on network)**.** *For any $u', u \in \mathcal{V}$ and $1 \leq i < j \leq N$, $k(t', t; u', u)$ is constant on $(t', t) \in I_i \times I_j$, and we denote the value as $K_{i,j}(u', u)$.*

A useful consequence of the assumption is that the influence function $\lambda(t,u)$ is also quantized in time, which is proved in the following lemma.

Lemma 5.5.2 (Discrete-time intensity)**.** *Under (A3) and Assumption 5.5, for any $u \in \mathcal{V}$, $\lambda(t,u)$ as defined in* (5.54) *is constant over $t \in I_j$. We denote the value as $\Lambda_j(u)$, and*

$$\Lambda_j(u) = \mu(u) + \sum_{\{k:\, j(k)<j\}} K_{j(k),j}(u_k, u). \tag{5.56}$$

Likelihood model and learnable parameters. According to (5.49), we define

$$\bar{\Lambda}_j = \sum_{u \in \mathcal{V}} \Lambda_j(u). \tag{5.57}$$

The following lemma derives the log-likelihood which only involves $\{\Lambda_j(u),\ j \in [N], u \in \mathcal{V}\}$.

Lemma 5.5.3 (Discrete-time likelihood on network)**.** *Under Assumption 5.3(A3) and Assumption 5.5, the log-likelihood defined in* (5.52) *can be written as*

$$\ell := \log \mathcal{L} = \sum_{k=1}^{n} \log \left((e^{h\bar{\Lambda}_{j(k)}} - 1) \frac{\Lambda_{j(k)}(u_k)}{\bar{\Lambda}_{j(k)}} \right) - h \sum_{j=1}^{N} \bar{\Lambda}_j, \tag{5.58}$$

The learnable parameters in the model are the baseline intensities $\mu(u)$ and the influence kernel $K_{i,j}(u', u)$, both are on the network, which determines the network conditional intensities $\Lambda_j(u)$ as in (5.56). Adopting the same time-grid setup in Section 5.4.1, we consider the extended time grid from $-N'+1$ to N, where the max time lag for influence is $\tau_{\max} = N'h$. Denoting the learnable parameters as θ, we have

$$\theta = \{\mu(u),\, u \in \mathcal{V}\} \cup \{K_{i,j}(u',u),\, u',u \in \mathcal{V},\, 0 < j-i \le N',\, -N' < i \le N,\, 0 < j \le N\}. \tag{5.59}$$

We similarly introduce the re-arranged kernel matrix Ψ where

$$\Psi_{i,l}(u',u) = K_{i,i+l}(u',u), \quad \forall u',u \in \mathcal{V}, \quad l = 1,\cdots,N'. \tag{5.60}$$

We will discuss the estimation of parameters from data in Section 5.5.4.

Prediction on future interval. Extending the argument in Section 5.3.3, we can predict the probability of having the next event happen during a future interval and at a location on $\mathcal{V}$ once the parameter θ is learned from data. Specifically, recall the notations in the continuous-time setting in Section 5.5.1, we are given n historical events and the goal is to predict the probability of the $(n+1)$-th event happening on the interval $(l,r]$, $r > l \ge \Delta_n^r$. Using the discrete-time model under (A3), $\Delta_n = I_{j(n)}$, and again suppose l and r are endpoints on the time grid, we have

$$l = j_l h, \quad r = j_r h, \quad j_r > j_l \ge j(n).$$

We then have

$$\begin{aligned}
\Pr[t_{n+1} \in (l = j_l h, r = j_r h],\, u_{n+1} = u | \tilde{\mathcal{F}}_n] &= \int_l^r \lambda(t,u) e^{-\int_{\Delta_n^r}^t \bar{\lambda}(s)ds} dt \quad \text{(by (5.50))} \\
&= \sum_{j=j_l+1}^{j_r} \frac{\Lambda_j(u)}{\bar{\Lambda}_j}(1 - e^{-h\bar{\Lambda}_j}) e^{-h\sum_{j'=j(n)+1}^{j-1} \bar{\Lambda}_{j'}}, \qquad (5.61)
\end{aligned}$$

where the second equality is derived similar as the proof of Lemma 5.5.3, making use of Lemma 5.5.2 under Assumption 5.5. The probability of having no event till l is

$$\Pr[t_{n+1} > l = j_l h | \tilde{\mathcal{F}}_n] = e^{-h \sum_{j'=j(n)+1}^{j_l} \bar{\Lambda}_{j'}}, \tag{5.62}$$

which is similar to the time-only case where we replace $\Lambda_{j'}$ with $\bar{\Lambda}_{j'}$ in the expression (5.30).

Example: Continuous-time counterfactual treatment effect analysis. We now highlight a key application of the time-uncertain point process on a network model for estimating treatment effects in individual continuous-time counterfactual analysis. This is illustrated through a simplified hypothetical example of blood pressure regulation via treatment. Consider two node network, representing patient's blood pressure level, and treatment. Suppose we have observed a sequence of observations on the network (Δ_i, u_i), $i = 1, 2, \ldots, n$, where $u_i \in \{1, 2\}$ indicating the two types of events: $u_i = 1$ means a "high blood pressure" event when the blood pressure is greater than a threshold, and $u_i = 2$ corresponds to "treatment" event, and Δ_i provides the knowledge of the information when the event took place, i.e., $\{t_i \in \Delta_i\}$ for the ith event. The width of the interval Δ_i indicate our knowledge about each event, and in the extreme case, they can tend to 0, meaning that we have precise knowledge about when the event took place. Now we are interested in evaluating, at the current time t, the effect of having a treatment in the time interval Δ_{n+1} (and this means $u_{n+1} = 2$), on a patient to reducing the risk of having a high-blood pressure effect in a specified future time window $\Delta^\star := [t + a_1, t + a_2]$, where $a_2 > a_1 > 0$ are two user specified constants for future window. The problem of treatment effect estimation can be formulate as follows. The effect of treatment is

$$\Pr[t_{n+2} \in \Delta^\star, u_{n+2} = 1 | \tilde{\mathcal{F}}_n \cup \{\Delta_{n+1}, u_{n+1} = 2\}]$$

and the effect of non treatment is

$$\Pr[t_{n+1} \in \Delta^\star, u_{n+1} = 1 | \tilde{\mathcal{F}}_n],$$

and the treatment effect is the difference of the two, which quantifies the reduction in the risk of having a "high blood pressure event" in the future time window, by having a treatment event in the currently specified time window. These values can be evaluated using the expression (5.61) above, once the model parameters are known. The model parameters can be estimated from the training data following the procedure outlined below.

5.5.3 Spatial-temporal Bernoulli process on network

We have derived the discrete time model (5.58) and (5.56) under (A3) and Assumption 5.5. Here, we introduce an equivalent formulation of the model using a spatial-temporal Bernoulli process on the graph $\mathcal{V}$. Below, we use the boldface notation to emphasize vectors. For example, $\boldsymbol{\mu} = \{\mu(u), u \in \mathcal{V}\} \in \mathbb{R}^V$ and $\boldsymbol{\Lambda}_j = \{\Lambda_j(u), u \in \mathcal{V}\} \in \mathbb{R}^V$ are length-V vectors. We use the boldface $\mathbf{x}$ to denote the vector, and $x(u)$ or x_u stands for the u-th entry. When to emphasize the dependence on parameter θ, we write $\boldsymbol{\Lambda}_j[\theta]$ for the vector and $\Lambda_j[\theta](u)$ for the u-th entry.

Given a trajectory of discrete-time unit-uncertainty event data, define a vector-valued binary sequence $(\mathbf{y}_1, \cdots, \mathbf{y}_N)$, and for $t \in [N]$, $\mathbf{y}_t \in \{0,1\}^V$, where $y_t(u) = 1$ if there exists an event (Δ_i, u_i) s.t. $\Delta_i = I_t$ and $u_i = u$; $y_t(u) = 0$ otherwise. We also define

$$\bar{y}_t := \sum_{u \in \mathcal{V}} y_t(u), \quad t \in [N],$$

and $\bar{y}_t = 1$ indicates that within time interval I_t, there is an event happening somewhere on the network. Using the binary vectors $\mathbf{y}_t$, (5.56) is equivalent to

$$\Lambda_t[\theta](u) = \mu(u) + \sum_{i<t} \sum_{u' \in \mathcal{V}} y_i(u') K_{i,t}(u', u), \tag{5.63}$$

and the log-likelihood (5.58) can be expressed as

$$\ell = \sum_{t=1}^{N} \left\{ \bar{y}_t \log \left((1 - e^{-h\bar{\Lambda}_t}) \sum_{u \in \mathcal{V}} y_t(u) \frac{\Lambda_t(u)}{\bar{\Lambda}_t} \right) - (1 - \bar{y}_t) h \bar{\Lambda}_t \right\}. \tag{5.64}$$

To ensure that the conditional intensity is physical, i.e. $\Lambda_t[\theta](u) > 0$ for any $u \in \mathcal{V}$ and all time t, one can introduce an assumption similar to Assumption 5.4 and require that, for positive constants $b < B$, $\Lambda_t[\theta](u) \in [b, B]$ for all u and t a.s. when model parameter θ lies in certain domain, including the true intensity $\Lambda_t^*(u)$. As a result, $\mathbf{\Lambda}_t^* \in [b, B]^V \subset \mathbb{R}_+^V$ for all t.

Use the vector form of $\mathbf{\Lambda}_t$ and $\mathbf{y}_t$ which are in $\mathbb{R}^V$, we have the following lemma which characterizes the general linear model of predicting y_t given history using $\mathbf{\Lambda}_t$, and this gives a counterpart of Lemma 5.3.3 on network. In this lemma, $\mathbf{\Lambda}_t$ refers to the true intensity $\mathbf{\Lambda}_t^*$ induced by the true model parameters which determine the law of the sequence $\{\mathbf{y}_t\}_t$.

Lemma 5.5.4. *Under Assumption 5.3(A3) and Assumption 5.5, and using the formulation of the Bernoulli process $\mathbf{y}_t$ on $\mathcal{V}$, for any $t \in [N]$, $\mathbf{\Lambda}_t \in \sigma\{\mathbf{y}_i, i \le t-1\}$ and*

$$\mathbb{E}[\mathbf{y}_t | \mathbf{y}_i, i \le t-1] = \left(\phi(h\bar{\Lambda}_t) \frac{\Lambda_t(u)}{\bar{\Lambda}_t} \right)_{u \in \mathcal{V}} = \Phi(h\mathbf{\Lambda}_t), \tag{5.65}$$

where $\Phi : \mathbb{R}_+^V \to \mathbb{R}_+^V$ is defined by

$$\Phi(\boldsymbol{x}) := \frac{1 - \exp\{-\boldsymbol{x}^T \mathbf{1}\}}{\boldsymbol{x}^T \mathbf{1}} \boldsymbol{x}. \tag{5.66}$$

Because $\boldsymbol{x} \in \mathbb{R}_+^V$ satisfies $\boldsymbol{x}^T \mathbf{1} > 0$, (5.66) *is well-defined.*

5.5.4 Influence kernel recovery by VI and GD

Log-likelihood of kernel parameter. Suppose we are given M i.i.d. sequence of observed Bernoulli processes $\{\mathbf{y}_t^{(m)}\}_{m=1}^M$ on the network $\mathcal{V}$, which are of the same length indexed by $-N' < t \le N$. For simplicity, we assume that the baseline intensity vector $\boldsymbol{\mu}$

is known and fixed, and we will explain the learning of $\boldsymbol{\mu}$ from data in Section 5.6.

We denote model parameter θ in (5.59) as $\theta = \{\boldsymbol{\mu}\} \cup \theta_K$, where θ_K consists of the $NN'V^2$ many influence kernel parameters, again denoted by z. We write $\boldsymbol{\Lambda}_t[z] = \boldsymbol{\Lambda}_t[\boldsymbol{\mu}, z]$, i.e., with $\theta = [\boldsymbol{\mu}, z]$, and similarly for other quantities depending on θ. Next, we introduce the data history vector $\eta_{t,u}$, doubly indexed by t and u and in $\mathbb{R}^{NN'V^2}$, s.t.

$$\langle \eta_{t,u}, z \rangle = \sum_{i=t-N'}^{t-1} \sum_{u' \in \mathcal{V}} y_i(u') K_{i,t}(u', u). \tag{5.67}$$

This allows (5.63) to be represented as $\Lambda_t[z](u) = \mu(u) + \langle \eta_{t,u}, z \rangle$, and $\bar{\Lambda}_t[z] = \sum_{u \in \mathcal{V}} \Lambda_t[z](u)$. With this notation, by (5.64), the log-likelihood of parameter θ on the m-the sequence can be written as

$$\ell^{(m)}[z] = \sum_{t=1}^{N} \left\{ \bar{y}_t^{(m)} \log \left((1 - e^{-h\bar{\Lambda}_t^{(m)}[z]}) \sum_{u \in \mathcal{V}} y_t^{(m)}(u) \frac{\Lambda_t^{(m)}[z](u)}{\bar{\Lambda}_t^{(m)}[z]} \right) - (1 - \bar{y}_t^{(m)}) h \bar{\Lambda}_t^{(m)}[z] \right\}, \tag{5.68}$$

$$\Lambda_t^{(m)}[z](u) = \mu(u) + \langle \eta_{t,u}^{(m)}, z \rangle, \quad \bar{\Lambda}_t^{(m)}[z] = \sum_{u \in \mathcal{V}} \Lambda_t^{(m)}[z](u), \tag{5.69}$$

and the total likelihood is averaged over the M trajectories.

VI scheme on network. With the mapping Φ defined as in (5.66), the per-trajectory L^2 loss can be written as

$$\ell_{\mathrm{VI}}^{(m)}(z) = \frac{1}{2h} \sum_{t=1}^{N} \| \Phi(h \boldsymbol{\Lambda}_t^{(m)}[z]) - \mathbf{y}_t^{(m)} \|_2^2.$$

For a vector $y \in \mathbb{R}^V$, we use both bracket $y(u)$ or subscript y_u to denote the u-th entry of the vector. Following the stochastic update scheme in Section 5.4.2, the per-trajectory VI field is

$$\hat{G}^{(m)}[z] = \sum_{t=1}^{N} \sum_{u \in \mathcal{V}} \left(\Phi(h\mathbf{\Lambda}_t^{(m)}[z]) - \mathbf{y}_t^{(m)} \right)_u \eta_{t,u}^{(m)}$$
$$= \sum_{t=1}^{N} \sum_{u \in \mathcal{V}} \left(\phi(h\bar{\Lambda}_t^{(m)}[z]) \frac{\Lambda_t^{(m)}[z](u)}{\bar{\Lambda}_t^{(m)}[z]} - y_t^{(m)}(u) \right) \eta_{t,u}^{(m)}, \tag{5.70}$$

We can update the solution z_k at the k-th iteration by looping over the training trajectories similarly as before.

GD scheme on network. The GD dynamic solves for the MLE based on the likelihood model (5.68)(5.69). Again we focus on the stochastic GD update, and by differentiating (5.68) the per-trajectory gradient field can be computed directly to be

$$\hat{F}^{(m)}(z) = \sum_{t=1}^{N} \sum_{u \in \mathcal{V}} \left(\frac{h(\phi(h\bar{\Lambda}_t^{(m)}[z]) - \bar{y}_t^{(m)})}{\phi(h\bar{\Lambda}_t^{(m)}[z])} + \frac{\bar{y}_t^{(m)}}{\bar{\Lambda}_t^{(m)}[z]} - \frac{y_t^{(m)}(u)}{\Lambda_t^{(m)}[z](u)} \right) \eta_{t,u}^{(m)}, \tag{5.71}$$

where we used that when $y_t(u) = 1$, $\sum_{u' \in \mathcal{V}} y_t(u')\Lambda_t(u') = \Lambda_t(u)$ because when time t has even then only one node can have event on it.

We see that (5.71) contains terms like $y_t(u)/\Lambda_t(u)$ which will be $\Lambda_t(u)^{-1}$ at an event location u, and thus can be large when $\Lambda_t(u)$ is small. In comparison, the VI field in (5.70) does not have such terms (the ratio $\Lambda_t(u)/\bar{\Lambda}_t$ is always between 0 and 1, which will not cause divergent magnitude). This suggests that the VI training dynamic can be more stable than the GD dynamic, echoing the same phenomenon in the time-only case (see comments at the end of Section 5.4.3).

Special structures for spatial-temporal kernels. Similar to Section 5.4.4, one can incorporate special structures of the kernel structure into the inference procedure when such prior knowledge is available. For example, assuming time stationarity, one will have a time-invariant kernel $K_{i,j}(u', u) = \Psi_{i,j-i}(u', u) = \psi_{j-i}(u', u)$, $\forall u, u' \in \mathcal{V}$, and there are $N'V^2$ many kernel parameters to learn from data. The likelihood model (5.68)(5.69) still holds, and the VI and GD optimization methods can also be extended.

In the network case, because the spatial-temporal kernel parameters K or Ψ are four-way tensors, there are different ways to impose low-rank structures on the kernel. Motivated by the underlying temporal continuity of the influence kernel function, as has been explained in Section 5.4.4, one can expect approximate low-rankness of the discrete kernel over time, e.g., the matrix $[\Psi_{i,l}(u', u)]_{i,l}$ for each pair of (u', u). As an additional simplification, we concatenate these matrices over all pairs of (u', u) to form a tall matrix of size $(N - N' + 1)V^2 \times N'$, using the middle-chunk trick in Section 5.4.4, and impose low-rankness on the resulting matrix.

The above two techniques exploit the temporal structures of the kernel, and it is also possible to impose constraints or regularizations based on the spatial information, i.e. graph topology of the network. In the real data application of this work, to discover a network structure using the Sepsis-Associated Derangements (SADs) data (Section 5.7), we do not assume any prior knowledge of the network where graph nodes represent medical status. Instead, we will use the inferred spatial-temporal kernel from event data to reveal influence relations among the nodes. Thus, in this work, we do not explore any spatial structure or regularization of the kernel, which is possible and would certainly be useful in other application scenarios.

5.6 Algorithm

The estimation methods using VI and GD have been introduced for the time-only case in Section 5.4 and extended to the on-network case in Section 5.5.4. We provide additional algorithmic details in this section.

5.6.1 Time-only case

Stochastic optimization by batches. In practice, the batch size in stochastic optimization algorithm balances between efficiency of parameter adaptation vs. random fluctuation of

Algorithm 3: Stochastic batch-based training for time-only case

Input : Training trajectories $\{y_t^{(m)}\}_{m=1}^{M}$, μ_0 as initial value of μ; Parameters: intensity lower bound b, barrier weight δ_b ; batch size M_B, maximum number of epochs $k_{\max}$, learning rate schedule $\{\gamma_k\}_k$; Optional: singular value threshold τ_{SVD}, smoothness weight δ_s

Output: Learned kernel θ_K and baseline μ

Initialize kernel $\theta_K \leftarrow 0$ and baseline $\mu \leftarrow \mu_0$
for $k = 1, \ldots, k_{\max}$ **do**
    while loop over the batches **do**
        $\theta \leftarrow [\mu, \theta_K]$
        Load M_B many training trajectories to form batch $Y_B = \{y^{(m)}\}_m$
        Compute intensities $\Lambda_t^{(m)}(\theta)$ in (5.36) at $t = 1, \cdots, N$ for each $y^{(m)}$ in Y_B, and let $b^{(m)} \leftarrow \min_{1 \le t \le N} \Lambda_t^{(m)}$
        Split $Y_B = Y_{B,1} \cup Y_{B,2}$, where $y^{(m)}$ in $Y_{B,1}$ if $b^{(m)} < b$, and otherwise in $Y_{B,2}$
        if using VI update **then**
            $g_B \leftarrow \frac{1}{M_B} \sum_{y^{(m)} \in Y_{B,2}} \hat{G}^{(m)}(\theta)$, $\hat{G}^{(m)}$ as in (5.37)
        else if using GD update **then**
            $g_B \leftarrow \frac{1}{M_B} \sum_{y^{(m)} \in Y_{B,2}} \hat{F}^{(m)}(\theta)$, $\hat{F}^{(m)}$ as in (5.42)
        end
        if $Y_{B,1}$ not empty **then**
            $g_B \leftarrow g_B + \delta_b \sum_{y^{(m)} \in Y_{B,1}} \partial_{\theta_K} B^{(m)}(\theta)$, $B^{(m)}$ as in (5.73)
        end
        Update $\theta_K \leftarrow \theta_K - \gamma_k g_B$
        Solve $\tilde{\mu}$ as root of (5.72) on Y_B using bisection search
        Update $\mu \leftarrow 0.9\mu + 0.1\tilde{\mu}$
    end
    (Optional) **if** do smoothness regularization of kernel **then**
        $\theta_K \leftarrow \theta_K - \gamma_k \delta_s \partial_{\theta_K} S(\theta_K)$, S as in (5.74)
    end
end
(Optional) **if** do low-rank truncation of kernel **then**
    Rearrange entries in θ_K into Ψ, and $\Psi' \leftarrow \Psi_{(N'+1):N,1:N'}$
    $r \leftarrow$ number of singular values of Ψ' larger than τ_{SVD}
    $V \leftarrow$ first r right singular vectors of Ψ'
    $\Psi \leftarrow \Psi V V^\top$, and rearrange Ψ into θ_K
end
return μ, θ_K

the solution. When batch size is small the solution can develop large variations across batches.

In our algorithm, we use a finite batch size M_B which can be up to a few hundred. In each step of parameter update, we load M_B many training trajectories, called a "batch" and denoted by Y_B. The batches are non-overlapping and loop over the training set, and when one pass of the training set is finished it is called an "epoch". We random shuffle the training set indices at the beginning of each epoch. Following the convection of stochastic optimization literature, the parameter update step size γ_k is called "learning rate". We set a learning rate schedule indexed by the epochs till the maximum number of epochs. The stochastic optimization algorithm is summarized in Algorithm 3.

Inferenece of baseline intensity μ. To be able to infer the baseline μ from data, our idea is to eliminate the variable μ in the process of learning θ_K making use of the first-order condition of the log-likelihood objective. Specifically, from (5.35) we have that, with $\theta = [\mu, \theta_K]$,

$$\partial_\mu \ell^{(m)}(\theta) = \sum_{t=1}^{N} h \left(\frac{y_t^{(m)}}{1 - e^{-h\Lambda_t^{(m)}(\theta)}} - 1 \right),$$

and for fixed θ_K, one can solve for the value of μ by the following 1D non-linear equation

$$0 = \sum_{m,\, y^{(m)} \in Y_B} \partial_\mu \ell^{(m)}(\theta) = \sum_{m,\, y^{(m)} \in Y_B} \sum_{t=1}^{N} h \left(\frac{y_t^{(m)}}{1 - e^{-h\Lambda_t^{(m)}(\theta)}} - 1 \right). \tag{5.72}$$

The equation can be solved by *bisection search* due to the monotonicity of the function $1 - e^{-h\Lambda_t^{(m)}(\theta)}$ with respect to μ, recalling that $\Lambda_t^{(m)}(\theta) = \mu + \langle \eta_t^{(m)}, \theta_K \rangle$.

Strictly speaking, the first-order condition corresponds to averaging over all the M training trajectories, and in (5.72) we use the batches average to approximated solve for μ given the current θ_K and update the baseline more diligently. In our algorithm, we update μ by a convex combination of the current μ with the solution $\tilde{\mu}$ from the batch Y_B so as to implementing an average over the batches. The initial value μ_0 of μ is set to be the empirical frequency of having an event averaged over the time horizon, i.e., $\mu_0 = \sum_{m=1}^{M} n^{(m)}/(MNh)$ where $n^{(m)} = \sum_{t=1}^{N} y_t^{(m)}$ is the count of events on $1 \le t \le N$ of the m-th trajectory. We observe the numerical convergence of this scheme to learn μ in practice, see Figure D.1(b).

Constraint and regularization of the kernel. Motivated by Assumption 5.4(ii), we inforce the lower-boundedness of the model intensity $\Lambda_t^{(m)} \geq b$ for all $1 \leq t \leq N$ and all training trajectories, where the minimum intensity threshold b is an algorithmic parameter. We introduce a (per-trajectory) barrier function $B^{(m)}$ to penalize the violations, defined as

$$B^{(m)}(\theta) = \sum_{t=1}^{N} \mathbf{1}_{\{\Lambda_t^{(m)}(\theta)<b\}} l_b(\Lambda_t^{(m)}(\theta)), \tag{5.73}$$

and is weighted by a penalty strength factor δ_b. The barrier function $l_b : \mathbb{R} \to \mathbb{R}$ is differentiable, and then $\partial_{\theta_K} B^{(m)}(\theta) = \sum_{t=1}^{N} \mathbf{1}_{\{\Lambda_t^{(m)}(\theta)<b\}} l_b'(\Lambda_t^{(m)}(\theta))\eta_t^{(m)}$. In experiments, we find that several choices of l_b can work well, and in our experiments we use (i) log-barrier $l_b(x) = -b\log(x/b)$ or (ii) quadratic barrier $l_b(x) = \frac{1}{2b}(x-b)^2/0.1$.

The underlying continuity of the kernel function suggests the low-rankness of the kernel matrix, as has been explained in Section 5.4.4. To leverage such prior structural information, we can apply a low-rank truncation of the learned kernel matrix by the stochastic optimization using the submatrix trick in Section 5.4.4, see the end of Algorithm 3. We only keep singular vectors for which the singular value is greater than τ_{SVD}, and this threshold is an algorithmic parameter.

Another optional regularization of the kernel matrix is by regularizing the variation of the kernel matrix Ψ across time. Specifically, writing θ_K into the form of matrix Ψ, one can adopt the smoothness penalty

$$S(\theta_K) = \frac{1}{2h^2}\left(\sum_{i=-N'+1}^{N-1}\sum_{l=1}^{N'}(\Psi_{i,l} - \Psi_{i+1,l})^2 + \sum_{i=-N'+1}^{N}\sum_{l=1}^{N'-1}(\Psi_{i,l} - \Psi_{i,l+1})^2\right) \tag{5.74}$$

weighted by a strength factor δ_s. In our algorithm, we update the kernel using the gradient of $\partial_{\theta_K} S$ after each epoch. The choices of the parameters b, δ_b, δ_s together with the optimization hyperparameters depend on the problem and will be detailed in our experiments (Appendix D.3).

5.6.2 On-network case

The methodology in Section 5.6.1 can be extended to the on-network model, implementing the VI and GD inference as has been introduced in Section 5.5.4. The setup of batch-based stochastic optimization, the constraint enforcement and regularization of the kernel matrices are similar, and a main difference lies in the learning of baseline vector $\boldsymbol{\mu}$ which again we utilize the first-order condition. The algorithm is summarized in Algorithm 5 in the Appendix, and we give technical details of the extensions below.

Similar to (5.72), we propose to solve for $\boldsymbol{\mu}$ given kernel θ_K by requiring the derivative of the log-likelihood with respect to each $\mu(u)$ to vanish. That is, with $\theta = [\boldsymbol{\mu}, \theta_K]$ and again averaged on a batch of trajectories Y_B,

$$\begin{aligned} 0 &= \sum_{m,\mathbf{y}^{(m)}\in Y_B} \frac{\partial \ell^{(m)}[\theta]}{\partial \mu(u)} \\ &= \sum_{m,\mathbf{y}^{(m)}\in Y_B} \sum_{t=1}^{N} \left(h\left(\frac{\bar{y}_t^{(m)}}{1-e^{-h\bar{\Lambda}_t^{(m)}[\theta]}} - 1 \right) - \frac{\bar{y}_t^{(m)}}{\bar{\Lambda}_t^{(m)}[\theta]} + \frac{y_t^{(m)}(u)}{\Lambda_t^{(m)}[\theta](u)} \right), \quad \forall u \in \mathcal{V}, \end{aligned}$$

and we recall that $\Lambda_t^{(m)}[\theta](u) = \mu(u) + \langle \eta_{t,u}^{(m)}, \theta_K \rangle$. These V equations are non-linear because all three terms inside the summation involve $\boldsymbol{\mu}$ through $\bar{\Lambda}_t^{(m)}$ and $\Lambda_t^{(m)}$. To simplify the algorithm, we propose a Gauss-Seidel-type iteration: given the current $\boldsymbol{\mu}$ and θ_K fixed, we solve for $\tilde{\mu}(u)$ for each $u \in \mathcal{V}$ by the equation

$$0 = \sum_{m,\mathbf{y}^{(m)}\in Y_B} \sum_{t=1}^{N} \left(h\left(\frac{\bar{y}_t^{(m)}}{1-e^{-h\bar{\Lambda}_t^{(m)}[\boldsymbol{\mu},\theta_K]}} - 1 \right) - \frac{\bar{y}_t^{(m)}}{\bar{\Lambda}_t^{(m)}[\boldsymbol{\mu},\theta_K]} + \frac{y_t^{(m)}(u)}{\tilde{\mu}(u) + \langle \eta_{t,u}^{(m)}, \theta_K \rangle} \right). \tag{5.75}$$

These V equations are then de-coupled, each one is a 1D non-linear equation and bisection search can be applied due to the monotonicity with respect to $\tilde{\mu}(u)$. After solving the vector

$\{\tilde{\mu}(u)\}_{u\in\mathcal{V}}$ on a batch, we do a convex combination with the current $\boldsymbol{\mu}$ same as before. The vector μ is initialized by $\mu_0(u) = \sum_{m=1}^{M} n^{(m)}(u)/(MNh)$ where $n^{(m)}(u) = \sum_{t=1}^{N} y_t^{(m)}(u)$ is the count of events happening at location u along the m-th trajectory. We empirically observe the convergence of our iteration scheme to learn $\boldsymbol{\mu}$ as shown in Figure D.4(b).

We also extend the constraint and regularizations on the time-only kernel to the spatial-temporal kernel on network. Specifically, the penalty to inforce lower boundedness of the model intensities applies to each node, i.e.

$$B^{(m)}[\theta] = \sum_{t=1}^{N}\sum_{u\in\mathcal{V}} \mathbf{1}_{\{\Lambda_t^{(m)}[\theta](u)<b\}} l_b(\Lambda_t^{(m)}[\theta](u)). \tag{5.76}$$

The optional low-rank truncation is applied to the tall concatenated Ψ matrix across all pair of nodes, as has been explained in Section 5.5.4. The optional smoothness regularization is by applying the penalty S in (5.74) to the time-only kernel on each pair of nodes.

Depending on the problem, e.g. the size of the problem namely N, N' and the graph size V, as well as the inference approach (GD or VI), we may need different parameters b, δ_b, δ_s as well as optimization hyperparameters like batch size and learning rate schedule. The choices will be detailed in the experiments (Appendix D.3).

5.7 Experiments

In this section, we compare the proposed model with various baselines on simulated and real data.

5.7.1 Simulated event data: Time-only

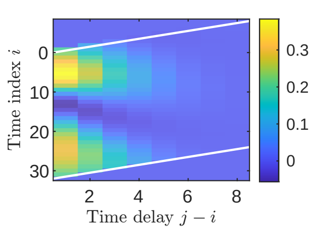
Dataset. For time-only event data, we set a time horizon $[-\tau_{\max}, T]$ with $\tau_{\max} = 4$ and $T = 16$. The model adopts a latent influence kernel function $k(t', t)$ defined as

$$k(t', t) = \sum_{j=1}^{13} 0.3(2)^{-j}(\cos(2 + 1.3\pi(j+1)((t'-9)/15)) + 0.6)e^{-8((t-t')j)^2/25}, \quad t > t', \tag{5.77}$$

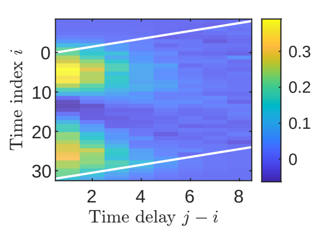
which, after specifying N and N', will induce a kernel matrix K in (5.23). In this example, the influence kernel matrix K will have both positive and negative entries, meaning that the history may trigger excitation and inhibition effects between events. In addition, the influence kernel is time-varying, meaning that these effects are non-stationary over time.

We use two types of discrete time grids: (a) $N' = 8, N = 32$, and (b) $N' = 80, N = 320$. The corresponding kernel matrices are visualized in Figure 5.4(a) and Figure 5.6(a) respectively in the form of Ψ defined as in (5.32). We use $\mu = 0.2$, and the event data are simulated by the law of a Bernoulli process as introduced in Section 5.3.3. The training and testing sets consist of $16,000$ and 500 trajectories, respectively.

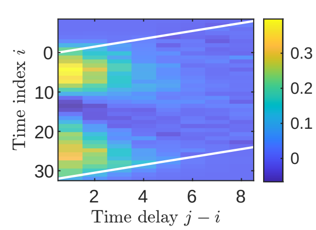
Method. We compute our model following Algorithm 3 as introduced in Section 5.6.1, and the methods using VI and GD are called `TULIK-VI` and `TULIK-GD` respectively. The choice of training hyperparameters is detailed in Appendix D.3.1. The training dynamic



(a) true kernel



(b) `TULIK-VI`



(c) `TULIK-GD`

Figure 5.4: Recovered kernels on simulated time-only event data with kernel function defined in (5.77), $N' = 8$ and $N = 32$. (a) The true kernel. (b) The kernel recovered by `TULIK-VI`. (c) The kernel recovered by `TULIK-GD`. The white lines indicate the trainable parameters.

Table 5.1: Relative Errors (REs) of learned μ, kernel matrix, and probability predictions, on simulated time-only event data with the kernel function defined in (5.77), $N' = 8$ and $N = 32$. We report means of REs over 10 replicas and standard deviations are in parentheses. For probability predictions, the best performance is in boldface. Note that μ is a scalar in the time-only example and its ℓ_1, ℓ_2 and ℓ_∞ errors are the same.

RE $\times 10^{-2}$	time-only event data with discrete time grids $N' = 8$ and $N = 32$								
	μ		kernel		prediction				
	TULIK-VI	TULIK-GD	TULIK-VI	TULIK-GD	TULIK-VI	TULIK-GD	GLM-L	GLM-S	HP-E
ℓ_1	0.40 (0.38)	0.47 (0.41)	16.37 (0.82)	18.39 (0.86)	**2.85** (0.11)	3.20 (0.15)	3.93 (0.19)	6.13 (0.19)	37.08 (0.65)
ℓ_2	–	–	12.07 (0.70)	13.46 (0.66)	**3.96** (0.16)	4.43 (0.20)	5.25 (0.21)	7.99 (0.14)	53.06 (1.14)
ℓ_∞	–	–	11.26 (1.40)	12.35 (1.32)	**6.02** (0.35)	6.73 (0.38)	8.09 (0.41)	12.85 (0.30)	92.83 (2.49)

Table 5.2: Relative Errors (REs) of learned μ, kernel matrix, and probability predictions, on simulated time-only event data with the kernel function defined in (5.77), $N' = 80$ and $N = 320$. The format are the same as in Table 5.1.

RE $\times 10^{-2}$	time-only event data with discrete time grids $N' = 80$ and $N = 320$								
	μ		Kernel		Prediction				
	TULIK-VI	TULIK-GD	TULIK-VI	TULIK-GD	TULIK-VI	TULIK-GD	GLM-L	GLM-S	HP-E
ℓ_1	1.46 (0.80)	0.72 (0.37)	14.49 (0.53)	18.23 (0.67)	**3.34** (0.23)	3.89 (0.19)	37.11 (0.59)	27.96 (0.49)	72.88 (1.74)
ℓ_2	–	–	11.58 (0.50)	13.39 (0.52)	**4.50** (0.19)	5.31 (0.25)	51.65 (0.48)	39.79 (0.79)	149.82 (5.44)
ℓ_∞	–	–	17.50 (1.52)	17.04 (2.27)	**8.64** (0.59)	9.44 (0.50)	98.82 (0.91)	101.33 (2.57)	393.29 (17.25)

on one training set, including the evolution of the training (negative) log-likelihood and the convergence of the learned μ are illustrated in Figure D.1.

For comparison, we consider three alternative baselines: (1) Generalized Linear Model (GLM) [129] with linear link function (GLM-L), where the link function is an identity

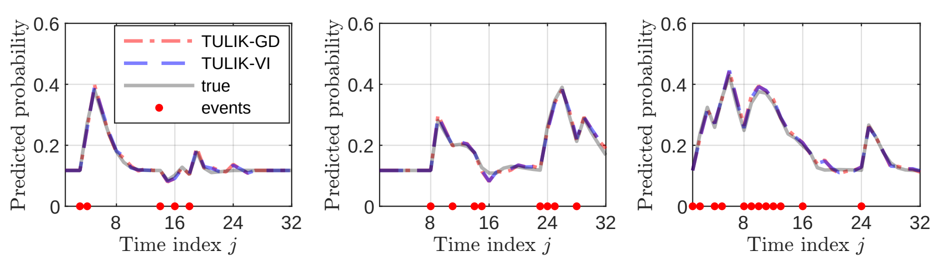
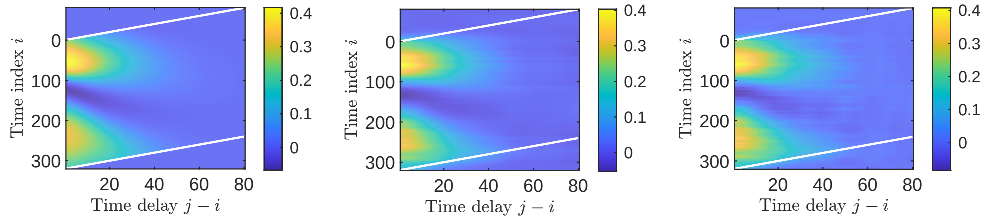


Figure 5.5: The probability predictions on 3 randomly selected testing sequences. Data is the simulated time-only event data with kernel function defined in (5.77), $N' = 8$ and $N = 32$.



(a) true kernel (b) `TULIK-VI` (c) `TULIK-GD`

Figure 5.6: Recovered kernels on simulated time-only event data with kernel function defined in (5.77), $N' = 80$ and $N = 320$. Same plots as in Figure 5.4. The recovered kernel is after a low-rank truncation.

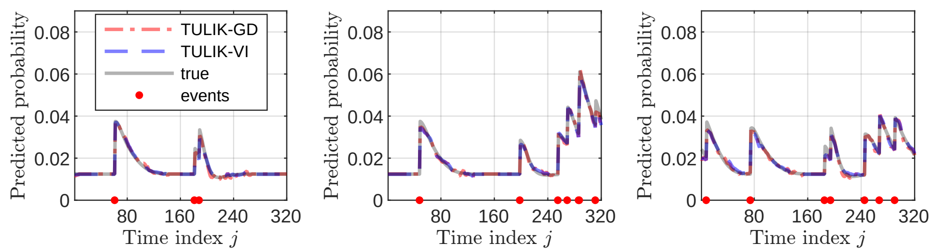


Figure 5.7: Same plots as in Figure 5.5. Data is the simulated time-only event data with kernel function defined in (5.77), $N' = 80$ and $N = 320$.

function with range $[0, 1]$; (2) GLM with Sigmoid link function (`GLM-S`); (3) Continuous-time Hawkes Process with Exponential kernel (`HP-E`) [131, 116, 132], see more in Section 5.2.1. The implementation details of the alternative baselines can be found in Appendix D.3.1.

Results. For data with $N' = 8$ and $N = 32$, both `TULIK-VI` and `TULIK-GD` can learn the true model parameters (μ and kernel matrix) well, resulting in good prediction performance, as shown quantitatively in Table 5.1. The proposed TULIK models outperform the other three baselines, where `TULIK-VI` gains slightly better performance than `TULIK-GD`. The recovered kernel matrices are plotted in Figure 5.4, showing a good agreement with the true kernel. The predicted probability is also close to the ground truth, as reflected by the almost overlapping curves in Figure 5.5.

For the data with $N' = 80$ and $N = 320$, which results in much higher dimensional trainable parameters, the proposed TULIK model can still achieve reasonable recovery of

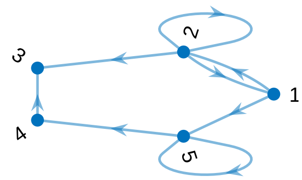


Figure 5.8: The visualization for the network structure of the on-network event data in Section 5.7.2.

kernel matrices and μ's and accurate probability predictions. Figure 5.6 shows that both `TULIK-VI` and `TULIK-GD` produce correct shapes and magnitudes of kernel matrices, where we applied low-rank truncation to the learned kernel matrix. Figure 5.7 visualizes the probability predictions in good agreement with the ground truth. From the quantitative errors in Table 5.2, our proposed model again achieves the best relative prediction errors, and VI slightly outperforms GD.

Example of stationary process. As has been introduced in Section 5.4.4, the proposed TULIK model can be applied to stationary processes with time uncertainty, where the kernel is time-invariant. This is a special case of the more general time-varying kernel. We provide experiments on such a simplified case in Appendix D.3.2. The results show that the proposed approach successfully recovers the model parameters and outperforms the alternative baselines.

5.7.2 Simulated event data: on network

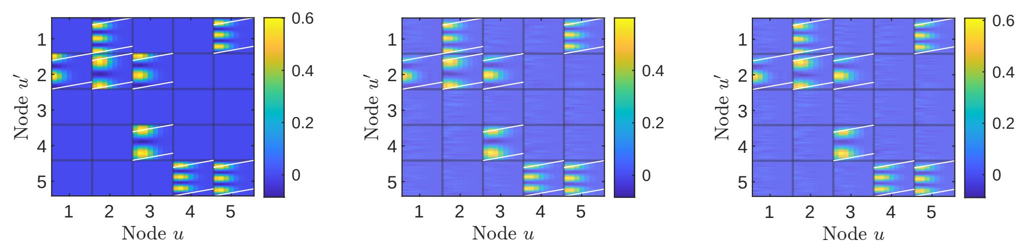
Dataset. For the on-network event data, we set a time horizon $[-\tau_{\max}, T]$ with $\tau_{\max} = 0.8$ and $T = 3.2$. The network has $V = 5$ nodes, in which 8 pairs of successors and predecessors are randomly sampled from $\mathcal{V}$ to form 8 directed edges. The network structure is visualized in Figure 5.8. For a directed edge $u' \to u$, the marginal kernel (temporal pattern) is defined as

$$k(t', t; u', u) = 0.35 \left(\cos\left(\omega_{u',u}(t+2)\right) + 0.75\right) e^{-20\left(t-t'-h_{u',u}\right)^2}, \tag{5.78}$$

where $\omega_{u',u}$ independently follows $\mathrm{Uniform}(2,6)$ and $h_{u',u}$ is independently drawn from $\mathrm{Uniform}(0,0.2)$. Given $N'=8$ and $N=32$, we can induce the four-way kernel tensor Ψ as in (5.60) and visualize it in Figure 5.9(a). The non-zero block matrix on u'-th row and u-th column corresponds to a directed edge $u' \to u$. The zero block matrices indicate disconnected pairs of nodes.

Given the kernel and $\mu(u)$ independently sampled from $\mathrm{Uniform}(0.25, 0.35)$ for $u \in \mathcal{V}$, we can generate on-network Bernoulli process data as in Section 5.5.3. The training data and testing data contains 40000 and 500 trajectories, respectively.

Method. We train the proposed model using Algorithm 5 (in Appendix) as introduced in Section 5.6.2. The choice of training hyperparameters is discussed in Appendix D.3.3.



(a) TULIK-GD (b) TULIK-GD (c) TULIK-GD

Figure 5.9: Recovered kernel tensors on simulated on-network event data with kernel function defined in (5.78), $V=5$, $N'=8$ and $N=32$. (a) The true kernel. (b) The kernel recovered by TULIK-VI. (c) The kernel recovered by TULIK-GD. The learned kernels are after low-rank truncation. The (u',u)-th block in the plot shows the kernel matrix $[\Psi_{i,l}(u',u)]_{i,l}$, for $u',u=1,\cdots,5$. The white slashes box out the trainable parameters of the kernel matrices corresponding to the directed edges.

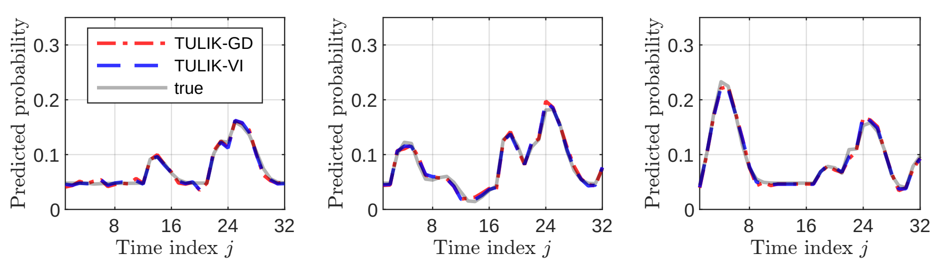


Figure 5.10: The probability predictions on 3 randomly selected testing sequences on one node (node 3 in the graph in Figure 5.8). Data is the simulated on-network event data with kernel function defined in (5.78), $V=5$, $N'=8$ and $N=32$.

Table 5.3: Relative Errors (REs) of learned μ, kernel tensor, and probability predictions, on simulated on-network event data with the kernel function defined in (5.78), $V = 5$, $N' = 8$ and $N = 32$. We report means of REs over 10 replicas and standard deviations are in parentheses. For probability predictions, the best performance is in boldface.

RE $\times 10^{-2}$	on-network event data with discrete time grids $N' = 8$ and $N = 32$								
	μ		Kernel		Prediction				
	TULIK-VI	TULIK-GD	TULIK-VI	TULIK-GD	TULIK-VI	TULIK-GD	GLM-L	GLM-S	HP-E
ℓ_1	7.28 (0.52)	6.04 (0.57)	19.11 (0.57)	21.07 (0.44)	**6.36** (0.09)	7.47 (0.09)	9.29 (0.11)	10.29 (0.11)	31.55 (0.45)
ℓ_2	7.40 (0.56)	6.19 (0.60)	13.49 (0.33)	14.70 (0.24)	**7.42** (0.09)	8.76 (0.09)	11.32 (0.12)	13.94 (0.14)	45.97 (0.41)
ℓ_∞	8.15 (0.82)	7.09 (0.85)	17.22 (2.00)	17.22 (2.16)	**9.62** (0.07)	11.09 (0.19)	15.01 (0.15)	25.97 (0.53)	72.12 (0.33)

The convergence of the training (negative) log-likelihood and the learned μ vector is illustrated in Figure D.4. We compare the proposed method with the baselines including GLM-L, GLM-S, and HP-E. The implementation details of the alternative baselines in the on-network setting are given in Appendix D.3.3.

Results. Both TULIK-VI and TULIK-GD can learn the vector μ and the kernel tensor, leading to accurate prediction as quantitatively revealed in Table 5.3. Similar to the time-only examples, the proposed TULIK models yield smaller prediction errors than the other baselines, where TULIK-VI is slightly better than TULIK-GD. After optimization loops, we apply low-rank truncation to the matrix $\boldsymbol{\Psi}$, and the learned kernel tensors are visualized in Figure 5.9. It can be seen that the kernel is well recovered, and in particular, the learned kernel reveals the correct causal network: when (u', u) is a missing (directed) edge in the network (Figure 5.8), the corresponding block in the learned kernel also takes almost zero values. The predicted probability is also consistent with the ground truth, as shown in Figure 5.10 (the prediction is successful on all the nodes, where the plots on one node are shown).

5.8 Real data analysis

5.8.1 Sepsis-Associated-Derangements (SADs) data

Dataset. We consider the data is publicly available through the 2019 PhysioNet Challenge [128], which consists of hospitalization records of Intensive Care Unit (ICU) patients. Each patient record includes three main types of variables: vital signs, laboratory (lab) measurements, and demographic information. The goal is to use this data to predict the onset of sepsis—a condition that arises when the body's response to infection causes tissue damage, organ failure, or death, and can quickly become life-threatening [160].

We use a processed version of this dataset, referred to as SADs [161, 162, 163], which was created by grouping all raw observations—including vital signs and lab measurements—into 13 medical indices using a standard procedure [164]. These indices represent common physiological derangements associated with sepsis and are treated as nodes in our graph. Table 5.4 lists the abbreviations and names of the indices. Vital signs are recorded hourly, introducing a natural one-hour time uncertainty in the data. Lab test results are typically returned within 24 hours, leading to a larger time uncertainty, meaning that the test result typically means patient condition around 24 hours around the test time; here, for simplicity, we assume the time uncertainty for the test is also 1 hour. Appendix D.3.4

Table 5.4: Selected 13 medical indices in SADs data

Node	Abbreviation	Full Name
1	RI Kernel	Renal Injury
2	EI	Electrolyte Imbalance
3	OCD	Oxygen Carrying Dysfunction
4	Shock	Shock
5	DCO	Diminished Cardiac Output
6	CI	Coagulopathy
7	Chl	Cholestatsis
8	HI	Hepatocellular Inury
9	OD(Lab)	Oxygenation Dysfunction (Lab)
10	Inf(Lab)	Inflammation (Lab)
11	OD(vs)	Oxygenation Dysfunction (vital sign)
12	Inf(vs)	Inflammation (vital sign)
13	Sepsis	Sepsis

Table 5.5: Sepsis prediction: True Positive Rate (TPR), True Negative Rate (TNR), and Balanced Accuracy (BA) from `TULIK-VI`. The standard deviations of TPR, TNR, and BA over 10 training replicas are in parentheses. The highest metrics are bolded. The training data and testing data consist of 1070 and 71 patients trajectories, respectively.

	`TULIK-VI`	`TULIK-VI` (Stationary)	`GLM-L`	`GLM-S`	`HP-E`
TPR	**0.6842** (0.0000)	0.4632 (0.0211)	0.5789 (0.0000)	0.6316 (0.0000)	0.0000 (0.0000)
TNR	**0.6904** (0.0058)	0.4519 (0.0197)	0.5769 (0.0000)	0.6731 (0.0000)	0.0192 (0.0000)
BA	**0.6873** (0.0029)	0.4575 (0.0203)	0.5779 (0.0000)	0.6523 (0.0000)	0.0096 (0.0000)

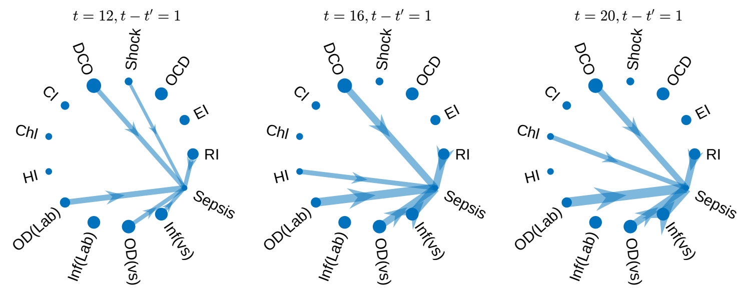
contains details on data pre-processing.

Given the 13 medical indices, the processed SADs data can be viewed as a point process on a network with $V = 13$ nodes, with time uncertainty of 1 hour. Each patient's data trajectory is a Boolean process on this network, which we refer to as a "signal" that takes values 0 or 1 at each node ($y_t(u) \in 0, 1$). We consider the 24 hours after the patient is admitted to the ICU and set the influence time lag $\tau_{\max}$ to 4 hours. As a result, the time horizon is $[-\tau_{\max}, T] = [-4, 20]$, covering a total of 24 hours. The time grid has $N = 20$ and $N' = 4$, with unit uncertainty of one hour.

The node, "Sepsis," marks the sepsis onset for each patient, based on the Sepsis-3 definition. For time points more than 6 hours before the onset, the observation on the sepsis node is always zero; after the onset of sepsis, the observation remains positive. To support early prediction of sepsis onset, the positive labels are shifted 6 hours earlier, meaning that prediction within 6 6-hour period before and after the initial onset of sepsis is meaningful, consistent with the definition in Physionet Challenge [128].

In the experiments below, the processed dataset contains 1,141 patients' record, all of whom were diagnosed with sepsis but with varying onset times. We randomly select 1,070 trajectories for training and 71 trajectories for testing.

Method. We use `TULIK-VI` to the SADs data, as it demonstrates more stable training behavior compared to `TULIK-GD` in our case. As in Section 5.7.2, we compare the pro-



(a) (b) (c)

Figure 5.11: Temporally evolving causal networks; for clear visualization purposez, we illustrate top 5 strongest edges. The time lag $t - t'$ is fixed at 1, which indicates one-hour effects. The future time t increases from 12 to 20 by a gap 4. The radii of nodes are proportional to the base intensities on nodes; Blue arrows represent excitations, whose widths are proportional to the magnitudes. The marker size and line widths are comparable across plots.

posed model with baseline methods, including `GLM-L`, `GLM-S`, and `HP-E`. Experimental details are provided in Appendices D.3.4 and D.3.4.

Results. The performance of all methods are reported in Table 5.5. We evaluate True Positive Rate (TPR), True Negative Rate (TNR), and Balanced Accuracy (BA), on the testing data. These evaluation metrics are standard for sepsis prediction [128]. As shown in the table, `TULIK-VI` consistently outperforms the baseline methods.

In particular, `TULIK` allowing for non-stationary modeling, significantly outperforms its stationary counterpart, indicating that the sepsis data likely contains substantial non-stationarity, effectively captured by our model. `TULIK`, with its modeling of time uncertainty and flexible triggering pattern modeling, also outperforms methods that do not account for time uncertainty, such as `GLM-L` and `GLM-S`, which use standard link functions, and continuous time `HP-E`, which employs a parametric exponential decaying triggering kernel. These results highlight the importance of incorporating both time uncertainty and nonparametric triggering structures in modeling sepsis dynamics.

To further interpret the results, we visualize the temporally evolving causal networks extracted from the kernel tensor learned by `TULIK-VI` in Figure 5.11, which indicates the

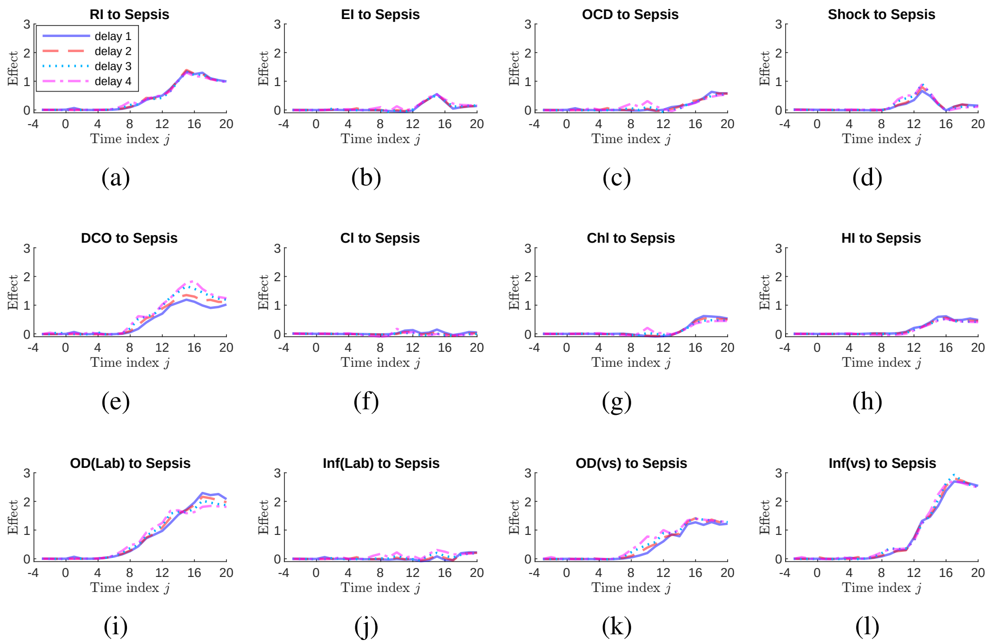


Figure 5.12: The effects from all medical indices to the sepsis onset over time across different delays.

inferred Granger causal relationships between medical indices. Node radii are proportional to the estimated baseline intensities $\mu(u)$; arrows represent excitatory effects, with arrow widths indicating the magnitude of influence. We show the top five strongest edges in each time snapshot of the network.

The learned network structure aligns with medical knowledge. For example, the node "DCO" (Diminished Cardiac Output) consistently exhibits a strong excitatory influence on sepsis onset, an observation well-established in sepsis research [163]. Additionally, as time progresses, more medical indices show a strong influence on sepsis onset, reflecting the development of multiple organ dysfunction syndrome in severe cases. Figure 5.12 shows that the effects of biological processes are nonlinear, suggesting that medical indices cannot deteriorate indefinitely, nor can their influence on sepsis onset grow without bound.

Overall, the real data results show that `TULIK-VI` is a promising framework for uncovering and hypothesizing the triggering effect between medical indices for sepsis prediction.

5.8.2 Burglary crime data

Dataset. To validate our method on another data, we consider a proprietary dataset provided by the Atlanta Police Department (APD), including over 213,000 burglary events recorded from the beginning of 2013 to the end of 2019, with timestamps and geographic coordinates (longitude and latitude). We focus on downtown area of Atlanta, defined by the longitude range $[-84.400, -84.375]$ and latitude range $[33.745, 33.770]$. Figure 5.13 displays the spatial distribution of burglary events in this area. Following the setup in [129], we uniformly divide the downtown area into 16 sub-regions, resulting in $V = 16$ nodes for the point process model. We treat each 24-hour period, from 12:00 AM to 11:59 PM, as one trajectory. The time horizon is defined as $(-\tau_{\max}, T] = (-8, 16]$, incorporating a one-hour uncertainty. The time grid includes $N = 16$ and $N' = 8$ intervals, each representing one hour. The processed dataset consists of 2,548 daily trajectories. We randomly select 2,250 for training and use the remaining 298 for testing.

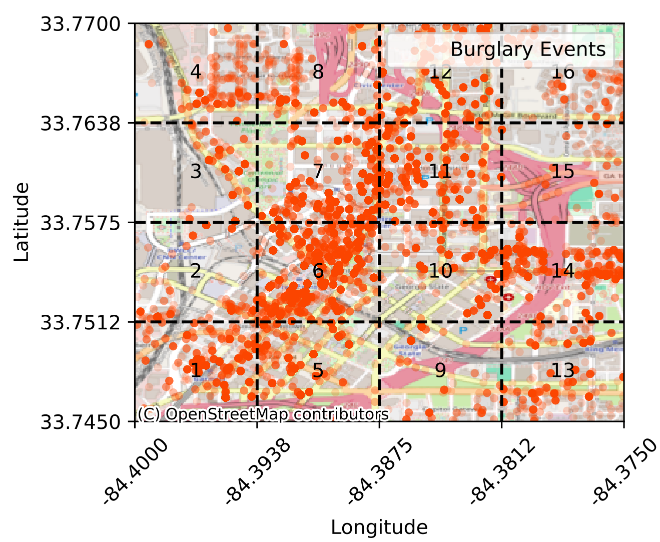


Figure 5.13: The distribution of the burglary crime events from 2013 to 2019 in Atlanta downtown area.

Table 5.6: Crime data modeling in downtown Atlanta: TPR, TNR and BAs from TULIK-VI. The time horizon is 1 day with uncertainty 1 hour. The standard deviations of TPR, TNR and BA over 10 training replicas are in parentheses. The largest metrics are bolded.

	TULIK-VI	TULIK-VI (Stationary)	GLM-L	GLM-S	HP
TPR	**0.6705** (0.0008)	0.6686 (0.0009)	0.6186 (0.0004)	0.6594 (0.0002)	0.4533 (0.0031)
TNR	**0.6710** (0.0010)	0.6683 (0.0008)	0.6187 (0.0004)	0.6532 (0.0000)	0.4733 (0.0098)
BA	**0.6707** (0.0002)	0.6684 (0.0004)	0.6187 (0.0003)	0.6563 (0.0001)	0.4633 (0.0036)

Method. We compared `TULIK-VI` with `GLM-L`, `GLM-S`, and `HP-E`. Details of the hyperparameter tuning procedure are provided in Appendix D.3.5.

Results. Table 5.6 presents the prediction performance of all methods. We report TPR, TNR, and BA on the testing data. For the burglary crime data, TPR and TNR are computed by averaging across all nodes for $t \in (0, 16]$, treating event occurrences as positives and non-events as negatives. `TULIK-VI` achieves noticeably higher prediction accuracy compared to the baseline methods.

Again, we observe that `TULIK` allowing for non-stationary modeling, outperforms its stationary counterpart. However, in this example, the performance gain is less pronounced than in the sepsis prediction case, suggesting that the burglary data may exhibit less non-stationarity. Nevertheless, `TULIK` still outperforms the alternative methods—`GLM-L`, `GLM-S`, and `HP-E`, the latter of which employs a parametric triggering kernel, highlighting the value of incorporating time uncertainty and flexible triggering structures.

5.9 Discussion

The work can be extended in several directions. First, while we focused on the unit-time uncertainty case in our algorithm development, our method can be extended to handle the arbitrary event uncertainty case in Section 5.3.2, for both the time-only and on-network scenarios. Second, it would be useful to allow the baseline intensity μ to vary with time t, and this can be handled under our discrete-time framework by introducing more learnable parameters. Meanwhile, it would be interesting to relax the requirement that each uncertainty window only contains one event (Assumption 5.1), which will lead to a Poisson process instead of a Bernoulli process and calls for new development of the time-uncertainty model. Finally, one can apply the approach to more real data applications, and the practical challenges may call for additional developments of the computational techniques, e.g., the use of the barrier to enforce positive intensity and the setting of optimization hyperparameters.

Appendices

APPENDIX A

LOW-RANK IMAGE MONITORING

A.1 Overlapping weighted Cramér-von Mises (CvM) estimator

The CvM estimator is proposed by [165]. An expedient strategy for determining the batch size m could ensure approximate independence among the batch means. Any applicable statistical test for independence can be employed for this purpose. In our study, we adopt a fixed batch size of $m = 50$ across all experimental configurations. For the purpose of self-containedness, we show the computation process of CvM estimator in Algorithm 4.

A.2 Proofs

Proof of Theorem 2.4.4. We begin by studying the statistical behaviors of in-control and out-of-control $\gamma_{i,t}$ separately.

- For the in-control case, Lemma 2.4.2 implies that $\gamma_{i,t} = \mathrm{o}(\sqrt{p_1 p_2})$ almost surely.

Algorithm 4: The overlapping weighted CvM estimator

Input: In-control monitoring statistics $\{T_t : t = 1, \ldots, n\}$, batch size m.

1: For $i = 1, \ldots, n - m + 1$, calculate

$$C_i = \frac{1}{m} \sum_{j=1}^{m} g\left(\frac{j}{m}\right) \frac{j^2}{m} \left(\bar{T}_{i,j} - \bar{T}_i\right)^2,$$

where the function $g(t) = -24 + 150t - 150t^2$, the partial batch mean $\bar{T}_{i,j} = \frac{1}{j} \sum_{j'=1}^{j} T_{i+j'}$ and the batch mean $\bar{T}_i = \frac{1}{m} \sum_{j=1}^{m} T_{i+j}$.

2: The CvM estimator is the average over all CvM estimators from the batches, namely

$$\Omega_0^2 = \frac{1}{n - m + 1} \sum_{i=1}^{n-m+1} C_i.$$

- For an out-of-control case, Lemma 2.4.3 implies

$$\gamma_{i,t} = \sqrt{p_1 p_2}\bar{\rho}_i + z_i + \mathrm{o}\left(\sqrt{p_1 p_2}\right) + \mathrm{o}_p\left(1\right),$$

where $\mathrm{o}_p\left(1\right)$ represents a random variable Z satisfying $\lim_{p_1,p_2\to\infty}\mathbb{P}\left(|Z|\geq\epsilon\right)=0$ for any positive constant ϵ. Then, we have

$$\frac{\mathbb{E}_1[\gamma_{i,t}]-\mathbb{E}_0[\gamma_{i,t}]}{\sqrt{p_1 p_2}} = \bar{\rho}_i + \mathrm{o}\left(1\right).$$

□

Proof of Theorem 2.4.5. Without specifying the probability measure, we can decompose T_t into the following components.

$$\begin{aligned}
\mathbb{E}_\upsilon[T_t] &= \mathbb{E}_\upsilon\Big[\left(y_t-\mathbb{E}_0[y_t]\right)^\top \mathrm{Cov}_0^{-1}(y_t)\left(y_t-\mathbb{E}_0[y_t]\right)\Big] \\
&= \mathbb{E}_\upsilon\Big[\left(y_t-\mathbb{E}_\upsilon[y_t]+\mathbb{E}_\upsilon[y_t]-\mathbb{E}_0[y_t]\right)^\top \mathrm{Cov}_0^{-1}(y_t)\left(y_t-\mathbb{E}_\upsilon[y_t]+\mathbb{E}_\upsilon[y_t]-\mathbb{E}_0[y_t]\right)\Big] \\
&= \mathrm{tr}\big(\mathrm{Cov}_0^{-1}(y_t)\mathrm{Cov}_\upsilon^{-1}(y_t)\big) + \left(\mathbb{E}_\upsilon[y_t]-\mathbb{E}_0[y_t]\right)^\top \mathrm{Cov}_0^{-1}(y_t)\left(\mathbb{E}_\upsilon[y_t]-\mathbb{E}_0[y_t]\right), \quad \upsilon=0,1.
\end{aligned}$$

Now we examine the difference between the in-control and out-of-control expectations:

$$\begin{aligned}
\mathbb{E}_1[T_t]-\mathbb{E}_0[T_t] &= \mathrm{tr}\big(\mathrm{Cov}_0^{-1}(y_t)\mathrm{Cov}_1^{-1}(y_t)\big) \\
&\quad + \left(\mathbb{E}_1[y_t]-\mathbb{E}_0[y_t]\right)^\top \mathrm{Cov}_0^{-1}(y_t)\left(\mathbb{E}_1[y_t]-\mathbb{E}_0[y_t]\right) - 2r \\
&= \mathrm{tr}\left(\Sigma^{-1}\widetilde{\Sigma}\right) + \delta^\top\Sigma^{-1}\delta - 2r.
\end{aligned}$$

By utilizing the property of the Schur complement, we can express the inverse of Σ in the following form:

$$\Sigma^{-1} = \begin{bmatrix} \Sigma_\beta^{-1} + \Sigma_\beta^{-1}P\left(\Sigma/\Sigma_\beta\right)^{-1}P^\top\Sigma_\beta^{-1} & -\Sigma_\beta^{-1}P\left(\Sigma/\Sigma_\beta\right)^{-1} \\ -\left(\Sigma/\Sigma_\beta\right)^{-1}P^\top\Sigma_\beta^{-1} & \left(\Sigma/\Sigma_\beta\right)^{-1} \end{bmatrix},$$

where $\Sigma/\Sigma_\beta = \Sigma_\gamma - P^\top \Sigma_\beta^{-1} P$ denotes the Schur complement of the block Σ_β within the matrix Σ. For $\operatorname{tr}\left(\Sigma^{-1}\widetilde{\Sigma}\right)$, we have

$$
\Sigma^{-1}\widetilde{\Sigma} = \begin{bmatrix} \Sigma_\beta^{-1} + \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} & -\Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} \\ -\left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} & \left(\Sigma/\Sigma_\beta\right)^{-1} \end{bmatrix} \begin{bmatrix} \Sigma_\beta & \widetilde{P} \\ \widetilde{P}^\top & \widetilde{\Sigma}_\gamma \end{bmatrix}
$$

$$
= \begin{bmatrix} I_r + \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top - \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} \widetilde{P}^\top & \left(\Sigma_\beta^{-1} + \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1}\right) \widetilde{P} - \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} \widetilde{\Sigma}_\gamma \\ -\left(\Sigma/\Sigma_\beta\right)^{-1} P^\top + \left(\Sigma/\Sigma_\beta\right)^{-1} \widetilde{P}^\top & -\left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} \widetilde{P} + \left(\Sigma/\Sigma_\beta\right)^{-1} \widetilde{\Sigma}_\gamma \end{bmatrix}.
$$

Then, its trace becomes

$$
\begin{aligned}
&\operatorname{tr}\left(\Sigma^{-1}\widetilde{\Sigma}\right) \\
&= r + \operatorname{tr}\left(\Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top - \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} \widetilde{P}^\top - \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} \widetilde{P} + \left(\Sigma/\Sigma_\beta\right)^{-1} \widetilde{\Sigma}_\gamma\right) \\
&= r + \operatorname{tr}\left[\left(\Sigma/\Sigma_\beta\right)^{-1}\left(P^\top \Sigma_\beta^{-1} P - \widetilde{P}^\top \Sigma_\beta^{-1} P - P^\top \Sigma_\beta^{-1} \widetilde{P} + \widetilde{\Sigma}_\gamma\right)\right] \\
&= r + \operatorname{tr}\left[\left(\Sigma/\Sigma_\beta\right)^{-1}\left(P^\top \Sigma_\beta^{-1} P - \widetilde{P}^\top \Sigma_\beta^{-1} P - P^\top \Sigma_\beta^{-1} \widetilde{P} + \widetilde{P}^\top \Sigma_\beta^{-1} \widetilde{P} - \widetilde{P}^\top \Sigma_\beta^{-1} \widetilde{P} + \widetilde{\Sigma}_\gamma\right)\right] \\
&= r + \operatorname{tr}\left\{\left(\Sigma/\Sigma_\beta\right)^{-1}\left\{\left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]^\top \left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right] + \widetilde{\Sigma}/\Sigma_\beta\right\}\right\} \\
&= r + \operatorname{tr}\left\{\left(\Sigma/\Sigma_\beta\right)^{-1}\left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]^\top \left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]\right\} + \operatorname{tr}\left[\left(\Sigma/\Sigma_\beta\right)^{-1}\left(\widetilde{\Sigma}/\Sigma_\beta\right)\right],
\end{aligned}
$$

where $\widetilde{\Sigma}/\Sigma_\beta = \widetilde{\Sigma}_\gamma - \widetilde{P}^\top \Sigma_\beta^{-1} \widetilde{P}$. For $\delta^\top \Sigma^{-1} \delta$, we have

$$
\begin{aligned}
&\delta^\top \Sigma^{-1} \delta \\
&= \begin{bmatrix} \delta_\beta^\top & \delta_\gamma^\top \end{bmatrix} \begin{bmatrix} \Sigma_\beta^{-1} + \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} & -\Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} \\ -\left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} & \left(\Sigma/\Sigma_\beta\right)^{-1} \end{bmatrix} \begin{bmatrix} \delta_\beta \\ \delta_\gamma \end{bmatrix} \\
&= \delta_\beta^\top \left[\Sigma_\beta^{-1} + \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1}\right] \delta_\beta \\
&\quad - \delta_\beta^\top \Sigma_\beta^{-1} P \left(\Sigma/\Sigma_\beta\right)^{-1} \delta_\gamma - \delta_\gamma^\top \left(\Sigma/\Sigma_\beta\right)^{-1} P^\top \Sigma_\beta^{-1} \delta_\beta + \delta_\gamma^\top \left(\Sigma/\Sigma_\beta\right)^{-1} \delta_\gamma \\
&= \left\|\Sigma_\beta^{-1/2} \delta_\beta\right\|^2 + \left\|\left(\Sigma/\Sigma_\beta\right)^{-1/2}\left(P^\top \Sigma_\beta^{-1} \delta_\beta + \delta_\gamma\right)\right\|^2.
\end{aligned}
$$

Now, we substitute $\operatorname{tr}\left(\Sigma^{-1}\widetilde{\Sigma}\right)$ and $\delta^\top\Sigma^{-1}\delta$ back into the expression for the shift size of T_t. Finally, we have

$$
\begin{aligned}
&\mathbb{E}_1[T_t] - \mathbb{E}_0[T_t] \\
&= \operatorname{tr}\left(\Sigma^{-1}\widetilde{\Sigma}\right) + \delta^\top\Sigma^{-1}\delta - 2r \\
&= \operatorname{tr}\left[(\Sigma/\Sigma_\beta)^{-1}\left(\widetilde{\Sigma}/\Sigma_\beta\right)\right] - r + \operatorname{tr}\left\{(\Sigma/\Sigma_\beta)^{-1}\left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]^\top\left[\Sigma_\beta^{-1/2}\left(P - \widetilde{P}\right)\right]\right\} \\
&\quad + \left\|\Sigma_\beta^{-1/2}\delta_\beta\right\|^2 + \left\|(\Sigma/\Sigma_\beta)^{-1/2}\left(P^\top\Sigma_\beta^{-1}\delta_\beta + \delta_\gamma\right)\right\|^2.
\end{aligned}
$$

□

APPENDIX B

NEURAL-NETWORK BASED CHANGE-POINT DETECTION

B.1 Additional Experiment Details

B.1.1 Details on Baselines

In this section, we introduce the technical details of the compared baselines. As a general scheme for change-point detection, the practitioners need to calculate the statistics at each time and compare the statistics with a pre-specified threshold. When the sequential statistics exceed a threshold at some time t, an alarm will be raised to claim a change, and the stopping time τ will be set to the value t.

Online Neural Network Classification (ONNC) ONNC is an online change-point detection method using neural networks proposed by [64]. The authors construct reference data stack from historical data, potentially affecting their detection efficiency since the reference stack can be (partially) positive class when it contains or locates after the change point. To be concrete, at time t, the test stack is $\{x_i : i = t-w+1, \ldots, t\}$ and the reference stack is $\{x_i : i = t-w-h+1, \ldots, t-h\}$, where w is the training/testing sample size and h is the lag between test and reference stacks. The neural network of ONNC, denoted by $g_\theta^{\mathrm{ONNC}}(x)$ with the parameter θ, is trained with the logistic loss function to learn the labels directly. Since they do not implement training and testing split, the test statistic at time t is calculated as

$$\eta_t^{\mathrm{ONNC}} = \frac{1}{w}\left[\sum_{i=t-w+1}^{t} \log\left(\frac{g_{\hat{\theta}_t}^{\mathrm{ONNC}}(x_i)}{1-g_{\hat{\theta}_t}^{\mathrm{ONNC}}(x_i)}\right) + \sum_{i=t-h-w+1}^{t-h} \log\left(\frac{1-g_{\hat{\theta}_t}^{\mathrm{ONNC}}(x_i)}{g_{\hat{\theta}_t}^{\mathrm{ONNC}}(x_i)}\right)\right],$$

where $\hat{\theta}_t$ is the trained parameter for ONNC at time t. They report their detection scores by iteratively compute $\bar{\eta}_t^{\mathrm{ONNC}} = \bar{\eta}_{t-w}^{\mathrm{ONNC}} + 1/h(\eta_t^{\mathrm{ONNC}} - \eta_{t-h-w}^{\mathrm{ONNC}})$. Then ONNC method raises the alarm under the stopping time $\tau_{\text{ONNC}} = \inf\{t : \bar{\eta}_t^{\text{ONNC}} > b\}$.

Online Neural Network Regression (ONNR) *Online Neural Network Regression (ONNR).* ONNR is also a change-point detection method using neural networks proposed by [64]. The difference from ONNC is that ONNR learns the density ratio between pre- and post-change data rather than learning the binary labels. ONNR uses two separate neural networks to train over two loss functions, $l^{1,\text{ONNR}}(\theta_{1,t})$ and $l^{2,\text{ONNR}}(\theta_{2,t})$, respectively. The finite-sample loss functions in ONNR on the training set are defined as

$$\begin{aligned}
l^{1,\text{ONNR}}(\theta_t) = & \frac{1-a}{2w}\sum_{i=t-h-w+1}^{t-h}\left(g_{\theta_t}^{\text{ONNR}}(x_i)\right)^2 + \frac{a}{2w}\sum_{i=t-w+1}^{t}\left(g_{\theta_t}^{\text{ONNR}}(x_i)\right)^2 \\
& - \frac{1}{w}\sum_{i=t-w+1}^{t} g_{\theta_t}^{\text{ONNR}}(x_i), \\
l^{2,\text{ONNR}}(\theta_t) = & \frac{1-a}{2w}\sum_{i=t-w+1}^{t}\left(g_{\theta_t}^{\text{ONNR}}(x_i)\right)^2 + \frac{a}{2w}\sum_{i=t-h-w+1}^{t-h}\left(g_{\theta_t}^{\text{ONNR}}(x_i)\right)^2 \\
& - \frac{1}{w}\sum_{i=t-h-w+1}^{t-h} g_{\theta_t}^{\text{ONNR}}(x_i),
\end{aligned}$$

where h is the lag size, w is training/testing sample size, $a \in (0,1)$ is a hyperparameter set to be 0.1 in our paper, which is the same as in [64], and $g_{\theta_t}^{\text{ONNR}}(x)$ is the neural network function for ONNR. The test statistic at time t is calculated as

$$\eta_t^{\text{ONNR}} = \frac{1}{w}\sum_{i=t-w+1}^{t}\left(g_{\hat{\theta}_{1,t}}^{\text{ONNR}}(x_i) - 1\right) + \frac{1}{w}\sum_{i=t-w+1}^{t}\left(g_{\hat{\theta}_{2,t}}^{\text{ONNR}}(x_i) - 1\right),$$

where $\hat{\theta}_{i,t}, i = 1, 2$ are trained parameters for ONNR at time t from two neural networks. Similarly with ONNC, the detection scores from ONNR are also iteratively computed through $\bar{\eta}_t^{\text{ONNR}} = \bar{\eta}_{t-w}^{\text{ONNR}} + 1/h(\eta_t^{\text{ONNR}} - \eta_{t-h-w}^{\text{ONNR}})$. ONNR method raises the alarm under the stopping time $\tau_{\text{ONNR}} = \inf\{t : \bar{\eta}_t^{\text{ONNR}} > b\}$.

Window-Limited Generalized Likelihood Ratio (WL-GLR) For WL-GLR, we assume f_0 is known and $f_1(\cdot, \theta)$ is partially known with a parametric form and an unknown parameter $\theta \in \Theta$. With a window size w, the sequential statistics of WL-GLR are given by

$$S_t^{\mathrm{WLG}} = \max_{(t-w)^+ \leq i < t} \sup_{\theta \in \Theta} \sum_{j=i}^{t} \ell(x_j, \theta),$$

where $\ell(x_t, \theta) = \log\left(f_1(x_t, \theta)/f_0(x_t)\right)$. In general, no recursion can be yielded in the GLR algorithm. Therefore, the computation cost of the GLR algorithm is huge. However, the computation is tractable when we assume the problem is Gaussian mean shift. Then, the statistics will degenerate to

$$S_t^{\mathrm{WLG}} = \max_{(t-w)^+ \leq i < t} \frac{\left[\sum_{j=i+1}^{t}(x_j - \hat{\mu}_p)\right]^\top \widehat{\Sigma}_p^{-1} \left[\sum_{j=i+1}^{t}(x_j - \hat{\mu}_p)\right]}{t-i},$$

where $\hat{\mu}_p$ and $\widehat{\Sigma}_p$ are estimated mean and covariance from reference data. We use the degenerated WL-GLR throughout the experiments. The stopping time is defined as $\tau_{\mathrm{WLG}} = \inf\{t : S_t^{\mathrm{WLG}} > b\}$.

Window-Limited CUSUM (WL-CUSUM) WL-CUSUM is proposed in [166]. When the pre-change density f_0 is fully known and the post-change density $f_1(\cdot, \theta)$ has a parametric form but an unknown parameter $\theta \in \Theta$, the log-likelihood ratio is adaptively defined as

$$\ell(x_t, \hat{\theta}_{t-1}) = \log \frac{f_1(x_t, \hat{\theta}_{t-1})}{f_0(x_t)},$$

where $\hat{\theta}_{t-1} \in \Theta$ is an estimation of θ. If a window size w is fixed, the estimation $\hat{\theta}_t$ relies on the data within the window $\{x_t, \ldots, x_{t-w+1}\}$. The option on the estimator is not rigid. The most common option is the Maximum Likelihood Estimator (MLE), taking the form:

$$\hat{\theta}_t = \arg\max_{\theta \in \Theta} \sum_{i=0}^{w-1} \log f_1\left(x_{t-i}, \theta\right).$$

Throughout the experiments, we assume f_0 and f_1 are parameterized by Gaussian distribution and $\hat{\theta}_t$ is estimated by MLE. With the initial value $S_0^{\text{WLC}} = 0$, the recursion takes the form

$$S_t^{\text{WLC}} = \max\left\{S_{t-1}^{\text{WLC}} + \ell(x_t, \hat{\theta}_{t-1}), 0\right\}.$$

The stopping time is determined by $\tau_{\text{WLC}} = \inf\{t : S_t^{\text{WLC}} > b\}$.

Multivariate Exponentially Weighted Moving Average (MEWMA) MEWMA calculates a statistic that combines the differences between the current observation and the expected values based on historical data, weighted by exponentially decreasing weights. This weighting scheme assigns more importance to recent observations while still considering the entire history. As introduced in [167], MEWMA computes

$$z_t = rx_t + (1-r)z_{t-1}, \quad t \geq 1$$

where $z_0 = 0$ and the decay rate belongs to the range $0 < r \leq 1$. The MEWMA chart raises the alarm under the stopping time

$$\tau_{\text{M}} = \inf\{t : z_t^\top \Sigma_{z_t}^{-1} z_t > b\}$$

where the covariance matrix of pre-change z_t is specified by $\Sigma_{z_t} = \left(r\left(1-(1-r)^{2t}\right)/(2-r)\right)\Sigma_0$ with Σ_0 being the covariance matrix of pre-change x_t.

Hotelling CUSUM (H-CUSUM) Instead of using the exact log-likelihood ratio in the recursion, the H-CUSUM procedure uses the Hotelling T-square statistic in the recursion. It can be treated as a non-parametric detection statistic (only uses the first and the second order moments). It is good in detecting mean shifts but not very effective in detecting other types of changes (such as covariance shifts). Define

$$
\begin{aligned}
g_0^{\mathrm{H}}(x_t) &= \frac{1}{2}(x_t - \hat{\mu}_p)^\top \left(\widehat{\Sigma}_p + \nu I_d\right)^{-1}(x_t - \hat{\mu}_p), \\
g^{\mathrm{H}}(x_t) &= g_0^{\mathrm{H}}(x_t) - \hat{d}_p
\end{aligned}
$$

where $\hat{\mu}_p$ and $\widehat{\Sigma}_p$ are estimated mean, and covariance matrix from the reference sequence assumed to be drawn from f_0; $\hat{d}_p$ usually set to be $\mathbb{E}_{x\sim f_0} g_0^{\mathrm{H}}(x) + \epsilon$ for some small constant ϵ and is computed by sample average on a test split of the reference sequence in practice. The positive scalar ν is a regularizing parameter because the covariance matrix estimator $\widehat{\Sigma}_p$ may be singular, e.g., when the data dimension is high compared to the size of the reference sequence. Once $\hat{\mu}_p$, $\widehat{\Sigma}_p$ and $\hat{d}_p$ are pre-computed on the reference data, the recursive CUSUM is computed upon the arrival of stream samples of x_t as

$$
S_t^{\mathrm{H}} = \max\left\{S_{t-1}^{\mathrm{H}} + g^{\mathrm{H}}(x_t), 0\right\}.
$$

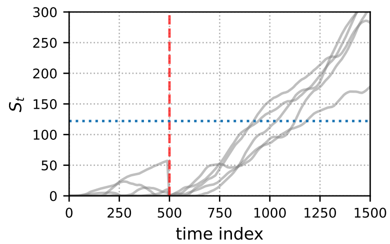
The associated stopping time procedure is $\tau_{\mathrm{H}} = \inf\{t : S_t^{\mathrm{H}} > b\}$.

B.1.2 Training Convergence

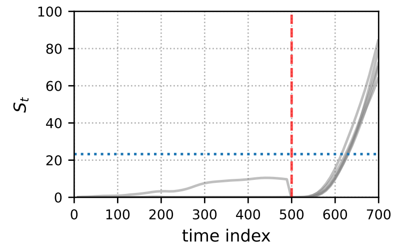
In this justification example, the setup on the pre-and post-change distributions and neural network structure inherits from Section 3.5.4. Figure 3.7 demonstrates the training convergence of the separate training on pre- and post-change sequences of the neural networks. Multiple sequences are plotted simultaneously. The figure serves as a sanity check to validate the stable convergence of the neural network in the proposed scheme.

B.1.3 Time Complexity Analysis

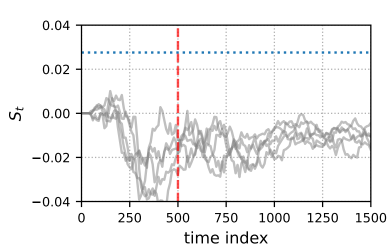
Table 3.3 presents the time complexity of NN-CUSUM and all other baselines. Regarding the data dimension d, only neural network-based method are linear while the other classic methods are either quadratic or cubic. This validates our motivation to handle high-



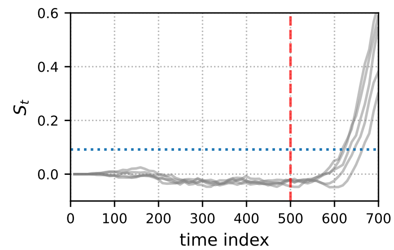
(a) Five randomly selected trajectories of detection statistics for NN-CUSUM on Higgs boson dataset.



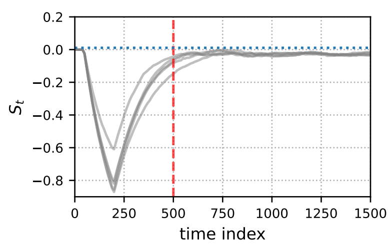
(b) Five randomly selected trajectories of detection statistics for NN-CUSUM on MiniBooNE dataset.



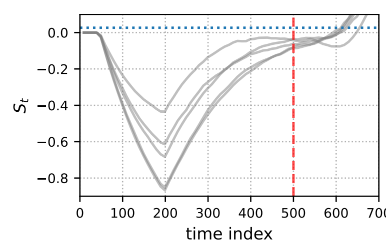
(c) Five randomly selected trajectories of detection statistics for ONNC on Higgs boson dataset



(d) Five randomly selected trajectories of detection statistics for ONNC on MiniBooNE dataset.



(e) Five randomly selected trajectories of detection statistics for ONNR on Higgs boson dataset.



(f) Five randomly selected trajectories of detection statistics for ONNR on MiniBooNE dataset.

Figure B.1: Five randomly selected trajectories of the detection statistics S_t for NN-CUSUM, ONNC, and ONNR on both datasets. For readability, the trajectories on the MiniBooNE dataset are truncated at time index 700. The red dashed line indicates the change point and the blue dotted line represents the detection threshold corresponding to 0.02 Type-I error.

dimensional data with neural networks for online change-point detection.

B.1.4 Sensitivity Analysis on the Stride of Sliding Window

Figure 3.8 shows EDD versus Type-I errors for NN-CUSUM, ONNC, and ONNR with different choices of the stride of sliding window on Higgs boson dataset and MiniBooNE dataset. To be concrete, NN-CUSUM varies its stride in $\{2, 10\}$, and ONNC and ONNR select the stride in $\{1, 2, 10\}$. NN-CUSUM has minimum stride 2 since it has half-stride structure. NN-CUSUM, ONNC, and ONNR show better performances with smaller stride of sliding window. NN-CUSUM with $s = 2$ achieve the best performance among all methods with different choices of the stride.

B.1.5 Sampled Trajectories of Detection Statistics

Figure B.1 shows five randomly selected trajectories of the detection statistics for NN-CUSUM, ONNC, and ONNR on both datasets. The red dashed line indicates the change point and the blue dotted line represents the detection threshold corresponding to 0.02 Type-I error. While all of the five randomly selected trajectories for NN-CUSUM on Higgs boson dataset exceed the detection threshold corresponding to 0.02 Type-I error, none of the trajectories for ONNC and ONNR exceed the detection threshold. On MiniBooNE dataset, all of the trajectories for NN-CUSUM, ONNC, and ONNR exceed the detection threshold. This confirms the results in Figure 3.9 and Table 3.4.

APPENDIX C

HIGHER-CRITICISM FOR SPARSE CHANGE-POINT DETECTION

C.1 Proofs

C.1.1 Technical Lemmas

We provide here several technical lemmas that will be useful in the proof of Theorem 4.5.4. Lemmas C.1.1-C.1.6 focus on univariate sequential change-point detection using the LR and GLR statistics of (4.7) and (4.9). For convenience, we assume in these lemmas that $n = 1$.

The following lemma shows that the maximum in Y_t^{GLR} and Y_t^{LR} is attained after the change with probability approaching one, provided t does not grow faster than the increase in the mean shift. This property seems well-known for Y_t^{LR} based on the discussion in [89]. The proof is somewhat simpler than in previous cases because the individual effect size μ_N goes to infinity in N. This lemma justifies the analysis of the P-values under H_1.

Lemma C.1.1. *Consider $V_{1,t,k}$ of* (4.7) *and $W_{1,t,k}$ of* (4.9)*. Suppose that $X_{1,1}, \ldots, X_{1,\tau-1} \overset{\mathsf{iid}}{\sim} \mathcal{N}(0,1)$ and $X_{1,\tau}, \ldots, X_{1,\mathcal{T}} \overset{\mathsf{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$ for some $1 \leq \tau \leq \mathcal{T}$, $\sigma^2 > 0$. Assume that $\mu \to \infty$ and $\mathcal{T} \to \infty$ while $\sqrt{\log(\mathcal{T})}/\mu \to 0$. Then,*

$$\Pr\left[\arg\max_{k \leq t} V_{1,t,k} < \tau\right] = o(1)$$

and

$$\Pr\left[\arg\max_{k \leq t} W_{1,t,k} < \tau\right] = o(1).$$

(here and throughout this section, $o(1)$ represents an expression converging to 0 uniformly in $t, \tau \leq \mathcal{T}$ as $\mu \to \infty$.)

Proof of Lemma C.1.1. Denote $X_n := X_{1,n}$ and consider the following events defined by $X_1, \dots, X_{\mathcal{T}}$.

$$\Omega_{V_t} := \{\arg\max_{k \le t} V_{1,t,k} < \tau\},$$
$$\Omega_{W_t} := \{\arg\max_{k \le t} W_{1,t,k} < \tau\},$$
$$B_t := \{\exists\, 0 < m < \tau,\, 0 \le k \le t - \tau, \text{ such that } X_{\tau-m} \ge X_{\tau+k}\}\,.$$

Note that $\Omega_{V_t} \subset B_t$ and $\Omega_{W_t} \subset B_t$. Consequently, it is enough to show that $\Pr[B_t] \to 0$.

Let $Z_1, \dots, Z_m \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$. Then

$$\Pr[\min\{Z_1, \dots, Z_m\} \le a] = 1 - \left(1 - \Phi\left(\frac{a-\mu}{\sigma}\right)\right)^m = 1 - \Phi^m\left(\frac{\mu - a}{\sigma}\right).$$
$$\Pr[\max\{Z_1, \dots, Z_m\} \le a] = \Phi^m\left(\frac{a-\mu}{\sigma}\right).$$

Define $A_a = \{\max\{X_1, \dots X_{\tau-1}\} \le a\}$. We have

$$\begin{aligned}
\Pr[B_t] &= \Pr[\max\{X_1, \dots, X_{\tau-1}\} \ge \min\{X_\tau, \dots, X_{\mathcal{T}}\}] \\
&\le \Pr[\max\{X_1, \dots, X_{\tau-1}\} \ge \min\{X_\tau, \dots, X_{\mathcal{T}}\} \mid A_a] \Pr[A_a] + \Pr[A_a^c] \\
&\le \Pr[a \ge \min\{X_\tau, \dots, X_{\mathcal{T}}\} \mid A_a] + \Pr[A_a^c] \\
&= 1 - \Phi^{\mathcal{T}-\tau+1}\left(\frac{\mu - a}{\sigma}\right) + \Pr[A_a^c]\,.
\end{aligned}$$

Let $a = \sqrt{4 \log \mathcal{T}}$, then recall that $\Phi(x) \sim 1 - \phi(x)/x$

$$\Pr[A_a] = \Phi^{\tau-1}(\sqrt{4\log\mathcal{T}}) \ge \Phi^{\mathcal{T}}(\sqrt{4\log\mathcal{T}}) \sim \left(1 - \frac{e^{-2\log\mathcal{T}}}{\sqrt{8\pi\log\mathcal{T}}}\right)^{\mathcal{T}} = 1 + o(1),$$

hence $\Pr[A_a^c] \to 0$ as $N \to \infty$. Additionally, since $\mu \to \infty$ and $\mu/\sqrt{\log(\mathcal{T})} \to \infty$, also $\mu - \sqrt{4\log(\mathcal{T})} \to \infty$ and we can use the upper bound $1 - \Phi(x) \le e^{-x^2/2}/\sqrt{2\pi}$ as $x \to \infty$.

This implies that for all N large enough,

$$\begin{aligned}\Phi^{\mathcal{T}-\tau}\left(\frac{\mu_N - \sqrt{4\log\mathcal{T}}}{\sigma}\right) &\geq \Phi^{\mathcal{T}}\left(\frac{\mu_N - \sqrt{4\log\mathcal{T}}}{\sigma}\right) \\ &\sim \left(1 - e^{-\frac{1}{2}\left(\mu_N - \sqrt{4\log(\mathcal{T})}\right)^2}\right)^{\mathcal{T}} \\ &= \left(1 - e^{-2\log(\mathcal{T})\left(\frac{\mu_N}{\sqrt{4\log(\mathcal{T})}} - 1\right)^2}\right)^{\mathcal{T}} \\ &\geq \left(1 - e^{-2\log(\mathcal{T})}\right)^{\mathcal{T}} = 1 + o(1),\end{aligned}$$

where the last transition is because $\mu/\sqrt{\log(\mathcal{T})} \to \infty$. We conclude that $\Pr\left[B_t\right] = o(1)$ which completes the proof of the lemma. $\square$

Lemma C.1.2. *Consider the setting and assumptions of Lemma C.1.1. Then*

$$\lim_{\mu\to\infty} \Pr\left[Y^{\mathsf{LR}}_{1,t} = \max_{\tau \leq k \leq t} V_{1,t,k}\right] = 1 + o(1),$$

uniformly in $\tau, t \leq \mathcal{T}$.

Proof of Lemma C.1.2. Denote $W_{t,k} := W_{1,t,k}$, $Y^{\mathsf{LR}}_t := Y^{\mathsf{LR}}_{1,t}$, $X_t := X_{1,t}$, and $S_{1,t} := S_t$.

Notice that $W_{t,m} = \mu\left[\sum_{j=m+1}^{t} X_j - \frac{\mu}{2}(t-m)\right]$ is distributed as $\mathcal{N}(\frac{\mu^2}{2}(t-m), \mu^2(t-m))$ under H_1 for $\tau \leq m \leq t$. For $\tau \leq m \leq t$,

$$\begin{aligned}\Pr\left[W_{t,m} < 0\right] &= \Pr\left[\mathcal{N}\left(\mu^2(t-m)/2, \mu^2(t-m)\right) < 0\right] \\ &= \Phi\left(-\mu\sqrt{t-m}/2\right) = e^{-\mu^2(t-m)/8}(1+o(1)). \qquad \text{(C.1)}\end{aligned}$$

In addition,

$$\max_{k\leq m} V_{m,k} - \max_{k\leq t} V_{t,k} \leq \max_{k\leq m}(V_{m,k} - V_{t,k}) = -V_{t,m}.$$

Therefore,

$$\Pr\left[\max_{k\le m} V_{m,k} - \max_{k\le t} V_{t,k} > 0\right] \le \Pr\left[-V_{t,m} > 0\right] = e^{-\mu^2(t-m)/8}(1+o(1)).$$

where the last equality is due to (C.1). Consider the event

$$B_{u,t} := \{\exists\, u \le m < t \,:\, \max_{k\le m} V_{m,k} > \max_{k\le t} V_{t,k}\}.$$

Notice that

$$\Pr\left[Y_t^{\mathsf{LR}} > \max_{\tau\le k\le t} V_{t,k}\right] \le \Pr\left[B_{1,t}\right] \le \Pr\left[B_{1,\tau}\right] + \Pr\left[B_{\tau,t}\right].$$

The first inequality is because

$$\begin{aligned}
\{Y_t^{\mathsf{LR}} > \max_{\tau\le k\le t} V_{t,k}\} &= \{\max_{1\le k\le t} V_{t,k} > \max_{\tau\le k\le t} V_{t,k}\} \\
&= \{\max_{1\le k\le \tau} V_{t,k} > \max_{\tau\le k\le t} V_{t,k}\} \\
&= \{\max_{1\le k\le \tau} V_{t,k} \ge \max_{1\le k\le t} V_{t,k}\},
\end{aligned}$$

and the last event by definition is $B_{1,t}$. We have $\Pr\left[B_{1,\tau}\right] = o(1)$ by similar arguments as in Lemma C.1.1. Additionally,

$$\begin{aligned}
\Pr\left[B_{\tau,t}\right] &\le o(1) + \sum_{m=\tau}^{t-1} \Pr\left[\max_{k\le m} V_{m,k} - \max_{k\le t} V_{t,k} > 0\right] \\
&\le (1+o(1)) \sum_{m=\tau}^{t-1} e^{-\mu^2(t-m)/8} \\
&\le (1+o(1)) \frac{e^{-\mu^2/8}}{1-e^{-\mu^2/8}} = o(1).
\end{aligned}$$

By definition, we have that $Y_t^{\mathsf{LR}} \geq \max_{\tau \leq k \leq t} V_{t,k}$. We conclude

$$\Pr\left[Y_t^{\mathsf{LR}} \neq \max_{\tau \leq k \leq t} V_{t,k}\right] = \Pr\left[Y_t^{\mathsf{LR}} > \max_{\tau \leq k \leq t} V_{t,k}\right] = o(1).$$

This completes the proof. $\square$

The following claim says that with probability going to 1, the maximum in the LR statistic is attained at the change point $k = \tau$.

Lemma C.1.3. *Consider the setting and assumption of Lemma C.1.2. Then,*

$$\Pr\left[Y_{1,t}^{\mathsf{LR}} = V_{1,t,\tau}\right] = 1 + o(1).$$

Proof of Lemma C.1.3. Denote $V_{t,k} := V_{1,t,k}$, $Y_t^{\mathsf{LR}} := Y_{1,t}^{\mathsf{LR}}$, and $S_{1,t} := S_t$. By Lemma C.1.2, we may assume

$$Y_t^{\mathsf{LR}} = \max_{\tau \leq k \leq t} V_{t,k},$$

almost surely, since the probability of the complementary event vanishes as $\mu \to \infty$. Since $Y_t^{\mathsf{LR}} \geq V_{t,\tau}$, it is sufficient to show that $\Pr\left[Y_t^{\mathsf{LR}} \geq V_{t,k} + \epsilon\right] \to 0$ for any $\epsilon > 0$. In the following, $Z_1, \ldots Z_m$ is a sequence of independent standard normal random variables. We also use the notational convention $\sum_{i=1}^{0} Z_i \equiv 0$ and assume without loss of generality that

$\mu > 0$. Fix $\epsilon > 0$.

$$
\begin{aligned}
\Pr\left[Y_t^{\mathsf{LR}} \geq V_{t,\tau} + \epsilon\right] &= \Pr\left[\max_{\tau \leq k \leq t}\left\{S_t - S_k - \frac{\mu}{2}(t-k)\right\} \geq S_t - S_\tau - \frac{\mu}{2}(t-\tau) + \frac{\epsilon}{\mu}\right] \\
&= \Pr\left[\max_{\tau \leq k \leq t}\left\{S_\tau - S_k - \frac{\mu}{2}(\tau-k)\right\} \geq \frac{\epsilon}{\mu}\right] \\
&= \Pr\left[\max_{\tau \leq k \leq t}\left\{-\left(S_k - S_\tau - \frac{\mu}{2}(k-\tau)\right)\right\} \geq \frac{\epsilon}{\mu}\right] \\
&= \Pr\left[\max_{\tau \leq k \leq t}\left\{-\left(S_k - S_\tau - \mu(k-\tau)\right) - \frac{\mu}{2}(k-\tau)\right\} \geq \frac{\epsilon}{\mu}\right] \\
&= \Pr\left[\min_{0 \leq m \leq t-\tau}\left\{\sigma\sum_{i=1}^{m} Z_i + \frac{\mu}{2}m\right\} \leq -\frac{\epsilon}{\mu}\right] \\
&= \Pr\left[\left\{\sigma Z_1 + \frac{\mu}{2} \leq -\frac{\epsilon}{\mu}\right\} \cup \left\{\sigma(Z_1+Z_2) + 2\frac{\mu}{2} \leq -\frac{\epsilon}{\mu}\right\} \cup \cdots \right. \\
&\qquad \left. \cup \left\{\sigma\sum_{i=1}^{t-\tau} Z_i + (t-\tau)\frac{\mu}{2} \leq -\frac{\epsilon}{\mu}\right\}\right] \\
&\leq \sum_{m=1}^{t-\tau}\Pr\left[\sigma\sum_{i=1}^{m} Z_i + m\frac{\mu}{2} \leq -\frac{\epsilon}{\mu}\right] = \sum_{m=1}^{t-\tau}\Pr\left[\sigma\sqrt{m}Z_1 + m\frac{\mu}{2} \leq -\frac{\epsilon}{\mu}\right] \\
&= \sum_{m=1}^{t-\tau}\Phi\left(\frac{-\frac{\epsilon}{\mu} - m\mu/2}{\sigma\sqrt{m}}\right) \leq \sum_{m=1}^{t-\tau}\Phi\left(-\frac{\sqrt{m}\mu}{2\sigma}\right) \\
&\leq \frac{1}{\sigma\sqrt{2\pi}}\sum_{m=1}^{t-\tau} e^{-m\mu^2/(8\sigma^2)} \leq \frac{1}{\sigma\sqrt{2\pi}}\frac{e^{-\mu^2/(8\sigma^2)}}{1 - e^{-\mu^2/(8\sigma^2)}}.
\end{aligned}
$$

The proof is completed by noticing that, as $\mu \to \infty$, the last expression converges to zero and is independent of t. □

Lemma C.1.4. *Consider* $W_{1,t,k}$ *of* (4.9). *Suppose that* $X_{1,\tau}, \ldots, X_{1,\mathcal{T}} \overset{\mathsf{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$ *for some* $1 \leq \tau \leq \mathcal{T}$, $\sigma^2 > 0$, *and* $\mu \to \infty$. *For* $\tau \leq v \leq t$

$$
\Pr\left[W_{1,t,\tau} \leq W_{1,t,v}\right] \leq \exp\left\{-\frac{1}{2}\frac{\mu^2(v-\tau)}{5\sigma^2}\right\}(1+o(1)). \tag{C.2}
$$

The following lemma bounds the probability of events of the form $W_{1,t,\tau} < W_{1,t,v}$ for $v < t$ after the change as $\mu \to \infty$. The bound behaves as if $J_k := W_{1,t,k}$ is a Gaussian random walk in this limit.

Proof of Lemma C.1.4. Denote $W_{t,k} := W_{1,t,k}$, $Y_t^{\mathsf{LR}} := Y_{1,t}^{\mathsf{LR}}$, $X_t := X_{1,t}$, and $S_{1,t} := S_t$. In the following, for integers $\tau < v < t$, denote $S_{t,v} := S_t - S_v$. We have

$$\begin{aligned}
\Pr\left[W_{t,\tau} < W_{t,v}\right] &= \Pr\left[\frac{S_{t,\tau}^2}{t-\tau} < \frac{S_{t,v}^2}{t-v}\right] \\
&= \Pr\left[\frac{S_{t,v}^2}{t-\tau} + \frac{S_{v,\tau}^2}{t-\tau} + 2\frac{S_{t,v}S_{v,\tau}}{t-\tau} < \frac{S_{t,v}^2}{t-v}\right] \\
&= \Pr\left[S_{t,v}^2\frac{v-\tau}{t-v} > S_{v,\tau}^2 + 2S_{t,v}S_{v,\tau}\right]
\end{aligned}$$

where the second equality uses the fact that $S_{t,\tau} = S_{t,v} + S_{v,\tau}$ and the third equality follows by grouping terms with $S_{t,v}^2$. We now solve the quadratic equation for $S_{t,v}$. This leads to

$$\begin{aligned}
\Pr\left[W_{t,\tau} < W_{t,v}\right] \leq \Pr&\left[S_{t,v}\frac{v-\tau}{t-v} > S_{v,\tau}\left(1 + \sqrt{1 + \frac{v-\tau}{t-v}}\right)\right] \\
&+ \Pr\left[S_{t,v}\frac{v-\tau}{t-v} < S_{v,\tau}\left(1 - \sqrt{1 + \frac{v-\tau}{t-v}}\right)\right].
\end{aligned}$$

The second term goes to zero as N goes to infinity whenever $v > \tau$ because the probability that one of $S_{t,v}$ and $S_{v,\tau}$ is negative goes to zero as their mean increases to infinity. We proceed by handling the first term. We denote by Z_1, Z_2, and Z_3 three independent standard normal RVs. Using that $S_{v,\tau}$ is independent of $S_{t,v}$ and both are sums of several independent

$\mathcal{N}(\mu, \sigma^2)$, we get

$$
\begin{aligned}
&\Pr\left[S_{t,v}\frac{v-\tau}{t-v} > S_{v,\tau}\left(1+\sqrt{1+\frac{v-\tau}{t-v}}\right)\right] \\
&= \Pr\left[\frac{S_{t,v}}{t-v} > \frac{S_{v,\tau}}{v-\tau}\left(1+\sqrt{1+\frac{v-\tau}{t-v}}\right)\right] \\
&= \Pr\left[\frac{\sigma Z_1}{\sqrt{t-v}} + \mu > (\frac{\sigma Z_2}{\sqrt{v-\tau}} + \mu)\left(1+\sqrt{1+\frac{v-\tau}{t-v}}\right)\right] \\
&= \Pr\left[\frac{Z_1}{\sqrt{t-v}} > \frac{Z_2}{\sqrt{v-\tau}}\left(1+\sqrt{1+\frac{v-\tau}{t-v}}\right) + \frac{\mu}{\sigma}\sqrt{1+\frac{v-\tau}{t-v}}\right] \\
&= \Pr\left[Z_1\sqrt{v-\tau} > Z_2\left(\sqrt{t-v}+\sqrt{t-\tau}\right) + \frac{\mu}{\sigma}\sqrt{t-\tau}\sqrt{v-\tau}\right] \\
&= \Pr\left[Z_3 \geq \frac{\mu}{\sigma}\sqrt{\frac{(t-\tau)(v-\tau)}{v-\tau+\left(\sqrt{t-v}+\sqrt{t-\tau}\right)^2}}\right] \\
&\leq \Pr\left[Z_3 \geq \frac{\mu}{\sigma}\sqrt{\frac{(t-\tau)(v-\tau)}{t-\tau+\left(\sqrt{t-\tau}+\sqrt{t-\tau}\right)^2}}\right] \\
&= \Pr\left[Z_3 \geq \frac{\mu}{\sigma}\sqrt{\frac{v-\tau}{5}}\right] = 1-\Phi\left(\frac{\mu\sqrt{v-\tau}}{\sqrt{5}\sigma}\right).
\end{aligned}
$$

From here, the advertised claim follows from Mill's ratio. □

Lemma C.1.5. *Consider the setting and assumptions of Lemma C.1.1. Then,*

$$\Pr\left[Y_{1,t}^{\mathsf{GLR}} = \max_{\tau\leq k\leq t} W_{1,t,k}\right] = 1 + o(1).$$

Proof of Lemma C.1.5. The proof is similar to Lemma C.1.2. Simply replace (C.1) with (C.2). □

The following claim says that with probability going to 1, the maximum in the GLR statistic is attained at the change point $k = \tau$.

Lemma C.1.6. *Consider the setting and conditions of Lemma C.1.5. Then*

$$\Pr\left[Y_{1,t}^{\mathsf{GLR}} = W_{1,t,\tau}\right] = 1 + o(1).$$

Proof of Lemma C.1.6. Denote $W_{t,k} := W_{1,t,k}$ and $Y_t^{\mathsf{LR}} := Y_{1,t}^{\mathsf{LR}}$. By Lemma C.1.5, we may assume

$$Y_t^{\mathsf{GLR}} = \max_{\tau \le k \le t} W_{t,k},$$

since the probability of the complementary event vanishes as $N \to \infty$. For $\tau < v \le t$, we use Lemma C.1.4 and the union bound to conclude

$$\begin{aligned}
&\Pr\left[\exists v > \tau \,:\, W_{t,\tau} < W_{t,v}\right] \le \Pr\left[\{W_{t,\tau} < W_{t,\tau+1}\} \cup \cdots \{W_{t,\tau} < W_{t,t-1}\}\right] \\
&\le \sum_{v=\tau+1}^{t-1} \Pr\left[W_{t,\tau} < W_{t,v}\right] \le \frac{1}{\sqrt{10\pi\sigma^2}} \sum_{v=\tau+1}^{t-1} e^{-\frac{\mu^2(v-\tau)}{5\sigma^2}} \\
&\le \frac{1}{\sqrt{10\pi\sigma^2}} \frac{e^{-\frac{\mu^2}{5\sigma^2}}}{1 - e^{-\frac{\mu^2/2}{5\sigma^2}}} \to 0,
\end{aligned}$$

uniformly in t as $\mu \to \infty$. □

C.1.2 Proof of Main Results

Proof of Corollary 4.5.2. Denote by $\{\tilde{Q}_n^{(N)}\ n = 1, \ldots, N\}$ a sequence of distributions that satisfies (4.15) with $\rho = \rho^-$. Let $\tilde{\pi}_1, \ldots, \tilde{\pi}_N$ be P-values obeying (4.14) with $Q_n^{(N)} = \tilde{Q}_n^{(N)}$ and let $\tilde{U}^{(N)} = U(\tilde{\pi}_1, \ldots, \tilde{\pi}_N)$ be some detection statistic based on these P-values.

By (4.18) and $\rho(\Delta) \le \rho^-$, for $t \in \Theta_0$ and $q > \rho^-$ we have the asymptotic inequality

$$\Pr\left[-2\log(\tilde{Q}_n^{(N)}) \ge 2q\log(N)\right] = N^{-\left(\frac{\sqrt{q}-\sqrt{\rho^-}}{\sigma}\right)^2 + o(1)} \ge \Pr\left[-2\log(Q_{n,t}^{(N)}) \ge 2q\log(N)\right].$$

Namely, every $-2\log(Q_{n,t}^{(N)})$ is asymptotically stochastically bounded by $-2\log(\tilde{Q}_n^{(N)})$. Thus, by (4.18a), every $\pi_{n,t}$ is asymptotically stochastically bounded below by $\tilde{\pi}_n$. The assumption that U is non-decreasing as its arguments decrease implies that $\tilde{U}^{(N)}$ stochastically dominates $U_t^{(N)}$ for all $t \le \mathcal{T}$ in the sense that for any $b > 0$ and N,

$$\Pr\left[\tilde{U}^{(N)} \ge b \mid H_1^{(N)}\right] \ge \Pr\left[U_t^{(N)} \ge b \mid H_1^{(N)}\right]. \tag{C.3}$$

Suppose by contradiction that there exists $a > 0$ and $\{\Delta_k\}_{k=1,2,\ldots}$ such that

$$\lim_{k\to\infty} \Pr\left[T_U \leq \Delta_k \mid H_1^{(N_k)}\right] - \Pr\left[T_U \leq \Delta_k \mid H_0^{(N_k)}\right] = a, \tag{C.4}$$

along some sub-sequence $N_1 < N_2 < \ldots < N_k$, $k = 1, 2, \ldots$, where $T_U = \inf\{t : U_t^{(N_k)} \geq b_t^{(N_k)}\}$ with $U_t^{(N_k)}$ based on $\{\pi_{n,t},\ n = 1, \ldots, N_k\}$. Under either hypothesis $h \in \{0, 1\}$,

$$\begin{aligned}\Pr\left[T_U \leq \Delta_k \mid H_h^{(N_k)}\right] &= \Pr\left[\bigcup_{t\leq\Delta_k} \{U_t^{(N_k)} \geq b_t^{(N_k)}\} \mid H_h^{(N_k)}\right] \\ &= \Pr\left[\max_{t\leq\Delta_k}\{U_t^{(N_k)} - b_t^{(N_k)}\} \geq 0 \mid H_h^{(N_k)}\right].\end{aligned}$$

Hence by (C.4) and considering the maximal elements, we may find indices $\{k_m\}_{m=1,2,\ldots}$ and $t_m \leq \Delta_{k_m}$ such that

$$\Pr\left[U_{t_m}^{(N_{k_m})} \geq b_{t_m}^{(N_{k_m})} \mid H_1^{(N_{k_m})}\right] - \Pr\left[U_{t_m}^{(N_{k_m})} \geq b_{t_m}^{(N_{k_m})} \mid H_0^{(N_{k_m})}\right] \geq a/2. \tag{C.5}$$

Strictly speaking, we construct $\{k_m\}_{m=1,2,\ldots}$ by first finding a sub-sequence $\{k_s\}_{s=1,2\ldots}$ along which, say,

$$\Pr\left[\max_{t\leq\Delta_{k_s}}\{U_t^{(N_{k_s})} - b_t^{(N_{k_s})}\} \geq 0 \mid H_1^{(N_{k_s})}\right] - \Pr\left[U_{t_s}^{(N_{k_s})} \geq b_{t_s}^{(N_{k_s})} \mid H_0^{(N_{k_s})}\right] \geq 3a/4,$$

and then find a subsequence of $\{k_s\}_{s=1,2\ldots}$ under $H_1^{(N_{k_s})}$ in a similar fashion.

By (C.3), and since the distribution of $\tilde{U}$ and any U_t are identical under the null, (C.5) implies

$$\Pr\left[\tilde{U}^{(N_{k_m})} \geq b_{t_m}^{(N_{k_m})} \mid H_1^{(N_{k_m})}\right] - \Pr\left[\tilde{U}^{(N_{k_m})} \geq b_{t_m}^{(N_{k_m})} \mid H_0^{(N_{k_m})}\right] \geq a/2.$$

Setting $b^{(N_{k_m})} = b_{t_m}^{(N_{k_m})}$, we obtain a contradiction to (4.17). □

Proof of Corollary 4.5.3. For every t such that $\rho(\Delta) \geq \rho^+$, the array of distributions

$\{Q_{n,t},\, n = 1, \ldots, N,\, t \in \Theta_1\}$ is asymptotically stochastically bounded from above by the sequence of distributions $\{\tilde{Q}_n^{(N)}\, n = 1, \ldots, N\}$ that satisfies (4.15) with $\rho = \rho^+$. Denote by $\tilde{\pi}_1, \ldots, \tilde{\pi}_N$ P-values obeying (4.14) with $Q_n^{(N)} = \tilde{Q}_n^{(N)}$ and by $\tilde{\mathrm{HC}}^*$ the higher criticism of $\tilde{\pi}_1, \ldots, \tilde{\pi}_N$. Note that HC_t^* stochastically dominates $\tilde{\mathrm{HC}}^*$ under $H_1^{(N)}$ if $\rho(\Delta) \geq \rho^+$ and stochastically identical to $\tilde{\mathrm{HC}}^*$ under $H_0^{(N)}$. By (4.16), since $\rho^+ > \rho^*(\beta, \sigma)$, there exists $\{\tilde{b}^{(N)}\}$ such that

$$\Pr\left[\tilde{\mathrm{HC}}^* \leq \tilde{b}^{(N)} \mid H_1^{(N)}\right] + \Pr\left[\tilde{\mathrm{HC}}^* > \tilde{b}^{(N)} \mid H_0^{(N)}\right] \to 0. \tag{C.6}$$

Set $b_t^{(N)} = \tilde{b}^{(N)}$ (fixed for all t) and consider the detection procedure $T_{\mathrm{HC}} = \inf\{t \,:\, \mathrm{HC}_t^* \geq \tilde{b}^{(N)}\}$. Suppose by contradiction that

$$\limsup_{N \to \infty} \sup_{t \in \Theta_1} \Pr\left[T_{\mathrm{HC}} \leq t \mid H_0^{(N)}\right] = a, \tag{C.7}$$

for some $a \geq 0$. This implies that there exists a sub-sequence $N_1 < N_2 < \ldots \leq N_k$, $k = 1, 2, \ldots$ such that

$$\sup_{t \in \Theta_1} \Pr\left[\inf_{s \leq t}\{\mathrm{HC}_s^*\} > \tilde{b}^{(N_k)} \mid H_0^{(N_k)}\right] \geq 3a/4,$$

and thus a sequence of times $t_1, t_2, \ldots$ (not necessarily in Θ_1) such that $\Pr\left[\mathrm{HC}_{t_k}^* > b^{(N_k)} \mid H_0^{(N_k)}\right] \geq a/2$. However, since each $\mathrm{HC}_{t_k}^*$ has distribution identical to that of $\tilde{\mathrm{HC}}^*$ under $H_0^{(N)}$, the previous assertion contradicts (C.6) if $a > 0$. It follows that $a = 0$ in (C.7) thus $\Pr\left[T_{\mathrm{HC}} \leq \mathcal{T} \mid H_0^{(N)}\right] \to 0$. Finally, for any t such that $\rho(\Delta) > \rho^*(\beta, \sigma)$,

$$\begin{aligned}\Pr\left[T_{\mathrm{HC}} \geq t \mid H_1^{(N)}\right] &\leq \Pr\left[\mathrm{HC}_t^* \leq b_t^{(N)} \mid H_1^{(N)}\right] = \Pr\left[\mathrm{HC}_t^* \leq \tilde{b}^{(N)} \mid H_1^{(N)}\right] \\ &\leq \Pr\left[\tilde{\mathrm{HC}}^* \leq \tilde{b}^{(N)} \mid H_1^{(N)}\right] \to 0.\end{aligned}$$

The result follows because the convergence of either term is uniform in t. $\square$

Proof of Theorem 4.5.4. Consider the case $\square = \mathsf{LR}$. Note that

$$\begin{aligned}
V_{t,\tau} &\overset{D}{=} \left(\sqrt{\Delta}\sigma Z + \frac{\Delta}{2}\mu_N(r)\right)\mu_N(r) = \frac{1}{2}\left(\sigma Z + \sqrt{\Delta}\mu_N(r)\right)^2 - \sigma^2 Z^2/2 \\
&= \frac{1}{2}\left(\sigma Z + \mu_N(r\Delta)\right)^2 - \sigma^2 Z^2/2 \\
&= \frac{1}{2}\left(\sigma Z + \mu_N(r\Delta)\right)^2 (1 + o_p(1)),
\end{aligned}$$

where in the last transition we used that $\mu_N(r) \to \infty$ as $N \to \infty$ and that

$$\frac{Z^2}{\mu_N^2(r(\Delta))} \sim N^{-r\Delta} \to 0.$$

In particular, $V_{t,\tau}$ is unbounded in N. By Lemma C.1.6,

$$Y_t^{\mathsf{LR}} = V_{t,\tau} + o_p(1),$$

From the above and using Lemma C.2.1,

$$\begin{aligned}
-2\log \pi_t^{\mathsf{LR}}(Y_t^{\mathsf{LR}}) &= -2Y_t^{\mathsf{LR}}(1 + o_p(1)) \\
&= -2(V_{t,\tau} + o_p(1))(1 + o_p(1)) = -2(V_{t,\tau})(1 + o_p(1)) \\
&= \left(\sigma Z + \mu_N(r\Delta)\right)^2 (1 + o_p(1)).
\end{aligned}$$

Next, consider the case $\square = \mathsf{GLR}$. We have

$$W_{t,\tau}^2 \overset{D}{=} \left(\sqrt{\Delta}\mu_N(r) + \sigma Z\right)^2 = \left(\mu_N(r\Delta) + \sigma Z\right)^2, \qquad Z \sim \mathcal{N}(0,1).$$

By Lemma C.1.6,

$$(Y_t^{\mathsf{GLR}})^2 = W_{t,\tau}^2 + o_p(1) \overset{D}{=} \left(\mu_N(r\Delta) + \sigma Z\right)^2 + o_p(1),$$

where $o_p(1)$ represents a RV that goes to 0 in probability as $N \to \infty$, uniformly in t. From here, Lemma C.2.3, and continuous mapping,

$$
\begin{aligned}
-2 \log \pi_t^{\mathsf{GLR}}(Y_t^{\mathsf{GLR}}) &\overset{D}{=} (Y_t^{\mathsf{GLR}})^2(1 + o_p(1)) \\
&\overset{D}{=} (\mu_N(r\Delta) + \sigma Z)^2 (1 + o_p(1)).
\end{aligned}
$$

In the last transition, we used $\mu_N(r) \to \infty$ as $n \to \infty$. $\square$

C.2 Properties of change-point detection statistics under the null

The results in this section were not used to prove our main results. They are provided here for readers interested in analytic expressions for survival functions of $Y_t^{\square}$, $\square \in \{\mathsf{LR}, \mathsf{GLR}\}$ that can serve as an alternative to obtaining the distribution of $Y_t^{\square}$ by Monte Carlo simulations involving multiple draws of an individual sequence $\{X_t\}$.

A considerable effort in the literature on change point detection focused on characterizing the "steady state" null behavior of $Y_t^{\square}$. The following two lemmas use such results to provide the asymptotic behavior of $\pi_t^{\square}$ as $t \to \infty$, from which we may obtain asymptotic P-values as an alternative to simulations. These results show that

$$
\tilde{\pi}_t^{\mathsf{LR}}(Y_t^{\mathsf{LR}}) = e^{-Y_t^{\mathsf{LR}}} \quad \text{and} \quad \tilde{\pi}_t^{\mathsf{GLR}}(Y_t^{\mathsf{GLR}}) = e^{-(Y_t^{\mathsf{GLR}})^2/2},
$$

are asymptotic P-values under (4.6a). The results on the asymptotic power of HC from [106] apply to these asymptotic P-values.

Lemma C.2.1. *Consider π_t^{LR} of* (4.8). *As $x \to \infty$ and $\log(t)/x \to 0$,*

$$
-2 \log \pi_t^{\mathsf{LR}}(x) = 2x(1 + o(1)) \tag{C.8}
$$

uniformly in t.

Lemma C.2.2 (Theorem 1 in [89]). *As $x \to \infty$, under H_0, T_x^{GLR} is asymptotically exponentially distributed with expectation*

$$\mathbb{E}\left[T_x^{\mathsf{GLR}}\right] \sim \frac{(2\pi)^{1/2}\exp(x^2/2)}{x\int_0^\infty z\nu^2(z)dz}, \tag{C.9}$$

where the special function

$$\nu(x) = \frac{2}{x^2}\exp\left\{-2\sum_{n=1}^{\infty}\frac{1}{n}\Phi\left(-\frac{x\sqrt{n}}{2}\right)\right\},$$

with Φ the standard normal cumulative distribution function.

Lemma C.2.3. *Suppose that $x \to \infty$ and $t \to \infty$ such that $t \leq cx^2$ for some $c > 0$. Then under H_0,*

$$-2\log\pi_t^{\mathsf{GLR}}(x) = x^2(1+o(1)), \tag{C.10}$$

where $o(1) \to 0$.

Proof of Lemma C.2.1. A standard result in change-point detection says that as $x \to \infty$ and $t \to \infty$ with $\log(t)/x \to 0$,

$$e^x \Pr\left[Y_t^{\mathsf{LR}} \geq x\right]/t \to C$$

for some constant C. (c.f. [168, Ch 4.3]). Consequently,

$$\begin{aligned}
-2\log\pi_t^{\mathsf{LR}}(x) &= -2\log\Pr\left[Y_t^{\mathsf{LR}} \geq x\right] \\
&= 2x - 2\log(Ct) + o(1) \\
&= 2x(1 - \log(C)/x - \log(t)/x) + o(1) \\
&= 2x(1+o(1))
\end{aligned}$$

□

Proof of Lemma C.2.3. It follows from Lemma C.2.2 that under H_0 and $t/x^2 = O(1)$, T_x is asymptotically exponentially distributed with mean

$$\mu_x := \mathbb{E}\left[T_x | H_0\right] = \sqrt{2\pi}\frac{e^{x^2/2}}{xC},$$

where C is some numerical constant. Therefore, for $x > 0$ we may write

$$\begin{aligned}\Pr\left[Y_t^{\mathsf{GLR}} \geq x\right] &= \Pr\left[T_x \leq t\right] \\ &= (1 - e^{-t/\mu_x})(1 + o(1)) = t/\mu_x(1 + o(1)).\end{aligned}$$

It follows that

$$\begin{aligned}-2\log\Pr\left[Y_t^{\mathsf{GLR}} \geq x\right] &= 2\log(\mu_x) - 2\log(t) + o(1) \\ &= x^2 + 2\log\frac{\sqrt{2\pi}}{xC} - 2\log(t) + o(1) \\ &= x^2(1 + o(1)).\end{aligned}$$

□

APPENDIX D

POINT PROCESS WITH TIME UNCERTAINTY

D.1 Proofs in Section 5.3

Proof of Lemma 5.3.1. By Assumption (A1), we know that $\Delta_k^r > \Delta_k^l \geq \Delta_{k-1}^r$, and then we have

$$\begin{aligned}
\Pr[\Delta_k|\tilde{\mathcal{F}}_{k-1}] &= S_k(\Delta_k^l) - S_k(\Delta_k^r) \\
&= \exp\left\{-\int_{\Delta_{k-1}^r}^{\Delta_k^l} h_k(s)ds\right\} - \exp\left\{-\int_{\Delta_{k-1}^r}^{\Delta_k^r} h_k(s)ds\right\} \\
&= \exp\left\{-\int_{\Delta_{k-1}^r}^{\Delta_k^r} h_k(s)ds\right\}\left(\exp\left\{\int_{\Delta_k^l}^{\Delta_k^r} h_k(s)ds\right\} - 1\right). \qquad \text{(D.1)}
\end{aligned}$$

This leads to the expression of the likelihood $\mathcal{L}$ as follows, denoting $\Delta_0^r = 0$,

$$\begin{aligned}
\mathcal{L} &= \Pr[t_1 \in \Delta_1, \ldots, t_n \in \Delta_n, t_{n+1} > T] \qquad \text{(D.2)} \\
&= \Pr[\Delta_1]\Pr[\Delta_2|\tilde{\mathcal{F}}_1]\cdots\Pr[\Delta_n|\tilde{\mathcal{F}}_{n-1}]S_{n+1}(T) \\
&= \left(\prod_{k=1}^n \exp\{-\int_{\Delta_{k-1}^r}^{\Delta_k^r} h_k(s)ds\}\left(\exp\{\int_{\Delta_k^l}^{\Delta_k^r} h_k(s)ds\} - 1\right)\right)\exp\{-\int_{\Delta_n^r}^T h_{n+1}(s)ds\}. \qquad \text{(D.3)}
\end{aligned}$$

Recall the definition of $\lambda(t)$ as in (5.16), then (D.3) can be written by integrals of $\lambda(t)$, which leads to the expression of $\mathcal{L}$ as

$$\begin{aligned}
\mathcal{L} &= \left(\prod_{k=1}^n \exp\{-\int_{\Delta_{k-1}^r}^{\Delta_k^r} \lambda(u)du\}\left(\exp\{\int_{\Delta_k^l}^{\Delta_k^r} \lambda(u)du\} - 1\right)\right)\exp\{-\int_{\Delta_n^r}^T \lambda(u)du\} \\
&= \left(\prod_{k=1}^n \left(\exp\{\int_{\Delta_k^l}^{\Delta_k^r} \lambda(u)du\} - 1\right)\right)\exp\{-\int_0^T \lambda(u)du\},
\end{aligned}$$

Taking log proves (5.17). $\square$

Proof of Lemma 5.3.2. By (5.18) and under Assumption (A2), we have

$$\lambda(t) = \mu + \sum_{k, t_k^r < j} \frac{1}{h(t_k^r - t_k^l + 1)} \sum_{i=t_k^l}^{t_k^r} \int_{I_i} k(t', t) dt', \quad \forall t \in I_j.$$

By the definition of Λ_j in (5.21),

$$\Lambda_j = \mu + \sum_{k,\, t_k^r < j} \frac{1}{(t_k^r - t_k^l + 1)} \sum_{i=t_k^l}^{t_k^r} \frac{1}{h^2} \int_{I_i} \int_{I_j} k(t', t) dt' dt.$$

The expression (5.24) then follows by the definition of $K_{i,j}$ in (5.23). $\square$

Proof of Lemma 5.3.3. Under (A3), the definition of Λ_t in (5.26) shows that $\Lambda_t \in \sigma\{y_i, i \leq t-1\}$. The expression of log-likelihood as in (5.26) gives that

$$\Pr[y_1, \cdots, y_N] = e^{\ell} = \prod_{j=1}^{N} \left((1 - y_j) e^{-h\Lambda_j} + y_j (1 - e^{-h\Lambda_j}) \right).$$

This gives that, for any $t \in [N]$,

$$\Pr[y_1, \cdots, y_t] = \prod_{j=1,\, y_j=1}^{t} (1 - e^{-h\Lambda_j}) \prod_{j=1,\, y_j=0}^{t} e^{-h\Lambda_j}.$$

Thus, by definition, $\Pr[y_t = 1 | y_i, i \leq t-1] = 1 - e^{-h\Lambda_t}$. $\square$

D.2 Proofs in Section 5.5

Proof of Lemma 5.5.1. Using the expression (5.51), we have that, denoting $\Delta_0^r = 0$,

$$
\begin{aligned}
\mathcal{L} &= \Pr[(\Delta_1, u_1)] \Pr[(\Delta_2, u_2)|\tilde{\mathcal{F}}_1] \cdots \Pr[(\Delta_n, u_n)|\tilde{\mathcal{F}}_{n-1}] S_{n+1}(T) \\
&= \prod_{k=1}^{n} \left(e^{-\int_{\Delta_{k-1}^r}^{\Delta_k^r} \bar{\lambda}(s)ds} \int_{\Delta_k^l}^{\Delta_k^r} \lambda(t,u) e^{\int_t^{\Delta_k^r} \bar{\lambda}(s)ds} dt \right) e^{-\int_{\Delta_n^r}^{T} \bar{\lambda}(s)ds} \\
&= e^{-\int_0^T \bar{\lambda}(s)ds} \prod_{k=1}^{n} \left(\int_{\Delta_k^l}^{\Delta_k^r} \lambda(t,u_k) e^{\int_t^{\Delta_k^r} \bar{\lambda}(s)ds} dt \right).
\end{aligned}
$$

This proves the expression (5.52). □

Proof of Lemma 5.5.2. Under (A3), $\Delta_i = I_{j(i)}$ for some $j(i) \in [N]$, and then for $t \in I_j$, by (5.54),

$$
\lambda[k](t,u) = \mu(u) + \sum_{i, j(i)<j} \frac{1}{h} \int_{I_{j(i)}} k(t', t; u_i, u) dt',
$$

because $\Delta_i^r < t$ if and only if $j(i) < j$. By Assumption 5.5, for $t' \in I_{j(i)}$ and $t \in I_j$, $k(t', t; u_i, u) \equiv K_{j(i),j}(u_i, u)$. Thus,

$$
\lambda[k](t,u) = \mu(u) + \sum_{i, j(i)<j} K_{j(i),j}(u_i, u), \quad \forall t \in I_j.
$$

The r.h.s. is a constant independent of t, and this proves the lemma. □

Proof of Lemma 5.5.3. Under (A3), we have (5.55) hold, namely

$$
\ell = \sum_{k=1}^{n} \log \left(\int_{I_{j(k)}^l}^{I_{j(k)}^r} \lambda(t,u_k) e^{\int_t^{I_{j(k)}^r} \bar{\lambda}(s)ds} dt \right) - \int_0^T \bar{\lambda}(s)ds.
$$

Applying Lemma 5.5.2, we have

$$
\int_{I_{j(k)}^l}^{I_{j(k)}^r} \lambda(t,u_k) e^{\int_t^{I_{j(k)}^r} \bar{\lambda}(s)ds} dt = \Lambda_{j(k)}(u_k) \int_0^h e^{\bar{\Lambda}_{j(k)}(h-t)} dt = \frac{\Lambda_{j(k)}(u_k)}{\bar{\Lambda}_{j(k)}} (e^{h\bar{\Lambda}_{j(k)}} - 1),
$$

and meanwhile,
$$\int_0^T \bar{\lambda}(s)ds = \sum_{j=1}^{N} h\bar{\Lambda}_j.$$
This proves the expression of ℓ as (5.58). $\square$

Proof of Lemma 5.5.4. The expression (5.63) implies that $\mathbf{\Lambda}_t$ is in $\sigma\{\mathbf{y}_i, i \leq t-1\}$. From the expression of the log-likelihood (5.64), one can verify that
$$\Pr[\bar{y}_t = 1|\mathbf{y}_i, i \leq t-1] = 1 - e^{-h\bar{\Lambda}_t},$$
and for each $u \in \mathcal{V}$,
$$\Pr[y_t(u) = 1|\mathbf{y}_i, i \leq t-1, \bar{y}_t = 1] = \frac{\Lambda_t(u)}{\bar{\Lambda}_t}.$$
As a result,
$$\Pr[y_t(u) = 1|\mathbf{y}_i, i \leq t-1] = (1 - e^{-h\bar{\Lambda}_t})\frac{\Lambda_t(u)}{\bar{\Lambda}_t} = (1 - e^{-h\mathbf{\Lambda}_t^T\mathbf{1}})\frac{\Lambda_t(u)}{\mathbf{\Lambda}_t^T\mathbf{1}},$$
where in the second equality we used that $\bar{\Lambda}_t = \mathbf{\Lambda}_t^T\mathbf{1}$ by definition (5.57). Writing in vector form, this gives $\mathbb{E}[\mathbf{y}_t|\mathbf{y}_i, i \leq t-1] = \Phi(\boldsymbol{x})$ with $\boldsymbol{x} = h\mathbf{\Lambda}_t \in \mathbb{R}_+^V$. $\square$

D.3 Additional experimental details

D.3.1 More details on the time-only data in Section 5.7.1

Choice of hyperparameters

In the experiment with $N = 32$ and $N' = 8$: The batch size $M_B = 400$, and the maximum number of epochs $k_{\max} = 300$. Learning rate: for `TULIK-VI`, the schedule is set as $\gamma_k = 0.4$ if $k \leq 100$ and $\gamma_k = 0.2$ if $100 < k \leq 300$; for `TULIK-GD`, the schedule is set as $\gamma_k = 0.2$ if $k \leq 100$ and $\gamma_k = 0.1$ if $100 < k \leq 300$. Regularization: We use the quadratic

Algorithm 5: Stochastic batch-based training for network case

Input : Training trajectories $\{\mathbf{y}_t^{(m)}\}_{m=1}^M$, $\boldsymbol{\mu}_0$ as initial value of $\boldsymbol{\mu}$; Parameters: intensity lower bound b, barrier weight δ_b ; batch size M_B, maximum number of epochs $k_{\max}$, learning rate schedule $\{\gamma_k\}_k$; Optional: singular value threshold τ_{SVD}, smoothness weight δ_s

Output: Learned kernel θ_K and baseline $\boldsymbol{\mu}$

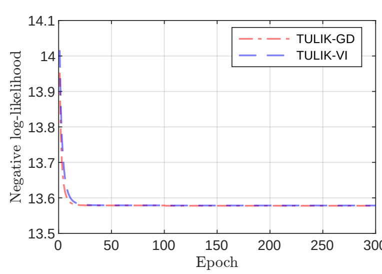
Initialize kernel $\theta_K \leftarrow 0$ and baseline on graph $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu}_0$
for $k = 1, \ldots, k_{\max}$ **do**
  while loop over the batches **do**
    $\theta \leftarrow [\boldsymbol{\mu}, \theta_K]$
    Load M_B many training trajectories to form batch $Y_B = \{\mathbf{y}^{(m)}\}_m$
    Compute intensities $\boldsymbol{\Lambda}_t^{(m)}[\theta]$ in (5.69) at $t = 1, \cdots, N$ for each $\mathbf{y}^{(m)}$ in Y_B, and let $b^{(m)} \leftarrow \min_{1 \le t \le N,\, u \in \mathcal{V}} \Lambda_t^{(m)}(u)$
    Split $Y_B = Y_{B,1} \cup Y_{B,2}$, where $\mathbf{y}^{(m)}$ in $Y_{B,1}$ if $b^{(m)} < b$, and otherwise in $Y_{B,2}$
    if using VI update **then**
      $g_B \leftarrow \frac{1}{M_B} \sum_{\mathbf{y}^{(m)} \in Y_{B,2}} \hat{G}^{(m)}[\theta]$, $\hat{G}^{(m)}$ as in (5.70)
    else if using GD update **then**
      $g_B \leftarrow \frac{1}{M_B} \sum_{\mathbf{y}^{(m)} \in Y_{B,2}} \hat{F}^{(m)}[\theta]$, $\hat{F}^{(m)}$ as in (5.71)
    end
    if $Y_{B,1}$ not empty **then**
      $g_B \leftarrow g_B + \delta_b \sum_{\mathbf{y}^{(m)} \in Y_{B,1}} \partial_{\theta_K} B^{(m)}(\theta)$, $B^{(m)}$ as in (5.76)
    end
    Update $\theta_K \leftarrow \theta_K - \gamma_k g_B$
    For each $u \in \mathcal{V}$, solve $\tilde{\mu}(u)$ as root of (5.75) on Y_B using bisection search
    Update $\mu(u) \leftarrow 0.9\mu(u) + 0.1\tilde{\mu}(u)$, $\forall u \in \mathcal{V}$
  end
  (Optional) **if** do smoothness regularization of kernel **then**
    $\theta_K \leftarrow \theta_K - \gamma_k \delta_s \partial_{\theta_K} \sum_{u', u \in \mathcal{V}} S([\Psi_{i,l}(u', u)]_{i,l})$, S as in (5.74)
  end
end
(Optional) **if** do low-rank truncation of kernel **then**
  Rearrange entries in θ_K into $\{[\Psi_{i,l}(u', u)]_{i,l},\ u', u \in \mathcal{V}\}$ and concatenate matrices $\boldsymbol{\Psi} \leftarrow [\Psi(u', u)]_{u', u \in \mathcal{V}}$, $\boldsymbol{\Psi}$ has size $(N + N')V^2 \times N'$
  For each (u', u), $\Psi'(u', u) \leftarrow \Psi_{(N'+1):N, 1:N'}(u', u)$, and concatenate matrices $\boldsymbol{\Psi}' \leftarrow [\Psi'(u', u)]_{u', u \in \mathcal{V}}$, $\boldsymbol{\Psi}'$ has size $(N - N' + 1)V^2 \times N'$
  $r \leftarrow$ number of singular values of $\boldsymbol{\Psi}'$ larger than τ_{SVD}
  $V \leftarrow$ first r right singular vectors of $\boldsymbol{\Psi}'$
  $\boldsymbol{\Psi} \leftarrow \boldsymbol{\Psi} V V^\top$, and rearrange $\boldsymbol{\Psi}$ into θ_K
end
return $\boldsymbol{\mu}$, θ_K

barrier in (5.73), the intensity lower bound $b = 0.01$, the barrier weight $\delta_b = 0.1$. We apply the smoothness penalty (5.74) with smoothness weight $\delta_s = 0.08$.

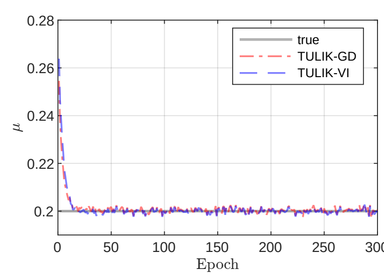
In the experiment with $N = 320$ and $N' = 80$: all the choices of optimization hyperparameters are the same as the experiment with $N = 32$ and $N' = 8$ except for $\delta_s = 0.004$. We apply low-rank truncation after the SGD loops. We select the singular value threshold $\tau_{\text{SVD}} \in \{0.2, 0.4, \ldots, 1.2\}$ by minimizing the ℓ_1 probabilistic prediction errors on a validation set with 500 trajectories. This gives $\tau_{\text{SVD}} = 0.6$ and 0.8 for `TULIK-VI` and `TULIK-GD` respectively. The truncated kernel matrices have rank 3 for both cases and are plotted in Figure 5.6.

Training dynamics for the data with $N = 32$ and $N' = 8$

Figure D.1(a) shows the descending dynamics of training negative log-likelihood functions for both `TULIK-VI` and `TULIK-GD`. Meanwhile, Figure D.1(b) shows that the proposed scheme to learn μ also numerically converges to the truth $\mu = 0.2$ in both `TULIK-VI` and `TULIK-GD`.



(a) negative log-likelihood over epochs



(b) trained μ over epochs

Figure D.1: Training dynamics on simulated time-only event data with kernel function defined in (5.77), $N' = 8$ and $N = 32$. (a) The training negative log likelihood over epochs of `TULIK-GD` and `TULIK-VI`. (b) The μ learned by `TULIK-GD` and `TULIK-VI` over epochs. The true value of μ is marked by the solid line at 0.2.

Detail of alternative baselines

Note that in the GLM model [129], one can allow `GLM-L` and `GLM-S` to have non-stationary coefficients. In GLM, the standard approach to infer parameters is to construct the likelihood function and recover the parameters as MLE [169]. In the work proposing `GLM-S` and `GLM-I`, GLMs are used as part of the model for the Bernoulli processes and they train models using VI approach, which is originally introduced in [159] to adapt to problems with weaker assumptions than the convexity of the likelihood function. For GLM-based baselines, their VI-based training formally minimizes the L^2 loss $L_{\text{GLM}}(\theta) = \frac{1}{M}\sum_{m=1}^{M} \ell_{\text{GLM}}^{(m)}(\theta)$, where

$$\ell_{\text{GLM}}^{(m)}(\theta) = \frac{1}{2}\sum_{t=1}^{N}(\phi_{\text{GLM}}(\Lambda_t^{(m)}(\theta)) - y_t^{(m)})^2.$$

The link function $\phi_{\text{GLM}}(x)$ can be selected from the linear function $x\mathbb{I}_{[0,1]}(x)$, where $\mathbb{I}_{[0,1]}(x)$ is an indicator function on $[0, 1]$, or the sigmoid function $1/(1 + e^{-x})$, corresponding to `GLM-L` or `GLM-S`.

The `HP-E` method trains on the continuous-time data, which, in this experiment, are converted from data with uncertainty by assigning $t_i = jh$ for the i-th $y_j = 1$. For the learning of `HP-E`, since we are using the exponential time-invariant kernel defined in Section 5.2.1, we solve the gradient for μ, α, and β and optimize using SGD.

D.3.2 Simulated time-only event data with a time-invariant kernel

Dataset. We set discrete time grids as $N' = 16$ and $N = 32$ and define a true time-invariant kernel vector ψ visualized as the gray curve in Figure D.2. Given ψ and $\mu = 0.2$, we generate Bernoulli process data from (5.27). The training and testing data consist of 4800 and 500 trajectories, respectively.

Table D.1: Relative Errors (REs) of learned μ, kernel matrix, and probability predictions, on simulated time-only event data with the kernel vector ψ defined in Figure D.2 with $N' = 16$ and $N = 32$. The format are the same as in Table 5.1.

RE $\times 10^{-2}$	time-only event data with discrete time grids $N' = 16$ and $N = 32$								
	μ		kernel		prediction				
	`TULIK-VI`	`TULIK-GD`	`TULIK-VI`	`TULIK-GD`	`TULIK-VI`	`TULIK-GD`	GLM-L	GLM-S	HP-E
ℓ_1	0.70 (0.58)	0.60 (0.44)	10.73 (3.49)	9.99 (2.80)	2.00 (0.64)	**1.89** (0.51)	2.28 (0.51)	5.26 (0.16)	47.19 (0.94)
ℓ_2	–	0.60 (0.44)	7.52 (2.51)	7.02 (2.06)	2.43 (0.77)	**2.28** (0.62)	2.80 (0.63)	6.36 (0.31)	52.16 (0.94)
ℓ_∞	–	0.60 (0.44)	5.86 (2.36)	5.46 (2.11)	3.58 (1.10)	**3.36** (0.90)	4.32 (1.12)	9.68 (1.07)	66.53 (0.95)

Method. We compute the proposed model following Algorithm 3 and use the vector fields for the time-invariant kernel, which have been mentioned in Section 5.4.4. The choice of training hyperparameters is as follows: We set the batch size $M_B = 400$, and the maximum number of epochs $k_{\max} = 60$. Learning rate: for `TULIK-VI`, the learning

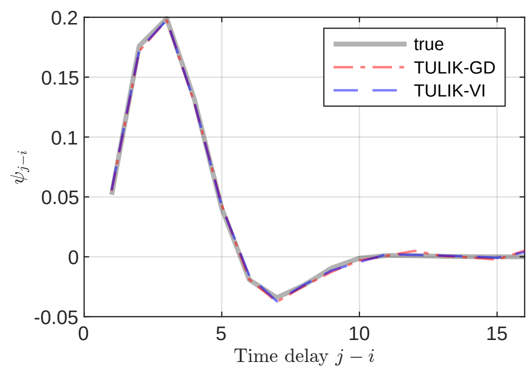


Figure D.2: True time-invariant kernel and recovered kernels on simulated time-only event data with $N' = 16$ and $N = 32$. True time-invariant kernel vector ψ is visualized as a solid curve. The kernels recovered by `TULIK-VI` and `TULIK-GD` are shown as a dashed curve and a dash-dotted curve, respectively.

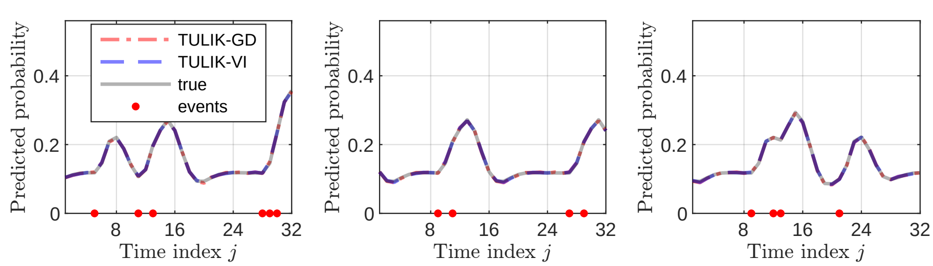


Figure D.3: The probability predictions on 3 randomly selected testing sequences. Data is the simulated time-only event data with time-invariant kernel vector ψ defined in Figure D.2 with $N' = 16$ and $N = 32$.

rate schedule is set as $\gamma_k = 0.4$ if $k \leq 20$ and $\gamma_k = 0.2$ if $20 < k \leq 60$; for `TULIK-GD`, the learning rate schedule is set as $\gamma_k = 0.2$ if $k \leq 20$ and $\gamma_k = 0.1$ if $20 < k \leq 60$. Regularization: We use the quadratic barrier in (5.73), the intensity lower bound $b = 0.01$, the barrier weight $\delta_b = 0.1$. We apply the smoothness penalty (5.74) with smoothness weight $\delta_s = 0.004$.

We compare the proposed methods with the baselines including GLM-L, GLM-S, and HP-E, which can also be computed on the stationary example.

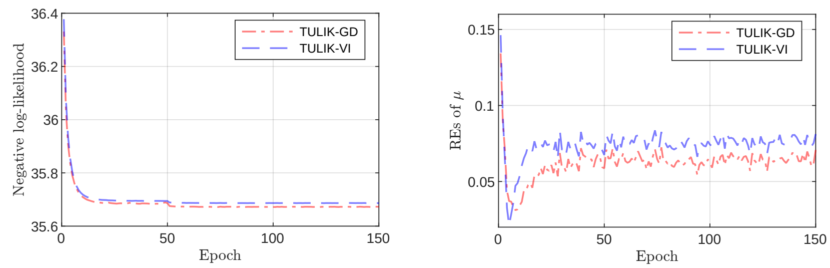
Results. In this time-invariant kernel example, the kernel ψ and the scalor μ are well recovered, as illustrated by Figure D.2 and the small REs in Table D.1. The probability predictions are also accurate, as shown in Figure D.3 and Table D.1. In comparison, TULIK methods outperform the other baselines, and in this case the advantage over `GLM-L` is mild (within one standard deviation).

D.3.3 More details on the on-network data in Section 5.7.2

Choice of hyperparameters

We use the batch size $M_B = 800$ and the maximum number of epochs $k_{\max} = 150$. Learning rate: for `TULIK-VI`, the learning rate schedule is set as $\gamma_k = 0.4$ if $k \leq 50$ and $\gamma_k = 0.2$ if $50 < k \leq 150$; for `TULIK-GD`, the learning rate schedule is set as $\gamma_k = 0.2$ if $k \leq 50$ and $\gamma_k = 0.1$ if $50 < k \leq 150$. Regularization: we use the on-network quadratic barrier (5.76) with the intensity lower bound $b = 0.03$ and the barrier weight $\delta_b = 0.1$. We adopt the smoothness weight $\delta_s = 0.004$.

After the SGD loops, we apply low-rank truncation to the kernel matrix as in Algorithm 5. We select τ_{SVD} from $\{0.2, 0.4, \ldots, 1.2\}$ by minimizing the ℓ_1 probabilistic prediction errors on a validation set with 500 trajectories, and obtain $\tau_{\text{SVD}} = 0.8$ and 1.0 for `TULIK-VI` and `TULIK-GD` respectively. The truncated matrices $\boldsymbol{\Psi}$ have rank-2 in both cases and are visualized in Figure 5.9.



(a) negative log-likelihood over epochs

(b) trained μ over epochs

Figure D.4: For simulated on-network event data in Section 5.7.2, we train the model on 40000 trajectories using both `TULIK-VI` and `TULIK-GD`. The results from `TULIK-VI` and `TULIK-GD` are visualized as dashed curves and dash-dotted curves, respectively. (a) The dynamics of training negative log-likelihood over epochs. (c) The dynamics of REs of learned μ over epochs.

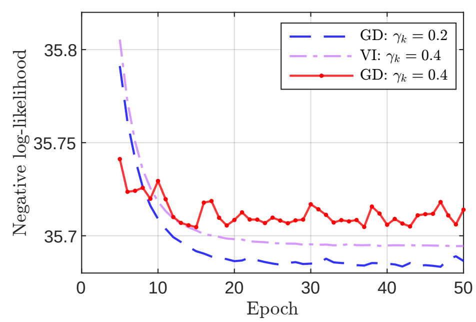


Figure D.5: Data is the on-network event data in Section 5.7.2. On the same training set, we increase the learning rate of GD to be 0.4 which is the same as VI, then the training negative log-likelihood oscillates and converges worse, showing instability in training.

Training dynamics

Figure D.4(a) shows the convergence of training log-likelihood functions for both `TULIK-VI` and `TULIK-GD`. Additionally, Figure D.4(b) validates the on-network approach to learning μ also numerically converges around 7% relative errors. Such relative errors correspond to absolute errors on each node around 0.01.

Details of alternative baselines

Note that for on-network event data, in both GLM-L and GLM-S, instead of the link functions, they use link vector field, which is the same link function as defined in time-only examples but implemented on each node, to compute the prediction probability. Their per-trajectory L^2 loss can be written as

$$\ell_{\mathrm{GLM}}^{(m)}(z) = \frac{1}{2}\sum_{t=1}^{N} \|\Phi_{\mathrm{GLM}}(\mathbf{\Lambda}_t^{(m)}[z]) - \mathbf{y}_t^{(m)}\|_2^2,$$

where $\Phi_{\mathrm{GLM}}(\mathbf{x}) = (\phi_{\mathrm{GLM}}(x(u)))_{u\in\mathcal{V}}$. For the learning of continuous-time multivariate HP-E, the kernel function becomes $k(t', t; u', u) = \alpha(u', u)\beta e^{-\beta(t-t')}$ and we perform SGD to learn $\boldsymbol{\mu}$, $(\alpha(u', u))_{u',u\in\mathcal{V}}$ and β.

Instability in GD training

Figure D.5 shows the dynamics of training negative log-likelihoods (from epoch 5 to epoch 50) for VI with a learning rate 0.4, and GD with a learning rate 0.2 and 0.4 respectively. It can be seen that, when the learning rate of GD is set to be the same as VI, the learning dynamic shows instability.

D.3.4 More on SADs data experiment in Section 5.8.1

We provide additional details for the SADs data and its results.

Data pre-processing

Table D.2 introduces the rules to group measurements into SADs and compute risk scores for SADs. The Bernoulli process $y_t(u)$ on the 13-node network is obtained from the risk scores on the first 12 nodes, and using spesis onset-or-not on the last node. Specifically, for the first 12 nodes, risk scores are assigned to indicate the severity of corresponding medical symptoms. Then for $u \in \{1, \cdots, 12\}$, namely the nodes except for sepsis, $y_t(u) = 1$ means the risk score of node u increases at hour t.

Table D.2: SAD constructions based on raw observations thresholding and measurements grouping. For some measurement, if its raw observation exceed corresponding threshold, then its risk score will be added to the total risk score of corresponding SAD.

SAD name	Measurement name	Physionet name	Threshold	Risk score
Renal Injury	creatinine	Creatinine	>1.3	0.667
	potassium	Potassium	>5.0	0.067
	phosphorus	Phosphate	>4.5	0.067
	bicarb (hco3)	HCO3	>26	0.067
	blood urea nitrogen (bun)	BUN	>20	0.133
Electrolyte Imbalance	calcium	Calcium	>10.5	0.167
	chloride	Chloride	<98 or >106	0.667
	magnesium	Magnesium	<1.6	0.167
Oxygen Carrying Dysfunction	hematocrit	Hct	<37	0.500
	hemoglobin	Hgb	<12	0.500
Shock	base excess	BaseExcess	< -3	0.100
	lactic acid	Lactate	>2.0	0.150
	ph	pH	<7.32	0.750
Diminished Cardiac Output	sbp cuff	SBP	<120	0.250
	dbp cuff	DBP	<80	0.250
	map cuff	MAP	<65	0.500
Coagulopathy	partial prothrombin time (ptt)	PTT	>35	0.250
	fibrinogen	Fibrinogen	<233	0.250
	platelets	Platelets	<150,000	0.500
Cholestasis	bilirubin direct	Bilirubin direct	>0.3	0.100
	bilirubin total	Bilirubin total	>1.0	0.500
	alkaline phosphatase	Alkalinephos	>120	0.400
Hepatocellular Injury	aspartate aminotransferase (ast)	AST	>40	1.000
Oxygenation Dysfunction (lab)	saturation of oxygen (sao2)	SaO2	<92 %	0.500
	end tidal co2	EtCO2	<35 or >45	0.250
	partial pressure of carbon dioxide (paco2)	PaCO2	<35 or >45	0.250
Inflammation (lab)	glucose	Glucose	>125	0.200
	white blood cell count	WBC	<4,000 or >12,000	0.800
Oxygenation Dysfunction (vital)	unassisted resp rate	Resp	>20	0.167
	spo2	O2Sat	<92 %	0.333
	fio2	FiO2	>21 %	0.500
Inflammation (vital)	temperature	Temp	<36 or >38	0.800
	pulse	HR	>90	0.200

Evaluation metrics

We evaluate the accuracy of sepsis onset prediction using the TPR, TNR, and BA. Since all patients in the processed data were eventually diagnosed as sepsis patients, the positive label cannot be defined simply based on sepsis onset. Instead, following the definition in [128], we consider a meaningful early prediction window $(t-12, t]$ for a sepsis onset at time t. A patient is classified as *positive* if this prediction window intersects with the

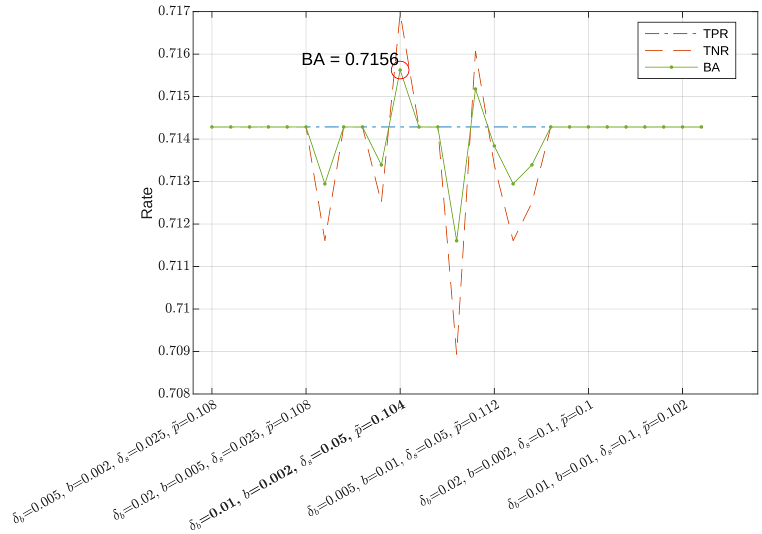


Figure D.6: After grid search in Section 5.8.1, the validation curve for SADs data learned by `TULIK-VI`. The curves with circle marker, with cross marker and with dot marker show TPR, TNR and BA over combinations of hyperparameters. The hyperparameters yielding largest BA on the validation set are bolded and the corresponding metrics on each curve are marked by red circles and texts.

time horizon $(0, 20]$, ensuring that early prediction is feasible within the given time horizon. Specifically, patients with sepsis onset at $t < 32$ are labeled as positive, while those with onset at $t \geq 32$ are labeled as negative, as meaningful early prediction is not possible within the current time horizon for very late onsets. For a positive patient, a prediction is considered a true positive if the first predicted sepsis onset occurs within the early prediction window $(t - 12, t] \cap (0, 20]$; otherwise, it is classified as a false positive. For a negative patient, a prediction is considered a true negative if no sepsis onset is predicted within $(0, 20]$; otherwise, it is classified as a false negative.

Choice of hyperparameters

We initialize the kernel tensor by filling all zero. The initialization of on-network μ can refer to Section 5.6.2. To get reasonable results, we randomly leave one validation set containing 70 many trajectories out of the training data. The remaining 1000 training trajectories will be used to update the kernel tensor and μ. We set the batch size $M_B = 107$, the maximum number of epochs $k_{\max} = 800$, and the learning rate schedule $\gamma_k = 0.4$

for all $k \leq 800$. We grid search the intensity lower bound $b \in \{0.002, 0.005, 0.01\}$, the barrier weight $\delta_b \in \{0.005, 0.01, 0.02\}$ in the quadratic loss, and the smoothness weight $\delta_s \in \{0.05, 0.1, 0.2\}$. Figure D.6 shows the validation curve for the grid search of optimal hyperparameters. We pick the hyperparameters that have the largest BA across different combinations. During the training, violations occurs more frequently than the simulations but most of them will be corrected by the barrier penalty. For each set of hyperparameters, we compute the probability predictions on the validation set after training. Denote $\hat{p}_t(u)$ as the prediction of $\mathbb{E}[y_t(u)|\mathbf{y}_i, i \leq t-1]$ in each method. We search for the probability threshold $\tilde{p}$ between $[0, 1]$ to convert $\hat{p}_t(u) > \tilde{p}$ into ones and $\hat{p}_t(u) \leq \tilde{p}$ into zeros to minimize the absolute difference between TPR and TNR on the validation set. Then we select the hyperparameters yielding largest BA on the validation set for the eventual training on the whole training data.

D.3.5 More on burglary crime data experiment in Section 5.8.2

We introduce more details for the burglary data experiments.

Data pre-processing

As shown in Figure 5.13, the Atlanta downtown area is uniformly divided into 16 sub-regions, leading to 16 nodes in the point process model. The indices of the sub-region follow a latitude-major ascending order (bottom to top, left to right in Figure 5.13). The Bernoulli process $y_t(u)$ on the 16-node network is determined by burglary-or-not within the hour t at the sub-region u.

Choice of hyperparameters

We initialize the kernel tensor as a zero tensor. The initialization of on-network μ is introduced in Section 5.6.2. To obtain reasonable results, we randomly leave out a validation set with 250 trajectories from the training data. The remaining 2000 training trajectories

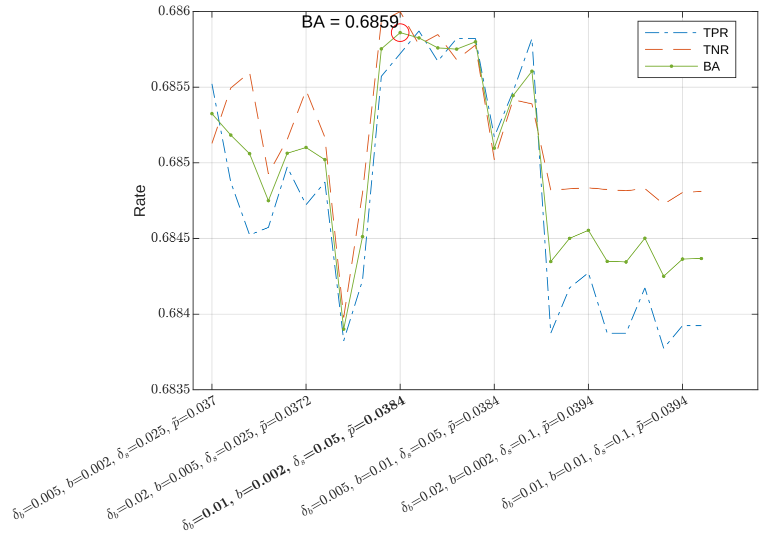


Figure D.7: Same plot as in Figure D.6. After grid search in Section 5.8.2, the validation curve for burglary crime data learned by `TULIK-VI`.

are used to learn the kernel tensor and μ. We set the batch size to $M_B = 400$, the maximum number of epochs to $k_{\max} = 800$, and the learning rate schedule to $\gamma_k = 0.2$ for all $k \leq 800$. We perform a grid search over the intensity lower bound $b \in \{0.002, 0.005, 0.01\}$, the barrier weight $\delta_b \in \{0.005, 0.01, 0.02\}$ in the quadratic loss, and the smoothness weight $\delta_s \in \{0.05, 0.1, 0.2\}$. Figure D.7 presents the validation curve for the grid search of optimal hyperparameters. The hyperparameters that yield the highest BA across different combinations are selected. The search for the probability threshold to convert the probability predictions to binary observations follows the same process as described in Section D.3.4.

REFERENCES

[1] F. Caltanissetta, L. Bertoli, and B. M. Colosimo, "In-situ monitoring of image texturing via random forests and clustering with applications to additive manufacturing," *IISE Transactions*, vol. 0, no. 0, pp. 1–15, 2023.

[2] I. Gibson *et al.*, *Additive Manufacturing Technologies*. Springer, 2021, vol. 17.

[3] S. A. Yourstone and W. J. Zimmer, "Non-normality and the design of control charts for averages," *Decision Sciences*, vol. 23, no. 5, pp. 1099–1113, 1992.

[4] Y.-M. Chou, A. M. Polansky, and R. L. Mason, "Transforming non-normal data to normality in statistical process control," *Journal of Quality Technology*, vol. 30, no. 2, pp. 133–141, 1998.

[5] B. De Ketelaere, M. Hubert, and E. Schmitt, "Overview of PCA-based statistical process-monitoring methods for time-dependent, high-dimensional data," *Journal of Quality Technology*, vol. 47, no. 4, pp. 318–335, 2015.

[6] C. Zhang, N. Chen, and J. Wu, "Spatial rank-based high-dimensional monitoring through random projection," *Journal of Quality Technology*, vol. 52, no. 2, pp. 111–127, 2020.

[7] W. H. Woodall, D. J. Spitzner, D. C. Montgomery, and S. Gupta, "Using control charts to monitor process and product quality profiles," *Journal of Quality Technology*, vol. 36, no. 3, pp. 309–320, 2004.

[8] W. H. Woodall, "Current research on profile monitoring," *Production*, vol. 17, pp. 420–425, 2007.

[9] M. R. Maleki, A. Amiri, and P. Castagliola, "An overview on recent profile monitoring papers (2008–2018) based on conceptual classification scheme," *Computers & Industrial Engineering*, vol. 126, pp. 705–728, 2018.

[10] T.-C. Chang and F.-F. Gan, "Monitoring linearity of measurement gauges," *Journal of Statistical Computation and Simulation*, vol. 76, no. 10, pp. 889–911, 2006.

[11] C. Zou, F. Tsung, and Z. Wang, "Monitoring general linear profiles using multivariate exponentially weighted moving average schemes," *Technometrics*, vol. 49, no. 4, pp. 395–408, 2007.

[12] J. Zhu and D. K. Lin, "Monitoring the slopes of linear profiles," *Quality Engineering*, vol. 22, no. 1, pp. 1–12, 2009.

[13] R. Noorossana, A. Amiri, S. Vaghefi, and E. Roghanian, “Monitoring process performance using linear profiles,” in *Proceedings of the 3rd International Industrial Engineering Conference, Tehran, Iran*, 2004.

[14] R. Kazemzadeh, R. Noorossana, and A. Amiri, “Monitoring polynomial profiles in quality control applications,” *The International Journal of Advanced Manufacturing Technology*, vol. 42, no. 7, pp. 703–712, 2009.

[15] J. Williams, J. B. Birch, W. H. Woodall, and N. M. Ferry, “Statistical monitoring of heteroscedastic dose-response profiles from high-throughput screening,” *Journal of Agricultural, Biological, and Environmental Statistics*, vol. 12, no. 2, pp. 216–235, 2007.

[16] J. M. Moguerza, A. Muñoz, and S. Psarakis, “Monitoring nonlinear profiles using support vector machines,” in *Iberoamerican congress on pattern recognition*, Springer, 2007, pp. 574–583.

[17] B. M. Colosimo and M. Pacella, “A comparison study of control charts for statistical monitoring of functional data,” *International Journal of Production Research*, vol. 48, no. 6, pp. 1575–1601, 2010.

[18] J. Lee, Y. Hur, S.-H. Kim, and J. R. Wilson, “Monitoring nonlinear profiles using a wavelet-based distribution-free CUSUM chart,” *International Journal of Production Research*, vol. 50, no. 22, pp. 6574–6594, 2012.

[19] H. Wang, S.-H. Kim, X. Huo, Y. Hur, and J. R. Wilson, “Monitoring nonlinear profiles adaptively with a wavelet-based distribution-free CUSUM chart,” *International Journal of Production Research*, vol. 53, no. 15, pp. 4648–4667, 2015.

[20] F. M. Megahed, W. H. Woodall, and J. A. Camelio, “A review and perspective on control charting with image data,” *Journal of Quality Technology*, vol. 43, no. 2, pp. 83–98, 2011.

[21] H. Yan, K. Paynabar, and J. Shi, “Image-based process monitoring using low-rank tensor decomposition,” *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 1, pp. 216–227, 2014.

[22] M. Koosha, R. Noorossana, and F. Megahed, “Statistical process monitoring via image data using wavelets,” *Quality and Reliability Engineering International*, vol. 33, no. 8, pp. 2059–2073, 2017.

[23] D. Eslami, H. Izadbakhsh, O. Ahmadi, and M. Zarinbal, “Spatial-nonparametric regression: An approach for monitoring image data,” *Communications in Statistics - Theory and Methods*, vol. 0, no. 0, pp. 1–24, 2021.

[24] A. Alaeddini, A. Motasemi, and S. H. A. Faruqui, “A spatiotemporal outlier detection method based on partial least squares discriminant analysis and area delaunay triangulation for image-based process monitoring,” *IISE Transactions*, vol. 50, no. 2, pp. 74–87, 2018.

[25] D.-T. Lee and B. J. Schachter, “Two algorithms for constructing a delaunay triangulation,” *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.

[26] F. M. Megahed, L. J. Wells, J. A. Camelio, and W. H. Woodall, “A spatiotemporal method for the monitoring of image data,” *Quality and Reliability Engineering International*, vol. 28, no. 8, pp. 967–980, 2012.

[27] Z. He, L. Zuo, M. Zhang, and F. M. Megahed, “An image-based multivariate generalized likelihood ratio control chart for detecting and diagnosing multiple faults in manufactured products,” *International Journal of Production Research*, vol. 54, no. 6, pp. 1771–1784, 2016.

[28] F. Amirkhani and A. Amiri, “A novel framework for spatiotemporal monitoring and post-signal diagnosis of processes with image data,” *Quality and Reliability Engineering International*, vol. 36, no. 2, pp. 705–735, 2020.

[29] H. Yan, K. Paynabar, and J. Shi, “Anomaly detection in images with smooth background via smooth-sparse decomposition,” *Technometrics*, vol. 59, no. 1, pp. 102–114, 2017.

[30] H. Yan, K. Paynabar, and J. Shi, “Real-time monitoring of high-dimensional functional data streams via spatio-temporal smooth sparse decomposition,” *Technometrics*, vol. 60, no. 2, pp. 181–197, 2018.

[31] W. Jiang, S. W. Han, K.-L. Tsui, and W. H. Woodall, “Spatiotemporal surveillance methods in the presence of spatial correlation,” *Statistics in Medicine*, vol. 30, no. 5, pp. 569–583, 2011.

[32] M. L. Lee, D. Goldsman, S.-H. Kim, and K.-L. Tsui, “Spatiotemporal biosurveillance with spatial clusters: Control limit approximation and impact of spatial correlation,” *IIE Transactions*, vol. 46, no. 8, pp. 813–827, 2014.

[33] M. L. Lee, D. Goldsman, and S.-H. Kim, “Robust distribution-free multivariate CUSUM charts for spatiotemporal biosurveillance in the presence of spatial correlation,” *IIE Transactions on Healthcare Systems Engineering*, vol. 5, no. 2, pp. 74–88, 2015.

[34] H. Yan, M. Grasso, K. Paynabar, and B. M. Colosimo, "Real-time detection of clustered events in video-imaging data with applications to additive manufacturing," *IISE Transactions*, vol. 54, no. 5, pp. 464–480, 2022.

[35] B. M. Colosimo and M. Grasso, "Spatially weighted PCA for monitoring video image data with application to additive manufacturing," *Journal of Quality Technology*, vol. 50, no. 4, pp. 391–417, 2018.

[36] M. Khanzadeh, S. Chowdhury, M. A. Tschopp, H. R. Doude, M. Marufuzzaman, and L. Bian, "In-situ monitoring of melt pool images for porosity prediction in directed energy deposition processes," *IISE Transactions*, vol. 51, no. 5, pp. 437–455, 2019.

[37] C. Liu, Z. Kong, S. Babu, C. Joslin, and J. Ferguson, "An integrated manifold learning approach for high-dimensional data feature extractions and its applications to online process monitoring of additive manufacturing," *IISE Transactions*, vol. 53, no. 11, pp. 1215–1230, 2021.

[38] S. Guo, W. ". Guo, and L. Bain, "Hierarchical spatial-temporal modeling and monitoring of melt pool evolution in laser-based additive manufacturing," *IISE Transactions*, vol. 52, no. 9, pp. 977–997, 2020.

[39] T. Gong *et al.*, "Distribution-free image monitoring with application to battery coating process," *IISE Transactions*, vol. 0, no. 0, pp. 1–14, 2024.

[40] P. W. Glynn and D. L. Iglehart, "Large-sample theory for standardized time series: An overview," in *Proceedings of the 17th Conference on Winter Simulation*, 1985, pp. 129–134.

[41] W. Whitt, "Stochastic-process limits: An introduction to stochastic-process limits and their application to queues," *Space*, vol. 500, pp. 391–426, 2002.

[42] S.-H. Kim, C. Alexopoulos, K.-L. Tsui, and J. R. Wilson, "A distribution-free tabular CUSUM chart for autocorrelated data," *IIE Transactions*, vol. 39, no. 3, pp. 317–330, 2007.

[43] M. Bagshaw and R. A. Johnson, "The effect of serial correlation on the performance of CUSUM tests II," *Technometrics*, vol. 17, no. 1, pp. 73–80, 1975.

[44] D. Siegmund, *Sequential Analysis: Tests and Confidence Intervals*. Springer Science & Business Media, 1985.

[45] M. Gavish and D. L. Donoho, "Optimal shrinkage of singular values," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2137–2152, 2017.

[46] Z.-D. Bai and Y.-Q. Yin, "Limit of the smallest eigenvalue of a large dimensional sample covariance matrix," *The Annals of Probability*, vol. 21, no. 3, pp. 1275–1294, 1993.

[47] W. Bryc and J. W. Silverstein, "Singular values of large non-central random matrices," *Random Matrices: Theory and Applications*, vol. 9, no. 04, p. 2 050 012, 2020.

[48] P. Otto, "Parallelized monitoring of dependent spatiotemporal processes," in *Frontiers in Statistical Quality Control 13*, Springer, 2021, pp. 165–183.

[49] Y. Okhrin, W. Schmid, and I. Semeniuk, "New approaches for monitoring image data," *IEEE Transactions on Image Processing*, vol. 30, pp. 921–933, 2021.

[50] Y. Xie, J. Huang, and R. Willett, "Change-point detection for high-dimensional time series with missing data," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 12–27, 2012.

[51] A. K. Gupta and D. K. Nagar, *Matrix Variate Distributions*. Chapman and Hall/CRC, 2018.

[52] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.

[53] A. Tartakovsky, I. Nikiforov, and M. Basseville, *Sequential analysis: Hypothesis testing and changepoint detection*. CRC Press, 2014.

[54] H. V. Poor and O. Hadjiliadis, *Quickest detection*. Cambridge University Press, 2008.

[55] L. Xie, S. Zou, Y. Xie, and V. V. Veeravalli, "Sequential (quickest) change detection: Classical results and new directions," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 2, pp. 494–514, 2021.

[56] D. Siegmund, B. Yakir, *et al.*, *The statistics of gene mapping*. Springer, 2007, vol. 1.

[57] I. Barnett and J.-P. Onnela, "Change point detection in correlation networks," *Scientific reports*, vol. 6, no. 1, p. 18 893, 2016.

[58] L. Peel and A. Clauset, "Detecting change points in the large-scale structure of evolving networks," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[59] S. Li, Y. Xie, M. Farajtabar, A. Verma, and L. Song, "Detecting changes in dynamic events over networks," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 2, pp. 346–359, 2017.

[60] W.-C. Chang, C.-L. Li, Y. Yang, and B. Póczos, "Kernel change-point detection with auxiliary deep generative models," *International Conference on Learning Representations*, 2019.

[61] M. Gupta, R. Wadhvani, and A. Rasool, "Real-time change-point detection: A deep neural network-based adaptive approach for detecting changes in multivariate time series data," *Expert Systems with Applications*, vol. 209, p. 118 260, 2022.

[62] S. Deldari, D. V. Smith, H. Xue, and F. D. Salim, "Time series change point detection with self-supervised contrastive predictive coding," in *Proceedings of the Web Conference 2021*, 2021, pp. 3124–3135.

[63] G. V. Moustakides and K. Basioti, "Training neural networks for likelihood/density ratio estimation," *arXiv preprint arXiv:1911.00405*, 2019.

[64] M. Hushchyn, K. Arzymatov, and D. Derkach, "Online neural networks for change-point detection," *arXiv preprint arXiv:2010.01388*, 2020.

[65] X. Wang, R. A. Borsoi, C. Richard, and J. Chen, "Change point detection with neural online density-ratio estimator," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2023, pp. 1–5.

[66] E. Page, "Cumulative sum charts," *Technometrics*, vol. 3, no. 1, pp. 1–9, 1961.

[67] J. Li, P. Fearnhead, P. Fryzlewicz, and T. Wang, "Automatic change-point detection in time series via deep learning," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 86, no. 2, pp. 273–285, 2024.

[68] X. Cheng and A. Cloninger, "Classification logit two-sample testing by neural networks for differentiating near manifold densities," *IEEE Transactions on Information Theory*, vol. 68, no. 10, pp. 6631–6662, 2022.

[69] X. Cheng and Y. Xie, "Neural tangent kernel maximum mean discrepancy," *Advances in Neural Information Processing Systems*, vol. 34, pp. 6658–6670, 2021.

[70] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[71] Y. Xie and D. Siegmund, "Sequential multi-sensor change-point detection," *The Annals of Statistics*, vol. 41, no. 2, pp. 670–692, 2013.

[72] S. M. Oh, J. M. Rehg, T. Balch, and F. Dellaert, "Learning and inferring motion patterns using parametric segmental switching linear dynamic systems," *International Journal of Computer Vision*, vol. 77, pp. 103–124, 2008.

[73] L. M. Candanedo and V. Feldheim, "Accurate occupancy detection of an office room from light, temperature, humidity and co2 measurements using statistical learning models," *Energy and buildings*, vol. 112, pp. 28–39, 2016.

[74] N. Kawaguchi *et al.*, "Hasc challenge: Gathering large scale human activity corpus for the real-world activity understandings," in *Proceedings of the 2nd Augmented Human International Conference*, ser. AH '11, Tokyo, Japan: Association for Computing Machinery, 2011, ISBN: 9781450304269.

[75] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature Communications*, vol. 5, no. 1, p. 4308, 2014.

[76] D. Dua, C. Graff, *et al.*, "Uci machine learning repository, 2017," *URL http://archive. ics. uci. edu/ml*, vol. 7, no. 1, p. 62, 2017.

[77] A. G. Tartakovsky and V. V. Veeravalli, "Asymptotically optimal quickest change detection in distributed sensor systems," *Sequential Analysis*, vol. 27, no. 4, pp. 441–475, 2008.

[78] H. P. Chan, "Optimal sequential detection in multi-stream data," *The Annals of Statistics*, vol. 45, no. 6, pp. 2736–2763, 2017.

[79] J. Gösmann, C. Stoehr, J. Heiny, and H. Dette, "Sequential change point detection in high dimensional time series," *Electronic Journal of Statistics*, vol. 16, no. 1, pp. 3608–3671, 2022.

[80] S. Hu, J. Huang, H. Chen, and H. P. Chan, "Likelihood scores for sparse signal and change-point detection," *IEEE Transactions on Information Theory*, vol. 69, no. 6, pp. 4065–4080, 2023.

[81] D. Donoho and J. Jin, "Higher criticism for detecting sparse heterogeneous mixtures," *The Annals of Statistics*, vol. 32, no. 3, pp. 962–994, 2004.

[82] D. Donoho and J. Jin, "Feature selection by higher criticism thresholding achieves the optimal phase diagram," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 367, no. 1906, pp. 4449–4470, 2009.

[83] D. Donoho and J. Jin, "Higher criticism for large-scale inference, especially for rare and weak effects," *Statistical Science*, vol. 30, no. 1, pp. 1–25, 2015.

[84] J. Li and D. Siegmund, "Higher criticism: p-values and criticism," *The Annals of Statistics*, vol. 43, no. 3, pp. 1323–1350, 2015.

[85] D. Donoho and A. Kipnis, "Higher criticism to compare two large frequency tables, with sensitivity to possible rare and weak differences," *The Annals of Statistics*, vol. 50, no. 3, pp. 1447–1472, 2022.

[86] D. Donoho and J. Jin, "Higher criticism thresholding: Optimal feature selection when useful features are rare and weak," *Proceedings of the National Academy of Sciences*, vol. 105, no. 39, pp. 14 790–14 795, 2008.

[87] A. Kipnis, "Higher criticism for discriminating word-frequency tables and authorship attribution," *The Annals of Applied Statistics*, vol. 16, no. 2, pp. 1236–1252, 2022.

[88] G. Lorden, "Procedures for reacting to a change in distribution," *The Annals of Mathematical Statistics*, vol. 42, no. 6, pp. 1897–1908, 1971.

[89] D. Siegmund and E. Venkatraman, "Using the generalized likelihood ratio statistic for sequential detection of a change-point," *The Annals of Statistics*, pp. 255–271, 1995.

[90] A. Delaigle, P. Hall, and J. Jin, "Robustness and accuracy of methods for high dimensional data analysis based on Student's t-statistic," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 73, no. 3, pp. 283–301, Jan. 2011. eprint: https://academic.oup.com/jrsssb/article-pdf/73/3/283/49162938/jrsssb_73_3_283.pdf.

[91] P. Hall and J. Jin, "Properties of higher criticism under strong dependence," *The Annals of Statistics*, vol. 36, no. 1, pp. 381–402, 2008.

[92] P. Hall and J. Jin, "Innovated higher criticism for detecting sparse signals in correlated noise," *The Annals of Statistics*, vol. 38, no. 3, pp. 1686–1732, 2010.

[93] H. P. Chan and G. Walther, "Optimal detection of multi-sample aligned sparse signals," *The Annals of Statistics*, vol. 43, no. 5, pp. 1865–1895, 2015.

[94] S. Bhamidi, J. Jin, and A. Nobel, "Change point detection in network models: Preferential attachment and long range dependence," *The Annals of Applied Probability*, vol. 28, no. 1, pp. 35–78, 2018.

[95] E. Pilliat, A. Carpentier, and N. Verzelen, "Optimal multiple change-point detection for high-dimensional data," *Electronic Journal of Statistics*, vol. 17, no. 1, pp. 1240–1315, 2023.

[96] I. V. Stoepker, R. M. Castro, E. Arias-Castro, and E. van den Heuvel, “Anomaly detection for a large number of streams: A permutation-based higher criticism approach,” *Journal of the American Statistical Association*, vol. 119, no. 545, pp. 461–474, 2024.

[97] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blažek, and H. Kim, “Detection of intrusions in information systems by sequential change-point methods,” *Statistical Methodology*, vol. 3, no. 3, pp. 252–293, 2006.

[98] Y. Mei, “Efficient scalable schemes for monitoring a large number of data streams,” *Biometrika*, vol. 97, no. 2, pp. 419–433, 2010.

[99] K. Liu, R. Zhang, and Y. Mei, “Scalable sum-shrinkage schemes for distributed monitoring large-scale data streams,” *Statistica Sinica*, vol. 29, no. 1, pp. 1–22, 2019.

[100] Q. Xu and Y. Mei, “Active quickest detection when monitoring multi-streams with two affected streams,” in *2022 IEEE International Symposium on Information Theory (ISIT)*, IEEE, 2022, pp. 1915–1920.

[101] J. Jin and Z. T. Ke, “Rare and weak effects in large-scale inference: Methods and phase diagrams,” *Statistica Sinica*, pp. 1–34, 2016.

[102] A. G. Tartakovsky and V. V. Veeravalli, “General asymptotic bayesian theory of quickest change detection,” *Theory of Probability & Its Applications*, vol. 49, no. 3, pp. 458–497, 2005.

[103] T. L. Lai, “Information bounds and quick detection of parameter changes in stochastic systems,” *IEEE Transactions on Information Theory*, vol. 44, no. 7, pp. 2917–2929, 1998.

[104] T. Tony Cai, X. Jessie Jeng, and J. Jin, “Optimal detection of heterogeneous and heteroscedastic mixtures,” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 73, no. 5, pp. 629–662, Aug. 2011. eprint: https://academic.oup.com/jrsssb/article-pdf/73/5/629/49496490/jrsssb_73_5_629.pdf.

[105] H. V. Poor, “Quickest detection with exponential penalty for delay,” *The Annals of Statistics*, vol. 26, no. 6, pp. 2179–2205, 1998.

[106] A. Kipnis, “Unification of rare/weak detection models using moderate deviations analysis and log-chisquared p-values,” *Statistica Scinica*, 2023, To appear.

[107] M. Pollak, “Optimal detection of a change in distribution,” *The Annals of Statistics*, vol. 13, no. 1, pp. 206–227, 1985.

[108] H. Liu, C. Gao, and R. J. Samworth, "Minimax rates in sparse, high-dimensional change point detection," *The Annals of Statistics*, vol. 49, no. 2, 2021.

[109] I. V. Stoepker, R. M. Castro, and E. Arias-Castro, "Sparse dnomaly detection across referentials: A rank-based higher criticism approach," *The Annals of Statistics*, 2025, To appear.

[110] Y. Chen, T. Wang, and R. J. Samworth, "High-dimensional, multiscale online change-point detection," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 84, no. 1, pp. 234–266, 2022.

[111] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *The Annals of Statistics*, vol. 29, no. 4, pp. 1165–1188, 2001.

[112] G. R. Shorack and J. A. Wellner, *Empirical processes with applications to statistics*. SIAM, 2009.

[113] E. Arias-Castro and M. Wang, "The sparse Poisson means model," *Electronic Journal of Statistics*, vol. 9, no. 2, pp. 2170–2201, 2015.

[114] E. Arias-Castro and A. Ying, "Detection of sparse mixtures: Higher criticism and scan statistic," *Electronic Journal of Statistics*, vol. 13, no. 1, pp. 208–230, 2019.

[115] E. Arias-Castro, R. Huang, and N. Verzelen, "Detection of sparse positive dependence," *Electronic Journal of Statistics*, vol. 14, no. 1, pp. 702–730, 2020.

[116] A. G. Hawkes, "Spectra of some self-exciting and mutually exciting point processes," *Biometrika*, vol. 58, no. 1, pp. 83–90, 1971.

[117] A. Reinhart, "A review of self-exciting spatio-temporal point processes and their applications," *Statistical Science*, vol. 33, no. 3, pp. 299–318, 2018.

[118] Y. Ogata, "Statistical models for earthquake occurrences and residual analysis for point processes," *Journal of the American Statistical Association*, vol. 83, no. 401, pp. 9–27, 1988.

[119] Y. Ogata, "Seismicity analysis through point-process modeling: A review," *Pure & Applied Geophysics*, vol. 155, no. 2-4, p. 471, 1999.

[120] J. Zhuang, "Next-day earthquake forecasts for the Japan region generated by the ETAS model," *Earth, Planets and Space*, vol. 63, no. 3, pp. 207–216, 2011.

[121] M. Farajtabar, N. Du, M. Gomez Rodriguez, I. Valera, H. Zha, and L. Song, "Shaping social activity by incentivizing users," in *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[122] E. Bacry, S. Delattre, M. Hoffmann, and J.-F. Muzy, "Some limit theorems for Hawkes processes and application to financial statistics," *Stochastic Processes and their Applications*, vol. 123, no. 7, pp. 2475–2499, 2013.

[123] P. Reynaud-Bouret and S. Schbath, "Adaptive estimation for Hawkes processes; application to genome analysis," *The Annals of Statistics*, vol. 38, no. 5, pp. 2781–2822, 2010.

[124] Y. Ogata, "Estimating the hazard of rupture using uncertain occurrence times of paleoearthquakes," *Journal of Geophysical Research: Solid Earth*, vol. 104, no. B8, pp. 17 995–18 014, 1999.

[125] P. J. Schneider and T. A. Weber, "Estimation of self-exciting point processes from time-censored data," *Physical Review E*, vol. 108, no. 1, p. 015 303, 2023.

[126] M.-A. Rizoiu *et al.*, "Interval-censored hawkes processes," *Journal of Machine Learning Research*, vol. 23, no. 338, pp. 1–84, 2022.

[127] Z. Dong, X. Cheng, and Y. Xie, "Spatio-temporal point processes with deep non-stationary kernels," in *Proceedings of the Eleventh International Conference on Learning Representations*, 2022.

[128] M. A. Reyna *et al.*, "Early prediction of sepsis from clinical data: The PhysioNet/Computing in Cardiology Challenge 2019," *Critical Care Medicine*, vol. 48, no. 2, pp. 210–217, 2020.

[129] A. Juditsky, A. Nemirovski, L. Xie, and Y. Xie, "Convex parameter recovery for interacting marked processes," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 3, pp. 799–813, 2020.

[130] S. Wei and Y. Xie, "Causal structural learning from time series: A convex optimization approach," in *2023 57th Asilomar Conference on Signals, Systems, and Computers*, IEEE, 2023.

[131] A. G. Hawkes, "Point spectra of some mutually exciting point processes," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 33, no. 3, pp. 438–443, 1971.

[132] A. G. Hawkes and D. Oakes, "A cluster process representation of a self-exciting process," *Journal of Applied Probability*, vol. 11, no. 3, pp. 493–503, 1974.

[133] D. Marsan and O. Lengline, “Extending earthquakes’ reach through cascading,” *Science*, vol. 319, no. 5866, pp. 1076–1079, 2008.

[134] K. Zhou, H. Zha, and L. Song, “Learning social infectivity in sparse low-rank networks using multi-dimensional Hawkes processes,” in *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, vol. 31, 2013, pp. 641–649.

[135] J. Yan *et al.*, “On machine learning towards predictive sales pipeline analytics,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, Feb. 2015.

[136] E. C. Hall and R. M. Willett, “Tracking dynamic point processes on networks,” *IEEE Transactions on Information Theory*, vol. 62, no. 7, pp. 4327–4346, 2016.

[137] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, “Seismic: A self-exciting point process model for predicting tweet popularity,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2015, pp. 1513–1522.

[138] C. Zhang, “Modeling high frequency data using Hawkes processes with power-law kernels,” *Procedia Computer Science*, vol. 80, pp. 762–771, 2016.

[139] N. R. Hansen, P. Reynaud-Bouret, and V. Rivoirard, “Lasso and probabilistic inequalities for multivariate point processes,” *Bernoulli*, vol. 21, no. 1, pp. 83–143, 2015.

[140] M. Okawa, T. Iwata, Y. Tanaka, H. Toda, T. Kurashima, and H. Kashima, “Dynamic Hawkes processes for discovering time-evolving communities’ states behind diffusion processes,” in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1276–1286.

[141] H. Mei and J. M. Eisner, “The neural Hawkes process: A neurally self-modulating multivariate point process,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[142] J. Zhuang, Y. Ogata, and D. Vere-Jones, “Stochastic declustering of space-time earthquake occurrences,” *Journal of the American Statistical Association*, vol. 97, no. 458, pp. 369–380, 2002.

[143] E. Lewis and G. Mohler, “A nonparametric EM algorithm for multiscale Hawkes processes,” *Journal of Nonparametric Statistics*, vol. 1, no. 1, pp. 1–20, 2011.

[144] K. Zhou, H. Zha, and L. Song, "Learning triggering kernels for multi-dimensional Hawkes processes," in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, 2013, pp. 1301–1309.

[145] E. Bacry, K. Dayri, and J.-F. Muzy, "Non-parametric kernel estimation for symmetric Hawkes processes. Application to high frequency financial data," *The European Physical Journal B*, vol. 85, no. 5, pp. 1–12, 2012.

[146] H. Wang, L. Xie, A. Cuozzo, S. Mak, and Y. Xie, "Uncertainty quantification for inferring Hawkes networks," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 7125–7134.

[147] M. Dubey, R. Palakkadavath, and P. Srijith, "Bayesian neural Hawkes process for event uncertainty prediction," *International Journal of Data Science and Analytics*, pp. 1–15, 2023.

[148] M. Osama, D. Zachariah, and P. Stoica, "Prediction of spatial point processes: Regularized method with out-of-sample guarantees," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[149] J. Yang *et al.*, "Predicting wildfire occurrence distribution with spatial point process models and its uncertainty assessment: A case study in the Lake Tahoe Basin, USA," *International Journal of Wildland Fire*, vol. 24, no. 3, pp. 380–390, 2015.

[150] X. Yang and B. Liu, "Uncertain time series analysis with imprecise observations," *Fuzzy Optimization and Decision Making*, vol. 18, no. 3, pp. 263–278, 2019.

[151] A. L. S. Maia, F. d. A. de Carvalho, and T. B. Ludermir, "Forecasting models for interval-valued time series," *Neurocomputing*, vol. 71, no. 16-18, pp. 3344–3352, 2008.

[152] Q. Song and B. S. Chissom, "Fuzzy time series and its models," *Fuzzy Sets and Systems*, vol. 54, no. 3, pp. 269–277, 1993.

[153] Q. Song and B. S. Chissom, "Forecasting enrollments with fuzzy time series — Part I," *Fuzzy Sets and Systems*, vol. 54, no. 1, pp. 1–9, 1993.

[154] Q. Song and B. S. Chissom, "Forecasting enrollments with fuzzy time series — Part II," *Fuzzy Sets and Systems*, vol. 62, no. 1, pp. 1–8, 1994.

[155] Q. Song, "A note on fuzzy time series model selection with sample autocorrelation functions," *Cybernetics & Systems*, vol. 34, no. 2, pp. 93–107, 2003.

[156] Q. Cai, D. Zhang, W. Zheng, and S. C. H. Leung, "A new fuzzy time series forecasting model combined with ant colony optimization and auto-regression," *Knowledge-Based Systems*, vol. 74, no. 1, pp. 61–68, 2015.

[157] M. Dallachiesa, B. Nushi, K. Mirylenka, and T. Palpanas, "Uncertain time-series similarity: Return to the basics," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1662–1673, 2012.

[158] D. J. Daley and D. Vere-Jones, *An Introduction to the Theory of Point Processes. Volume I: Elementary Theory and Methods*. Springer, 2003.

[159] A. B. Juditsky and A. Nemirovski, "Signal recovery by stochastic optimization," *Automation and Remote Control*, vol. 80, no. 10, pp. 1878–1893, 2019.

[160] M. Singer *et al.*, "The third international consensus definitions for sepsis and septic shock (sepsis-3)," *Jama*, vol. 315, no. 8, pp. 801–810, 2016.

[161] S. Wei, Y. Xie, C. S. Josef, and R. Kamaleswaran, "Causal graph recovery for sepsis-associated derangements via interpretable hawkes networks," in *International Conference on Machine Learning*, ser. Workshop on Interpretable Machine Learning in Healthcare (IMLH), 2021.

[162] S. Wei, Y. Xie, C. S. Josef, and R. Kamaleswaran, "Granger causal chain discovery for sepsis-associated derangements via continuous-time Hawkes processes," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2023, pp. 2536–2546.

[163] S. Wei, Y. Xie, C. S. Josef, and R. Kamaleswaran, "Causal graph discovery from self and mutually exciting time series," *IEEE Journal on Selected Areas in Information Theory*, vol. 4, pp. 747–761, 2023.

[164] ABIM, *American board of internal medicine: Laboratory reference ranges*.

[165] C. Alexopoulos, N. T. Argon, D. Goldsman, G. Tokol, and J. R. Wilson, "Overlapping variance estimators for simulation," *Operations Research*, vol. 55, no. 6, pp. 1090–1103, 2007.

[166] L. Xie, G. V. Moustakides, and Y. Xie, "Window-limited CUSUM for sequential change detection," *IEEE Transactions on Information Theory*, vol. 69, no. 9, pp. 5990–6005, 2023.

[167] C. A. Lowry, W. H. Woodall, C. W. Champ, and S. E. Rigdon, "A multivariate exponentially weighted moving average control chart," *Technometrics*, vol. 34, no. 1, pp. 46–53, 1992.

[168] B. Yakir, *Extremes in random fields: A theory and its applications*. John Wiley & Sons, 2013.

[169] P. McCullagh, *Generalized linear models*. Routledge, 2019.

VITA

Tingnan Gong was born in Jiaxing, Zhejiang Province, China in April 1998. He received the B.Sc. in Statistics from University of Science and Technology of China in 2020. Afterwards, He joined the Industrial Engineering Ph.D. program in the H. Milton Stewart School of Industrial and Systems Engineering at Georgia Institute of Technology, under supervision of Prof. Yao Xie and Prof. Seong-Hee Kim. With a fulfilling period and many stimulating experiences, he has completed his Ph.D. studies and now he is ready for new adventures. These will take him to Mingshi Investment Management, where he will be a quantitative researcher.