Technometrics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/utch20

Sequential Change-Point Detection for Mutually Exciting Point Processes

Haoyun Wang, Liyan Xie, Yao Xie, Alex Cuozzo & Simon Mak

To cite this article: Haoyun Wang, Liyan Xie, Yao Xie, Alex Cuozzo & Simon Mak (2023) Sequential Change-Point Detection for Mutually Exciting Point Processes, Technometrics, 65:1, 44-56, DOI: 10.1080/00401706.2022.2054862

To link to this article: https://doi.org/10.1080/00401706.2022.2054862

TECHNOMETRICS
2023, VOL. 65, NO. 1, 44–56
https://doi.org/10.1080/00401706.2022.2054862

Taylor & Francis
Taylor & Francis Group

Sequential Change-Point Detection for Mutually Exciting Point Processes

Haoyun Wang[a], Liyan Xie[a], Yao Xie[a], Alex Cuozzo[b], and Simon Mak[b]

[a]H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA; [b]Department of Statistical Science, Duke University, Durham, NC

ABSTRACT

We present a new CUSUM procedure for sequential change-point detection in self- and mutually-exciting point processes (specifically, Hawkes networks) using discrete events data. Hawkes networks have become a popular model in statistics and machine learning, primarily due to their capability in modeling irregularly observed data where the timing between events carries a lot of information. The problem of detecting abrupt changes in Hawkes networks arises from various applications, including neuroengineering, sensor networks, and social network monitoring. Despite this, there has not been an efficient online algorithm for detecting such changes from sequential data. To this end, we propose an online recursive implementation of the CUSUM statistic for Hawkes processes, which is computationally and memory-efficient and can be decentralized for distributed computing. We first prove theoretical properties of this new CUSUM procedure, then show the improved performance of this approach over existing methods, including the Shewhart procedure based on count data, the generalized likelihood ratio statistic, and the standard score statistic. This is demonstrated via simulation studies and an application to population code change-detection in neuroengineering.

ARTICLE HISTORY
Received February 2021
Accepted March 2022

KEYWORDS
Change-point detection; CUSUM; Hawkes processes; Neuroengineering; Online monitoring

1. Introduction

Point processes are widely used for modeling discrete events data, which consists of a series of event times and additional associated information. Recently, a class of mutually-exciting nonhomogeneous point processes called Hawkes processes (Hawkes 1971) has gained much popularity in the statistics and machine learning literature. The intensity function of the Hawkes process consists of a deterministic part and a stochastic part, which captures the triggering or inhibiting effects of past events on future events. For example, each earthquake is usually followed by a sequence of aftershock activities and the occurrence rate of aftershocks can be represented in the stochastic part of the intensity function (Ogata 1988). Hawkes processes provide a flexible model for capturing spatio-temporal correlations, and have been successfully applied in a wide range of domains including seismology (Ogata 1988, 1998), criminology (Mohler et al. 2011), epidemiology (Rizoiu et al. 2018), social networks (Yang and Zha 2013), finance (Hawkes 2018), and neural activity (Reynaud-Bouret et al. 2013).

Detection of abrupt changes in the Hawkes process is a fundamental problem, which aims to detect the change as quickly as possible subject to false alarm constraints. For instance, in sensor network monitoring, we would like to detect any change as soon as possible using a stream of event data; such changes may represent a shift in system status or event anomalies. There are, however, key challenges for detecting changes in Hawkes processes; this includes the complex spatial and temporal dependence of the event data and long-term dependencies. To address such challenges, we need to develop computationally efficient online detection algorithms with performance guarantees.

A motivating application for our work is the change-point detection of biological neural networks. This is a fundamental topic in neuroengineering (Eliasmith and Anderson 2003), an emerging area at the intersection of physical and biological sciences. The goal is to detect neural states and state changes from experimental spike train data, which records the sequence of times when a neuron fires an action potential. Hawkes processes provide an appealing model for such data: its mutually exciting property naturally mimics neuron-to-neuron influence's electrochemical dynamics. The model's probabilistic nature can also capture noisy influences on the network, resulting from unobserved neurons or external stimuli. There has been much work on applying Hawkes processes for neuroscience problems, for example, for inferring functional connectivity (Lambert et al. 2018) and uncertainty quantification (Wang et al. 2020b). Change-points over a biological network often arise from sparse population code changes (Tang et al. 2018; Reynaud-Bouret and Roy 2007). Figure 1 illustrates an example of this change. Here, each dot represents a neuron in the visual cortex. Colored dots show neurons that respond to seeing a cat or a dog, and shared dots represent common features between both animals (e.g., mammal, pet). The sequential detection of this change-point sheds light on the relationship between stimulus and response

CONTACT Yao Xie yao.xie@isye.gatech.edu H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.

Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/TECH.

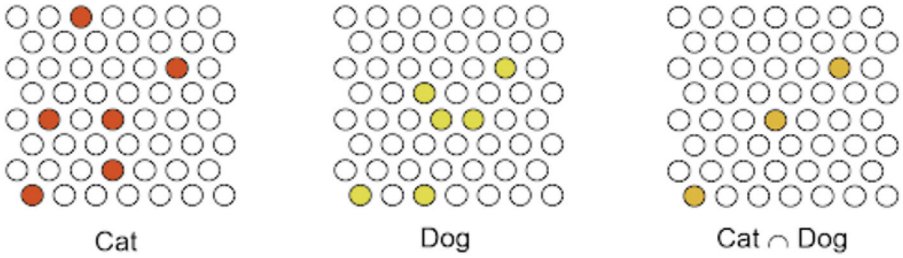

Figure 1. Visualizing sparse population coding for neuronal networks. Each dot represents a neuron, and colored dots show neurons which respond to seeing a cat or a dog.

timing, providing a better understanding of each neuron's role in the population code, which can then be used for rehabilitation of neuronal networks.

Other applications include detecting the existence of hot topics over social networks which are of interest for the social media (Li et al. 2017). Change-point detection of the distribution of crime cases can help the police department to take quick reaction and reallocate patrols (Mohler et al. 2011). Detecting changes in the spread of COVID cases will enable people to recognize a threatening new-born variant (Chiang et al. 2021).

While there has been much work on fitting Hawkes processes in the literature (see Reinhart 2018 for a recent survey), change-point detection for Hawkes processes is left an important topic with only very little attention and much less studied. In Wang et al. (2020a), the offline change-point detection problem for high-dimensional Hawkes processes was studied, and the goal is to estimate (multiple) change-points. In Rambaldi et al. (2018), a model selection scheme was proposed to identify the presence of exogenous events that increase the intensity of the Hawkes process for a given time period. A cumulant-based multi-resolution segmentation algorithm was proposed in Zhou et al. (2020) to find the optimal partition of the nonstationary Hawkes process into several nonoverlapping segments. On the contrary, we focus on the *sequential* detection problem, which aims to detect the change as quickly as possible. Online change-point detection for Hawkes processes was considered in Li et al. (2017), where the generalized likelihood ratio (GLR) test was used to detect the change with unknown post-change parameters. In that work, the expectation-maximization (EM) algorithm was used to estimate unknown post-change parameters, which does not allow for an efficient recursive implementation and could be time-consuming. For the target problem of neuronal network detection, we would like to detect the change in real-time from streaming data, using a more computationally efficient procedure.

In this article, we present a novel CUSUM procedure for sequential change detection in Hawkes processes. The recursive CUSUM is based on a log-likelihood ratio statistic, which is further modified to improve computational efficiency with the practical consideration of removing historical data with long lags. The new CUSUM procedure is computationally and memory efficient as a recursive procedure, which is crucial for its online implementation in practice, such as sensor network and social network monitoring problems. We study the theoretical properties of this new CUSUM procedure, including an analysis of its average run length (ARL) and expected detection delay (EDD). We then compare the proposed CUSUM procedure with existing change detection algorithms based on the GLR statistic (Li et al. 2017) and the score statistic in a comprehensive simulation study. Finally, we apply our method to the aforementioned motivating neuroengineering problem on population code change-detection for biological neural networks. Numerical results show that the proposed CUSUM procedure outperforms existing alternative methods.

The rest of the article is organized as follows. Section 2 introduces the basics for Hawkes processes. Section 3 sets up the change-point detection problem, outlines the proposed CUSUM procedure, discusses algorithmic developments for computational and memory efficiency, and presents its theoretical properties. Section 4 discusses some alternative detection methods. Sections 5 and 6 compares the proposed CUSUM approach with existing methods for a simulation study and a real-world application using neural spike train data. Section 7 concludes the article with some discussions.

2. Preliminaries

We first provide some background on point processes and Hawkes processes, which will be used in later sections.

A temporal point process is a random process whose realization consists of a sequence of discrete events occurring at times $\{t_i, i = 1, 2, \ldots\}$, with $t_i \in \mathbb{R}^+$. Let the history $\mathcal{H}_{t^-}$ be the sequence of times of events $\{t_1, t_2, \ldots, t_n\}$ up to but *not* including time t. Let N_t represents the number of events before time t, then N_t is a counting process which can be defined as: $dN_t = \sum_{t_i \in \mathcal{H}_t} \delta(t - t_i)dt$, where δ is the Dirac function. The sequence of discrete event times $\{t_i, i = 1, 2, \ldots\}$ can be regarded as when the counting process N_t has jumped.

A point process can be characterized by its conditional intensity function, denoted as $\lambda(t)$. This conditional intensity function is also known as the hazard function (Rasmussen 2011), and is defined as $\lambda(t) = f^*(t)/(1 - F^*(t))$. Here, $f^*(t)$ is the probability density function of the next event time conditional on the past, and $F^*(t) = \mathbb{P}\{t_{n+1} < t|\mathcal{H}_{t^-}\}$ is the associated conditional cumulative distribution function capturing the probability of the $(n+1)$th event happening before time t. Thus, if we consider a small time interval $[t, t + dt)$, we have

$$\lambda(t)dt = \frac{f^*(t)dt}{1 - F^*(t)} = \frac{\mathbb{P}(t_{n+1} \in [t, t + dt))}{\mathbb{P}(t_{n+1} \geq t)} = \mathbb{P}\{t_{n+1} \in [t, t + dt)|\mathcal{H}_{t^-}\}.$$

2.1. One-Dimensional Point Processes

For one-dimensional Hawkes process, the intensity function takes the form (Hawkes 1971):

$$\lambda(t) = \mu(t) + \alpha \int_0^t \varphi(t-\tau) dN_\tau, \tag{1}$$

where $\mu(t)$ is the base intensity, α is the influence parameter, and $\varphi(t)$ is a normalized kernel function satisfying $\int \varphi(t)dt = 1$. A commonly used kernel function is the exponential kernel $\varphi(t) = \beta e^{-\beta t}$ with $\beta > 0$. We assume $0 \le \alpha < 1$ to ensure a stationary process.

Given event times $\{t_1, t_2, \ldots, t_n\}$ which happened before a given time $t < \infty$, the log-likelihood function for the Hawkes process can be written as follows (see Daley and Vere-Jones 2003 for details):

$$\ell_t = \sum_{i=1}^n \log\left[\mu(t_i) + \alpha \sum_{t_j<t_i} \varphi(t_i - t_j)\right] - \int_0^t \mu(s)ds - \alpha \sum_{i=1}^n \int_{t_i}^t \varphi(s - t_i)ds. \tag{2}$$

In case of the exponential kernel $\varphi(t) = \beta e^{-\beta t}$ and a constant base intensity μ, (2) reads

$$\ell_t = \sum_{i=1}^n \log\left[\mu + \alpha \sum_{t_j<t_i} \beta e^{-\beta(t_i - t_j)}\right] - \mu t - \alpha \sum_{i=1}^n \left[1 - e^{-\beta(t-t_i)}\right].$$

As we will see in the following, this log-likelihood plays a key role in sequential change detection procedures.

2.2. Network Point Processes

The multivariate Hawkes process on a network with D nodes is represented by a series of event times together with their location $\{(t_i, u_i), i = 1, 2, \ldots\}$, where $t_i \in \mathbb{R}^+$ is the event time and $u_i \in [D]$ is the node on which the ith event occurs. Here we use $[D]$ to represent the set $\{1, \ldots, D\}$. The intensity function for node i at time t is

$$\lambda_i(t) = \mu_i(t) + \sum_{j\in[D]} \alpha_{ij} \int_0^t \varphi_{ij}(t-s) dN_s^j,$$

where $\mu_i(t)$ is the base intensity at node i, α_{ij} is the influence parameter from node j to node i, $\varphi_{ij}(t)$ is a normalized kernel function, and N_t^j is a counting process on node j: $dN_t^j = \sum_{k:t_k<t, u_k=j} \delta(t - t_k)dt$. The log-likelihood function for the network setting up to time t is given by:

$$\ell_t(A) = -\sum_{i\in[D]} \int_0^t \lambda_i(s)ds + \sum_{i\in[D]} \int_0^t \log(\lambda_i(s)) dN_s^i, \tag{3}$$

where $A = (\alpha_{ij})_{i,j\in[D]} \in \mathbb{R}^{D\times D}$ is the matrix representation for the influence parameters.

The log-likelihood expression in Equation (3) reveals a useful property which we later exploit for distributed change-point detection. Note that this log-likelihood can be decoupled as the summation over D nodes, in that it consists of the sum of the log-likelihood at each node. Furthermore, the intensity function $\lambda_i(\cdot)$ only involves events observed on the neighbors of i, that is, the nodes which influence node i. This property allows us to develop a *distributed* change-point detection procedure, where each node can compute their likelihood in parallel, and only needs to communicate with neighboring nodes and not over the entire network (assuming such neighborhood information is known beforehand).

3. Proposed CUSUM Detection Framework

3.1. Problem Set-up

The problem of change-point detection for Hawkes networks can be set-up as follows. Assume there exists a true change-point time $\kappa > 0$, and the event data follows one point process before the change-point and follows another point process afterward. We consider in this work two specific cases: (a) the null (pre-change) point process is a Poisson process, whereas the alternative (post-change) point process is a Hawkes point process; (b) the null point process is a Hawkes point process, whereas the alternative point process is a different Hawkes point process, for example, the influence parameter A has been shifted. Note that the first scenario can be seen as a specific case of the second, since a Poisson process can be viewed as a specific Hawkes process with influence parameters set as 0.

Consider now a hypothesis test for detecting temporal pattern shifts in the Hawkes process. Assuming the Hawkes process is stationary and the change-point κ is an *unknown* variable, this test can be formulated as:

$$\begin{aligned}
H_0: \quad & \lambda_i^*(s) = \mu_i(s) + \sum_{j\in[D]} \alpha_{ij,0} \int_0^s \varphi_{ij}(s-v) dN_v^j; \\
& \qquad\qquad i \in [D], s \ge 0, \\
H_1: \quad & \lambda_i^*(s) = \mu_i(s) + \sum_{j\in[D]} \alpha_{ij,0} \int_0^s \varphi_{ij}(s-v) dN_v^j; \\
& \qquad\qquad i \in [D], 0 \le s \le \kappa, \\
& \lambda_i^*(s) = \mu_i(s) + \sum_{j\in[D]} \alpha_{ij,1} \int_\kappa^s \varphi_{ij}(s-v) dN_v^j; \\
& \qquad\qquad i \in [D], s > \kappa.
\end{aligned} \tag{4}$$

Here, $\lambda_i^*(s)$ denotes the true intensity for node i at time s. The pre-change parameters $\{\alpha_{ij,0}\}_{i,j\in[D]}$ can typically be elicited from prior knowledge of the process or estimated from reference data. The post-change parameters $\{\alpha_{ij,1}\}_{i,j\in[D]}$ are known in some scenarios, but more often it corresponds to an unexpected anomaly and we may not have enough data to estimate this in advance. Alternatively, we can treat the post-change parameters as the targeted smallest change to be detected. A change detection procedure resolves the two hypotheses using a stopping time T, which is a function of the event sequence, as explained next.

3.2. A Recursive CUSUM Statistic

We now present the cumulative sum (CUSUM) statistics based on the log-likelihood ratio. The CUSUM procedure was first proposed in Page (1954), assuming both pre- and post-change

parameters are provided (or estimated). The CUSUM is known to be computationally efficient since it can be computed recursively. The CUSUM procedure is most commonly defined for iid observations, but there is much recent development in extending CUSUM for non iid observations (Tartakovsky et al. 2015; Tartakovsky 2020; Xie et al. 2021).

We first define the log-likelihood ratio function which will be used as the main building block of our procedure. For a hypothesized change-point τ, the log-likelihood ratio of the model (4) up to time t can be derived as:

$$\ell_{t,\tau} = \sum_{i\in[D]} \int_\tau^t \log\left(\frac{\lambda_{i,\tau}(s)}{\lambda_{i,\infty}(s)}\right) dN_s^i - \sum_{i\in[D]} \int_\tau^t (\lambda_{i,\tau}(s) - \lambda_{i,\infty}(s))ds, \tag{5}$$

where

$$\lambda_{i,\tau}(t) = \begin{cases} \mu_i(t) + \sum_{j\in[D]} \alpha_{ij,1} \int_\tau^t \varphi_{ij}(t-s)dN_s^j, & t > \tau, \\ \lambda_{i,\infty}(t), & 0 \le t \le \tau, \end{cases}$$

is the intensity for node i if the change-point happens at τ, and $\lambda_{i,\infty}(t) = \mu_i(t) + \sum_{j\in[D]} \alpha_{ij,0} \int_0^t \varphi_{ij}(t-s)dN_s^j$ is the intensity under the null hypothesis. Here, ∞ is used to indicate that the event that the change never happens.

Given assumed post-change parameters, $\{\alpha_{ij,1}\}_{i,j\in[D]}$, the stopping time for CUSUM is given by

$$T_{\mathrm{C}} = \inf\{t : \sup_{\tau<t} \ell_{t,\tau} > b\}, \tag{6}$$

where $\ell_{t,\tau}$ is the log-likelihood ratio statistic defined in Equation (5), and $b > 0$ is a prespecified threshold. The procedure stops when the log-likelihood ratio from some hypothesized change-point τ exceeds threshold b.

In contrast to the original CUSUM procedure (Page 1954) where the samples are taken in a discrete-time fashion, here the CUSUM statistic is continuous-time and has memory. In particular, due to the memory of the Hawkes process, the observations are *non-iid* and have complex temporal dependence. Because of this dependence, the simple recursive approach for standard CUSUM does not extend to the current (more complex) Hawkes process setting, and further developments are needed.

To derive an computationally efficient recursive algorithm for CUSUM in the network Hawkes process, we start with a lemma for the log-likelihood ratio $\ell_{t,\tau}$. This lemma shows that, although the supremum of the log-likelihood ratio statistic over the unknown change-point appears to be on a continuum, it will be obtained at the observed event times.

Lemma 1. Given the event times $\{t_i, i = 1, 2, \ldots\}$, for any fixed t and k, $t > t_k$, it follows that $\sup_{t_k<\tau\le\min\{t_{k+1},t\}} \ell_{t,\tau} = \lim_{\tau\to t_k^+} \ell_{t,\tau} =: \ell_{t,t_k^+}$, and $\sup_{0\le\tau\le t_1} \ell_{t,\tau} = \ell_{t,0}$.

The proof of this lemma is provided in the supplementary materials. Lemma 1 says that we only need to consider the values of the log-likelihood evaluated as the past event times, rather than a continuum of possible values for τ. As we show later, this will greatly simplify the computation of the log-likelihood ratio statistic.

For computational efficiency, we can further simplify the calculation in (6) (which involves $\sup_{\tau<t} \ell_{t,\tau}$) by considering t on a discretized grid with a prespecified grid size $\gamma > 0$. In this case, we would only need to calculate the detection statistic $\sup_{\tau<n\gamma} \ell_{n\gamma,\tau}$ for $n \in \mathbb{Z}$.

Finally, with the discretization for both τ and t, the log-likelihood ratio $\ell_{n\gamma,t_k^+}$ and $\ell_{(n+1)\gamma,t_k^+}$ have the following relationship, given $n\gamma \ge t_k$,

$$\ell_{(n+1)\gamma,t_k^+} = \ell_{n\gamma,t_k^+} + \sum_{i\in[D]} \int_{n\gamma}^{(n+1)\gamma} \log\left(\frac{\lambda_{i,t_k^+}(s)}{\lambda_{i,\infty}(s)}\right) dN_s^i - \sum_{i\in[D]} \int_{n\gamma}^{(n+1)\gamma} (\lambda_{i,t_k^+}(s) - \lambda_{i,\infty}(s))ds. \tag{7}$$

Equation (7) provides a recursive procedure for computing the log-likelihood ratios $\ell_{n\gamma,t_k^+}$, as long as the one-dimensional integrals can be evaluated or approximated numerically. If we have additional access to the cumulative kernels $\Phi_{ij}(t) = \int_0^t \varphi_{ij}(s)\mathbb{1}(s \ge 0)ds$, where $\mathbb{1}(\cdot)$ is the indicator function, this recursion can be computed without numerical integration as follows:

$$\begin{aligned} \ell_{(n+1)\gamma,t_k^+} = \ell_{n\gamma,t_k^+} &+ \sum_{i\in[D]} \int_{n\gamma}^{(n+1)\gamma} \log\left(\frac{\lambda_{i,t_k^+}(s)}{\lambda_{i,\infty}(s)}\right) dN_s^i \\ &+ \sum_{i,j\in[D]} \int_0^{(n+1)\gamma} \alpha_{ij,0}(\Phi_{ij}((n+1)\gamma - s) \\ &- \Phi_{ij}(n\gamma - s))dN_s^j \\ &- \sum_{i,j\in[D]} \int_{t_k^+}^{(n+1)\gamma} \alpha_{ij,1}(\Phi_{ij}((n+1)\gamma - s) \\ &- \Phi_{ij}(n\gamma - s))dN_s^j. \end{aligned} \tag{8}$$

Algorithm 1 summarizes the key steps in the proposed CUSUM procedure. We provide a further remark on the choice of the grid size $\gamma > 0$. Note that different choices of γ

Algorithm 1: CUSUM for network Hawkes processes

Input: event times $\{(t_i, u_i), i = 1, 2, \ldots\}$; pre-change parameters $\{\alpha_{ij,0}\}_{i,j\in[D]}$; post-change parameters $\{\alpha_{ij,1}\}_{i,j\in[D]}$; threshold b; grid size γ;
Initialization: $n \leftarrow 0, S_0 \leftarrow 0, \ell_{0,0} \leftarrow 0$;
while $S_{n\gamma} < b$ **do**
  $n \leftarrow n + 1$;
  for $\tau \in \{t_k^+ : t_k < n\gamma\} \cup \{0\}$ **do**
    if $\tau < (n-1)\gamma$ **then**
      calculate $\ell_{n\gamma,\tau}$ using (7) or (8);
    else
      calculate $\ell_{n\gamma,\tau}$ using (5);
    end
  end
  $S_{n\gamma} \leftarrow \max_\tau \ell_{n\gamma,\tau}$;
  if $S_{n\gamma} > b$ **then**
    Output: stopping time $T_{\mathrm{C}} \leftarrow n\gamma$, $\hat{\tau} \leftarrow \arg\max_\tau \ell_{n\gamma,\tau}$;
  end
end

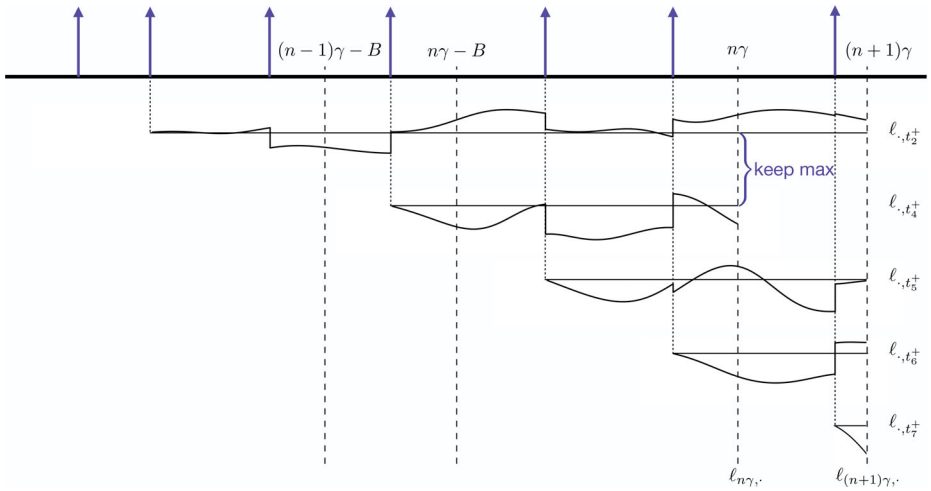

Figure 2. Illustration of the memory-efficient CUSUM algorithm. At time $t = n\gamma$, only one of the log-likelihood ratios $\ell_{n\gamma,\tau}$ for potential change-points $\tau < (n-1)\gamma - B$ is kept in memory (in this example, the one immediately after the second event). When updating the statistics to $t = (n+1)\gamma$, all log-likelihood ratios $\ell_{n\gamma,\tau}$ for $\tau \le n\gamma - B$ are compared and again only the largest one is kept for future updates.

corresponds to different updating frequencies for the CUSUM statistics, hence, γ is an important parameter for the algorithm. There is a performance tradeoff in choosing the parameter γ: a very large choice of γ may result in a large detection delay, whereas a very small γ may leads to unnecessary computational complexity. The effect of γ on the algorithm is investigated further in numerical studies, and it would be interesting to develop a method for choosing γ adaptively.

3.3. Modification for Memory Efficiency

We now present a modification of Algorithm 1 to improve memory efficiency of the procedure. Note that the "exact" CUSUM algorithm (Algorithm 1) requires keeping track of the entire history of events, since all past events influence the intensity function. However, in practice, events which happened long ago will have little bearing on the current intensity, since its mutually-exciting property diminishes over time. One way to improve memory efficiency is to simply remove such events, since their influence on the present and the future (and thereby the performance of the method) would be small.

Consider the following *truncated kernel* with a width $B > 0$:

$$\widetilde{\varphi}_{ij}(t) = \begin{cases} \varphi_{ij}(t), & t \le B, \\ 0, & t > B. \end{cases}$$

Under the truncated kernel, an event has no influence over the whole process after B into the future, and we only need to keep events during $[t-B, t]$ in our memory for computation. With the truncated kernel $\widetilde{\varphi}_{ij}$, the intensity for node i can then be approximated by

$$\widetilde{\lambda}_{i,\tau}(t) = \begin{cases} \mu_i(t) + \sum_{j\in[D]} \alpha_{ij,1} \int_{t-B}^{t} \varphi_{ij}(t-s) dN_s^j, & \tau < t-B, \\ \mu_i(t) + \sum_{j\in[D]} \alpha_{ij,1} \int_{\tau}^{t} \varphi_{ij}(t-s) dN_s^j, & t-B \le \tau \le t, \\ \mu_i(t) + \sum_{j\in[D]} \alpha_{ij,0} \int_{t-B}^{t} \varphi_{ij}(t-s) dN_s^j, & \tau > t. \end{cases} \tag{9}$$

Here for all $\tau \ge 0$, $\widetilde{\lambda}_{i,\tau}(t)$ only depends on event data during $[t-B, t]$. Moreover, the intensity for $\tau < t-B$ does not depend on τ, which enables us to update the log-likelihood ratio recursively for small τ. If we also have access to the cumulative kernels $\Phi_{ij}, i, j \in [D]$, the recursion step in Equation (8) can be approximated by

$$\begin{aligned} \ell_{(n+1)\gamma,t_k^+} = \ell_{n\gamma,t_k^+} &+ \sum_{i\in[D]} \int_{n\gamma}^{(n+1)\gamma} \log\left(\frac{\widetilde{\lambda}_{i,t_k^+}(s)}{\widetilde{\lambda}_{i,\infty}(s)}\right) dN_s^i \\ &+ \sum_{i,j\in[D]} \int_{n\gamma-B}^{(n+1)\gamma} \alpha_{ij,0}(\widetilde{\Phi}_{ij}((n+1)\gamma - s) \\ &- \widetilde{\Phi}_{ij}(n\gamma - s)) dN_s^j \\ &- \sum_{i,j\in[D]} \int_{\max\{t_k^+, n\gamma-B\}}^{(n+1)\gamma} \alpha_{ij,1}(\widetilde{\Phi}_{ij}((n+1)\gamma - s) \\ &- \widetilde{\Phi}_{ij}(n\gamma - s)) dN_s^j, \end{aligned} \tag{10}$$

where $\widetilde{\Phi}_{ij}(t) = \Phi_{ij}(\min\{t, B\})$, and the summation is taken only for event times during $[n\gamma - B, (n+1)\gamma]$.

Figure 2 shows an illustration of the memory-efficient CUSUM procedure. The details are summarized in Algorithm 2.

The computing and memory resources required for this procedure depend on both the network size and the number of events observed while monitoring. Each time we update the CUSUM statistics from $t = (n-1)\gamma$ to $n\gamma$, we track the log-likelihood ratios $\ell_{(n-1)\gamma,\cdot}$ for potential change-points τ that are event times from $(n-2)\gamma - B$ to $n\gamma$, along with a summary of the log-likelihood ratios for $\tau < (n-2)\gamma - B$. For each $\ell_{\cdot,\tau}$, when updating from $t = (n-1)\gamma$ to $t = n\gamma$ by (10), each $\widetilde{\lambda}_{\cdot,\cdot}(\cdot)$ is calculated using at most $N_{n\gamma} - N_{(n-1)\gamma-B}$ events, and the integral over counting measure is the summation over at most $(N_{n\gamma} - N_{(n-1)\gamma-B})D$ terms. Overall, the computation complexity for one update is $O((N_{n\gamma} - N_{(n-2)\gamma-B})^3 D)$. Under the stability condition $\|A\| < 1$ (where $\|\cdot\|$ is the spectral norm), a multi-dimensional Hawkes process can be shown to have a finite third-order moment and is ergodic (Achab et al. 2017). In this case, the computation complexity of the memory-efficient CUSUM is linear in the time t.

Note that the dependency of the computation complexity on network size D can be eliminated if, for each $j \in [D]$, $(\widetilde{\Phi}_{ij})_{i\in[D]}$ are identical. When adding the term on the second and third

Algorithm 2: Memory-efficient CUSUM for network Hawkes processes

Input: event times $\{(t_i, u_i), i = 1, 2, \ldots\}$; pre-change parameters $\{\alpha_{ij,0}\}_{i,j\in[D]}$; post-change parameters $\{\alpha_{ij,1}\}_{i,j\in[D]}$; threshold b; grid size γ;
Initialization: $n \leftarrow 0, S_0 \leftarrow 0, \ell_{0,0} \leftarrow 0, \hat{\tau} \leftarrow 0$;
while $S_{n\gamma} < b$ **do**
  $n \leftarrow n + 1$;
  for $\tau \in \{t_k^+ : (n-1)\gamma \le t_k < n\gamma\}$ **do**
    calculate $\ell_{n\gamma,\tau}$ using (5) with (9);
  end
  for $\tau \in \{t_k^+ : (n-1)\gamma - B \le t_k < (n-1)\gamma$ **do**
    calculate $\ell_{n\gamma,\tau}$ using (7) with (9) or (10);
  end
  for $\tau \in \{t_k^+ : (n-2)\gamma - B \le t_k < (n-1)\gamma - B\}$ **do**
    if $\ell_{(n-1)\gamma,\tau} > \ell_{(n-1)\gamma,\hat{\tau}}$ **then**
      $\hat{\tau} \leftarrow \tau$;
    end
  end
  update $\ell_{n\gamma,\hat{\tau}}$ from $\ell_{(n-1)\gamma,\hat{\tau}}$ using (7) with (9) or (10);
  $S_{n\gamma} \leftarrow \max_\tau \ell_{n\gamma,\tau}$;
  if $S_{n\gamma} > b$ **then**
    Output: stopping time
    $T_C \leftarrow n\gamma, \hat{\tau} \leftarrow \arg\max_\tau \ell_{n\gamma,\tau}$;
  end
end

row in (10), the summation over i can be precomputed by saving $\sum_i \alpha_{ij,0}$, $\sum_i \alpha_{ij,1}$ for each $j \in [D]$. Thus, it only takes $O((N_{n\gamma} - N_{(n-2)\gamma-B})^3)$ steps to perform the nth update.

Regarding memory usage of the procedure, note that for the updates up until time $(n-1)\gamma$, we only need to keep track of event data with an occurrence time after $(n-1)\gamma - B$. Hence, the memory usage for the nth update (apart from loading the network parameters) is $O(N_{n\gamma} - N_{(n-1)\gamma-B})$, the average is a constant with respect to time t.

We illustrate the effect of the truncation width B using a numerical example. The model is described in Section 5, where the kernel functions are all exponential $\varphi_{ij}(t) = \beta e^{-\beta t}, t \ge 0$, $\forall i, j \in [D]$ with $\beta = 1$. Figure 3 shows the comparison between CUSUM statistics with and without kernel truncation. Both CUSUM statistics with the truncated kernel have the same trend as the exact CUSUM statistic. When $B = 1/\beta$, 63.2% of the cumulative influence (which corresponds to the integral of the truncated kernel since the complete influence kernel integrates to one) is preserved, and the truncated statistic deviates from the exact CUSUM, which may result in a false alarm. When $B = 2/\beta$, 86.5% of the cumulative influence is preserved, and there appears to be little difference between the truncated and exact CUSUM statistics.

Performance. We discuss the performance of the proposed procedure via two widely used metrics for sequential change-point detection: (a) Average Run Length (ARL), defined as the expected value of the stopping time when there is no change, that is, $\mathbb{E}_\infty[T]$, where $\mathbb{P}_\infty$ is the probability measure on the sequence of event times when the change never occurs, and $\mathbb{E}_\infty$ is its corresponding expectation; (b) Expected Detection Delay (EDD), defined as the expected delay between the stopping time and the true change-point. Two common definitions for EDD can be found in Lorden (1971) and Pollak (1985), both of which consider the worst-case delay over all possible change-point values. In particular, if the true change-point is κ, then the EDD can be defined as $\mathbb{E}_\kappa[T - \kappa | T > \kappa]$, where $\mathbb{P}_\kappa$ denotes the probability measure on the observations when the change occurs at time κ, and $\mathbb{E}_\kappa$ denotes the corresponding expectation. The theoretical properties of the ARL and EDD and be found in Section B in the supplementary materials.

4. Alternative Detection Procedures

In practical problems, the post-change parameters are not always known due to the lack of anomalous data (e.g., there could be various types of anomalies, and one may not know which anomaly to expect). This section discusses two alternate approaches to change-point detection on the Hawkes process: the score statistics and the GLR statistics. Neither method requires any knowledge of the post-change parameters.

4.1. Score Statistics

We consider the score statistics for constructing a detection procedure. The score statistic can detect any deviations from the null hypothesis (Xie and Siegmund 2012). It is particularly suitable for detecting small deviations (i.e., locally most efficient) and does not require estimating post-change parameters. The score function is defined as the derivative of the log-likelihood as in (3) over the parameters $\alpha_{ij,0}$, $i, j \in [D]$, on which we would like to detect the change. With $\boldsymbol{\alpha}_i = (\alpha_{ij})_{j\in[D]} \in \mathbb{R}^{D\times 1}$, the score function on each node i is

$$\frac{\partial \ell_t}{\partial \boldsymbol{\alpha}_i} = -\int_0^t \frac{\partial \lambda_{i,\infty}(s)}{\partial \boldsymbol{\alpha}_i} ds + \int_0^t \lambda_{i,\infty}^{-1}(s) \frac{\partial \lambda_{i,\infty}(s)}{\partial \boldsymbol{\alpha}_i} dN_s^i, \quad (11)$$

since $\lambda_{i,\infty}$ only depends on the parameter $\boldsymbol{\alpha}_i$. Then the full score function has become $\frac{\partial \ell_t}{\partial \mathrm{vec}(A)} = \left(\frac{\partial \ell_t}{\partial \boldsymbol{\alpha}_1^T}, \ldots, \frac{\partial \ell_t}{\partial \boldsymbol{\alpha}_D^T}\right)^T$ where $A = (\alpha_{ij})_{ij\in[D]}$ and $\mathrm{vec}(A) = (\boldsymbol{\alpha}_1^T, \ldots, \boldsymbol{\alpha}_D^T)^T$. In Theorem 3.4 of Ogata (1978), it is shown that the limiting distribution of the score function at the true parameter $A_0 = (\alpha_{ij,0})_{i,j\in[D]}$ is normally distributed, that is, $\frac{1}{\sqrt{t}} \frac{\partial \ell_t}{\partial \mathrm{vec}(A)}\Big|_{A=A_0} \to \mathcal{N}(0, I_0)$, where

$$I_0 = \lim_{t\to\infty} \frac{1}{t} \mathbb{E}\left[\frac{\partial \ell_t}{\partial \mathrm{vec}(A)} \frac{\partial \ell_t}{\partial \mathrm{vec}^T(A)}\right]\Big|_{A=A_0} = -\lim_{t\to\infty} \frac{1}{t} \mathbb{E}\left[\frac{\partial^2 \ell_t}{\partial \mathrm{vec}(A) \partial \mathrm{vec}^T(A)}\right]\Big|_{A=A_0}$$

is the Fisher information matrix. Note that the Fisher information here is a diagonal block matrix, since for any $i, j, i', j' \in [D]$, $i \ne i'$, $\lambda_{i,\infty}$ is a constant with respect to $\alpha_{i'j'}$, and $\frac{\partial^2 \ell_t}{\partial \alpha_{ij} \partial \alpha_{i'j'}} = 0$. Each block of I_0 corresponds to $\boldsymbol{\alpha}_i, i \in [D]$, the influence from all nodes to node i. The limiting distribution of the score function at the true parameter can then be shown to be

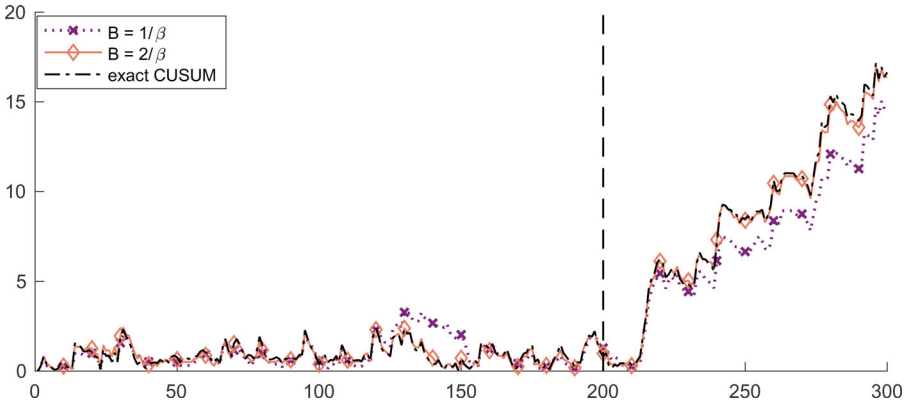

Figure 3. Comparison between CUSUM statistics with different truncation level B. The change-point occurs at time 200.

$\frac{1}{t}\frac{\partial \ell_t}{\partial \text{vec}^T(A)} I_0^{-1} \frac{\partial \ell_t}{\partial \text{vec}(A)}\Big|_{A=A_0} \to \chi^2_{D^2}$, where χ^2_ν is the Chi-squared distribution with degrees-of-freedom ν.

For change-point detection, we adopt the conventional sliding window approach, that is, calculating the score statistic inside the sliding window $[t-w, t)$ for a suitably chosen window length w, and raise an alarm whenever the statistic exceeds the threshold b. The corresponding stopping time can be written as:

$$T_S = \inf\left\{t : \frac{1}{w}\frac{\partial(\ell_t - \ell_{t-w})}{\partial \text{vec}^T(A)} I_0^{-1} \frac{\partial(\ell_t - \ell_{t-w})}{\partial \text{vec}(A)}\Big|_{A=A_0} \geq b\right\}.$$

With a sufficiently large window length, the score statistic under the null hypothesis should be around D^2, the expected value of the Chi-squared random variable $\chi^2_{D^2}$. After the change-point, the expected score function at the pre-change parameters is no longer 0. We would expect the score statistic to be noticeably larger than D^2, and thus, the change is detected.

Like CUSUM statistics, a memory-efficiency problem arises for the score statistics, since the intensity $\lambda_{i,\infty}$ depends on the whole history of events. We can again replace $\lambda_{i,\infty}$ with $\widetilde{\lambda}_{i,\infty}$ using the aforementioned truncated kernels (a similar approach can also be used for the GLR statistics, discussed next). For practical reasons, we would also need to choose a grid size γ and compute the score statistics only on the resulting grid. We discuss the relation between grid size and EDD for a fixed ARL later in Section 5.

4.2. GLR Statistics

When the post-change parameters are unknown, another way to perform change-point detection is via the generalized likelihood ratio (GLR) statistic. The idea is to find the parameters which best fit the data, then compare the likelihood ratio between the fitted parameters and the pre-change ones. This GLR statistics approach for multi-dimensional Hawkes processes was discussed in Li et al. (2017). Using a sliding window of fixed length w, the log-likelihood ratio can be defined within each window as

$$\ell_{t,t-w,\hat{A}} = \sum_{i\in[D]} \int_{t-w}^{t} \log\left(\frac{\hat{\lambda}_{i,t-w}(s)}{\lambda_{i,\infty}(s)}\right) dN_s^i - \int_{t-w}^{t} (\hat{\lambda}_{i,t-w}(s) - \lambda_{i,\infty}(s))ds,$$

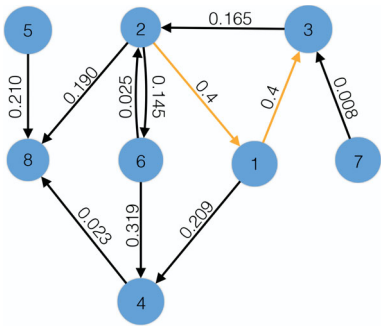

Figure 4. Visualizing the simulated network Hawkes model. Here, the background intensity is proportional to the size of each node, with edges indicating directed influences between nodes. Orange edges show the topological change after the change-point.

where $\hat{\lambda}_{i,t-w}$ is the intensity assuming $t-w$ is the change-point and $\hat{A}$ is the post-change parameter estimates. $\hat{\lambda}_{i,t-w}(s) = \mu_i(s) + \sum_{j\in[D]} \hat{\alpha}_{ij} \int_{t-w}^{s} \varphi_{ij}(s-v) dN_v^j$, and $\hat{A} = (\hat{\alpha}_{ij})_{i,j\in[D]}$ is the parameter that maximizes the likelihood in the current window $[t-w, t]$:

$$\hat{A} = \arg\max_A \sum_{i\in[D]} \int_{t-w}^{t} \log \lambda_{i,t-w}(s) dN_s^i - \int_{t-w}^{t} \lambda_{i,t-w}(s) ds.$$

A change is detected when the log-likelihood ratio exceeds certain threshold b:

$$T_G = \inf\{t : \ell_{t,t-w,\hat{A}} \geq b\}.$$

For each window, a convex optimization problem is solved to find the maximum likelihood estimate $\hat{A}$ that best fits the data. However, this operation makes the GLR statistic computationally more expensive than CUSUM and the score statistic. To address this issue, we can use $\hat{A}$ from the previous window as an initialization for the gradient descent algorithm to find the MLE in the next step—this "warm-start" may lead to faster convergence for finding the MLE.

Compared with the score statistic, the GLR statistic is computationally more expensive, and it does not necessarily have better performance (which can be partly due to the optimization error in computing MLE), as shown in our example in Section 5. However, the GLR statistic is numerically more stable than the score statistic, especially for large networks. The reason is that we usually may not estimate the Fisher information with high

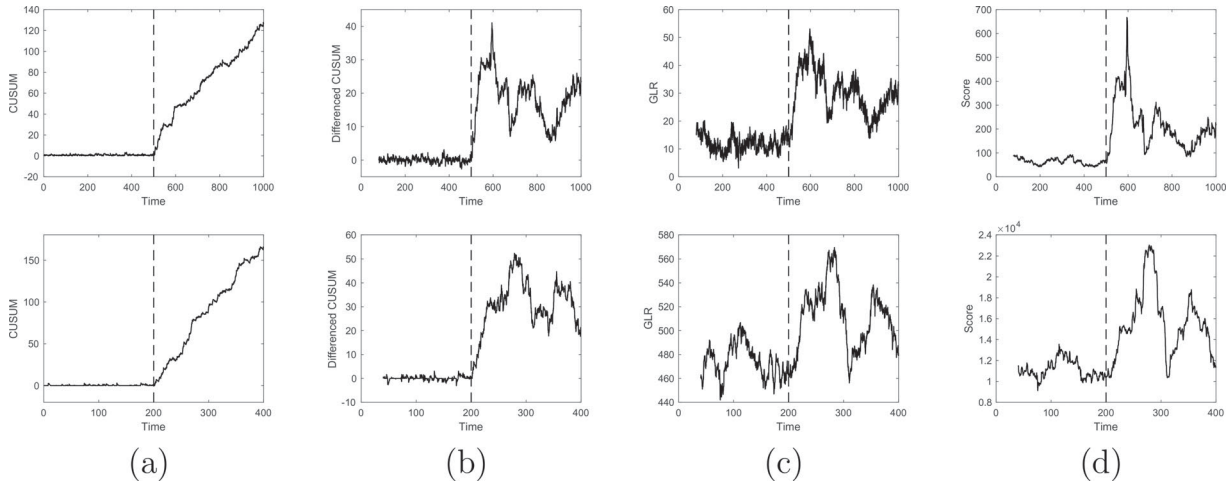

Figure 5. Examples of detection statistics. The first row is generated on the small network of size 8, with window length $w = 80$ for both GLR and score statistics. The second row is generated on the large network of size 100, with window length $w = 40$. Both rows use the same sequence of events, and the window length is chosen numerically to optimize performance (see discussions in 5.1). The vertical dashed line is the time of the true change-point κ. Column (b) shows the difference of CUSUM between time t and $t - w$.

accuracy. Even provided with the exact pre-change parameter A_0, there is no close-form solution for the Fisher information, and it can only be estimated by simulation or real data. The score statistic also involves inverting the Fisher information, which can suffer from a high condition number and numerical instability.

5. Numerical Experiments

In this section, we compare several change-point detection procedures using simulated examples on a small network with 8 nodes and a larger one with 100 nodes. The small network is shown in Figure 4. The base intensity is proportional to the size of the node ranging from 0.5 to 1, and the edges indicate the asymmetrical influences between nodes. The edges in black are the pre-change parameters, while the edges in orange show a topological change between nodes 1, 2, and 3. There are two emerging edges after the change-point, while all other edges remain the same. For the 100-node network, the background intensity is set to be 0.05 for all nodes. All nodes work independently in the pre-change scenario, while the post-change network consists of 200 directed edges with weight 0.2 chosen at random.

Throughout this section, we use the exponential decaying kernel $\varphi_{ij}(t) = e^{-t}, \forall i, j$ to generate event data. The kernel functions are truncated at $B = 5$ to leverage the computational and memory efficient procedures in Section 3.2. The update rate is set at $\gamma = 5$.

Figure 5 visualizes the CUSUM, GLR, and score statistics on the same sequence of events for both networks. As expected, CUSUM grows steadily larger after the change-point for both network settings. The differenced CUSUM, GLR and score statistics show similar post-change fluctuations, as they can be understood as various measurements of how the process within the sliding window differs from the pre-change scenario. The proposed CUSUM procedure appears to be the least noisy before the change-point, which may be another explanation of why CUSUM has the best performance (see later), apart from its cumulative nature.

5.1. Performance Comparison

We investigate the performance of these detection statistics by comparing a plot of its EDD versus log(ARL). From theoretical analysis, we expect such plot of the CUSUM statistic to be close to linear. We first introduce a simple baseline: the Shewhart control chart (Shewhart 1925, 1931), which counts the number of events in a sliding window and stops when the number of events falls out of a specific range:

$$T_{\text{Sh}} = \inf \left\{ t : \sum_{i \in [D]} (N_t^i - N_{t-w}^i) \notin [b_1, b_2] \right\}.$$

This Shewhart chart can detect changes in average intensity. Note that it does not take into account the network structure or the location of events. In this example, we only consider the case where the average intensity will be increased after the change-point, and thus, choose a one-sided interval by letting $b_1 = 0$.

Figure 6 shows the comparison results. For both networks, the CUSUM procedure with exact post-change parameters achieves the best performance, followed by the score statistic and the GLR. All three methods are better than the baseline Shewhart chart. Window lengths are chosen numerically for the latter three procedures, such that the performance is (approximately) optimal for an ARL between 500 and 50,000. The details are shown in Table 1. When simulating the EDD, we set the change-point κ to be slightly larger than the window length w so that it is possible for the EDD to be less than w.

Running time. Table 1 also summarizes the average time in seconds needed to compute the selected procedures over a time horizon 50,000, with no change-point on a personal computer (Apple M1 chip). The base intensity of the networks are scaled such that the two networks have similar average number of events. Clearly apart from the Shewhart chart, CUSUM enjoys the quickest running time and remains computationally efficient as the network size increases, thus, demonstrating the scalability of the proposed approach.

Misspecification in post-change parameters. We also consider the CUSUM procedure when the assumed post-change parameter differs from the true parameters in Figure 6, indicating a

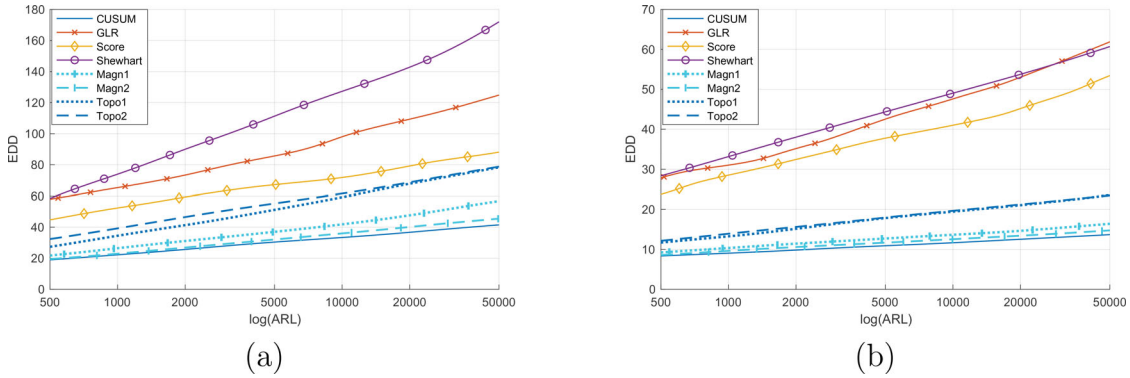

Figure 6. Performance comparison of different detection statistics on (a) network of size 8, (b) network of size 100.

Table 1. Optimal window length and running time of selected procedures.

		CUSUM	GLR	Score	Shewhart
Network size = 8	w	NA	80	80	120
	time	1.508	37.14	3.013	0.046
Network size = 100	w	NA	40	40	60
	time	1.304	54.89	12.71	0.030

model mismatch. For the small network of size 8, while the real change occurs on two pairs of nodes making the influence factor both 0.4, we consider a CUSUM procedure which assumes a correct post-change network topology, but with magnitudes of the post-change parameters on the edges to be 200% (Magn1) and 50% (Magn2) of the true magnitudes, respectively. We also consider a CUSUM procedure which assumes the correct magnitude of the post-change parameter, but an incorrect post-change network topology. In one case, a change in the influence from $2 \to 1$, $1 \to 3$, $6 \to 1$, $1 \to 7$ is expected (Topo1), and in another, only the change in the influence from $2 \to 1$ is expected (Topo2). For the large network of size 100, we consider the case when the magnitudes of the post-change parameters are 200% (Magn1) and 50% (Magn2) of the true ones. For the topological model mismatch, recall that the true change happens on 200 edges, we select 200 more edges (Topo1) or drop 50 edges (Topo2), both at random as the misspecified cases. Even with misspecified post-change parameters in either influence magnitude or network topology, the CUSUM procedure can still achieves better performance than the GLR and the score procedures for both network settings. This demonstrates that the proposed CUSUM procedure is reasonably robust to the misspecification of the post-change parameters.

Though misspecified, the cases provided above still partly capture the true change. When the estimated post-change parameters deviate greatly from the true parameters, the CUSUM procedure can fail to achieve an EDD linear in log(ARL), as shown in Figure 7. For the network of size 8, the post-change parameters is estimated to perceive a change in the influence from $6 \to 1$, $1 \to 7$, with a magnitude of 0.4. For the network of size 100, the misspecified post-change parameters select 200 edges randomly (independent of the true topology of the change) with a magnitude of 0.2.

In real scenarios, the abrupt change may represent an unexpected anomaly, and we do not have enough data to estimate the post-change parameters. In such cases, we may choose a targeted topology of the post-change parameters to detect a certain type of structural change and choose the magnitude to reflect a minimum size of the change to be detected. For certain applications, it is also possible to enumerate the potential changes and run several detection procedures in parallel, each responsible for monitoring the process against one type of change. We can also see which type of change causes an alarm to help identify the change pattern and location. Alternatively, there are also adaptive CUSUM procedures (Xie et al. 2020), which use "future" samples to estimate a potential post-change parameter and use as a plug-in estimator in the CUSUM statistics. However, such methods may incur an additional delay in detection.

5.2. Effect of Grid Size γ

As mentioned earlier, the choice of the grid size γ involves a tradeoff between algorithm performance and computational complexity. To investigate this tradeoff, we compare the proposed CUSUM, the GLR, and the score statistics with a grid size γ ranging from 0.1 to 50 on the network of size 8. For the EDD, we assume that the change-point is uniformly distributed between two grid points. Figure 8 shows the effect of the grid size on CUSUM, the GLR, and the score procedure. We see that a large grid size γ results in both a larger ARL and EDD. If we instead tune the threshold b to fix the ARL, the EDD may still increase with a larger γ.

To understand the effect of γ on computation complexity, we will consider the GLR statistic as the computation for the GLR is the most expensive. To solve the convex optimization problem for each window, we use the EM algorithm as described in Li et al. (2017), and terminate when the update in the log-likelihood is less than 10^{-4}. Figure 8(c) shows the average iterations needed per window for different grid sizes. We see that, as γ increases, the computation required (in terms of number of iterations) increases as well, which matches intuition.

6. Detecting Neuronal Network Population Code Change

We now return to the motivating problem of detecting population code changes in neuronal networks. The data considered are neural spike trains, which record the sequence of times when a neuron fires an action potential. The multivariate Hawkes processes from Section 2.2 have been used for modeling spike train data (Lambert et al. 2018; Wang et al. 2020b), and capture

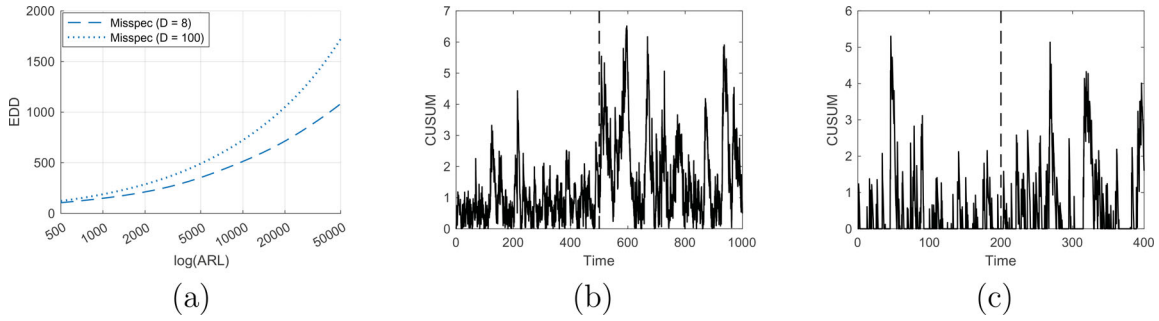

Figure 7. Examples when CUSUM fails to detect under misspecified post-change parameters. (a) The EDD versus log(ARL) plot. (b and c) Examples of the CUSUM procedure under misspecification for the network of size 8 and 100, using the same sequence of events with Figure 5, respectively.

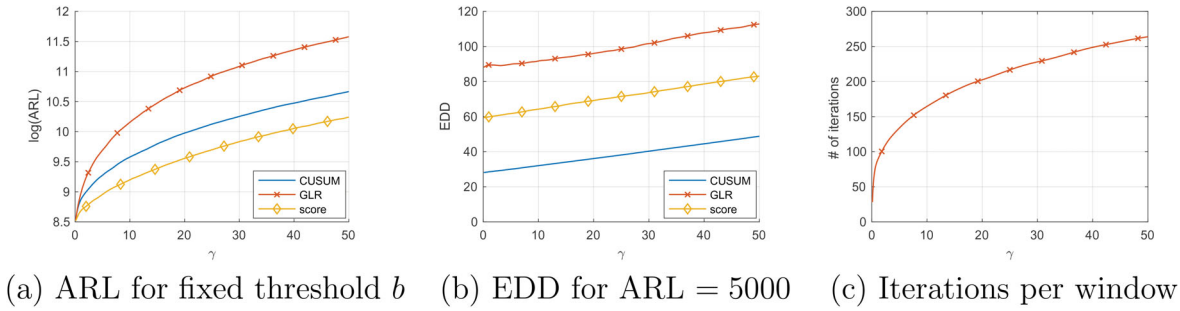

Figure 8. Effect of the grid size γ on CUSUM, GLR, and score statistics. The window length for the GLR and the score statistics is 60 and for Shewhart is 120, which are optimized for better EDD performance. (a) shows the effect of γ on the ARL at fixed threshold $b = 6.319, 37.66, 148.2$ for CUSUM, the GLR and the score statistics, respectively, making ARL $= 5000$ at $\gamma = 0.1$. (b) shows the effect of γ on the performance by tuning b so that ARL is 5000. (c) shows the average number of iterations per window needed in the GLR statistic for the optimization problem to converge.

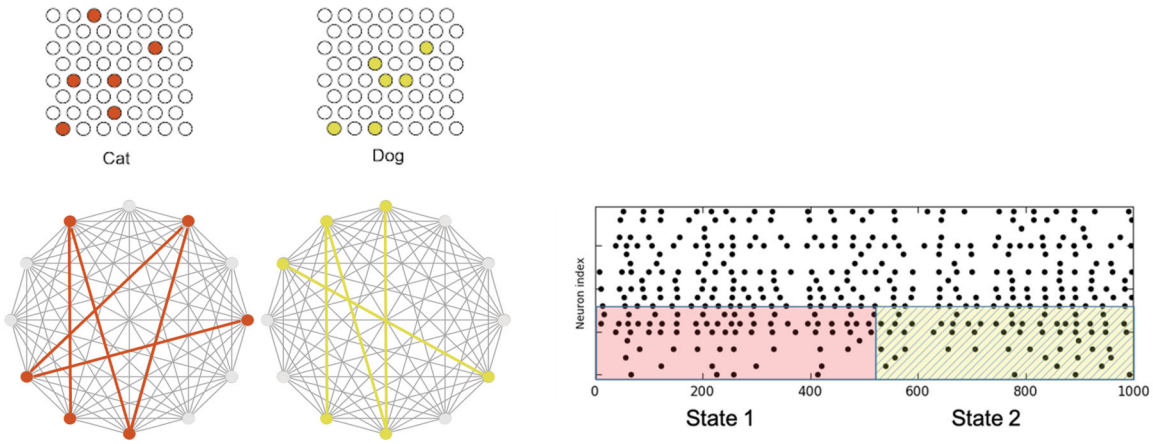

Figure 9. (Left) A plausible network topology for "Cat" and "Dog"; (Right) Spike train data for each neuron under a population code change.

two appealing features for neuronal networks. First, the base intensities μ_i capture noisy influences on neuron i, resulting from either unobserved neurons or external stimuli. Second, the influence parameters α_{ij} capture the functional influence from neuron j to neuron i due to electrochemical dynamics.

We are interested in detecting the change-point in the underlying population code from neural data. These are *abrupt* changes, as the behavior of populations of neurons respond quickly (usually in just a few ms) to changing input. Population codes are a distributed representation of information used widely across many neural architectures and have been most widely documented in the cortex. As opposed to dense representations, population codes consist of sparsely activated subsets of neurons in which the information is distributed amongst the entire subset. Figure 9 illustrates this idea. The left plots show a plausible neuronal network topology for the population coding of seeing a cat or a dog. The right plots show the corresponding spike train data on the neuronal network, as one changes states from a cat to a dog. Identifying this population code change-point from experimental data provides scientists a better understanding of the role of each neuron, which can be used for repairing neuronal networks.

Here, the *sequential* nature of change-point detection can be advantageous for practical implementation. Real-time detection of biological neural networks is known as "continuous detection" in the neuroscience literature (see e.g., Goense and Ratnam 2003). This is in contrast to the more standard "trial-based" (or fixed-sample) testing, where the beginning and duration of the testing interval are predetermined. A key advantage of continuous detection over trial-based testing is a reduction in experi-

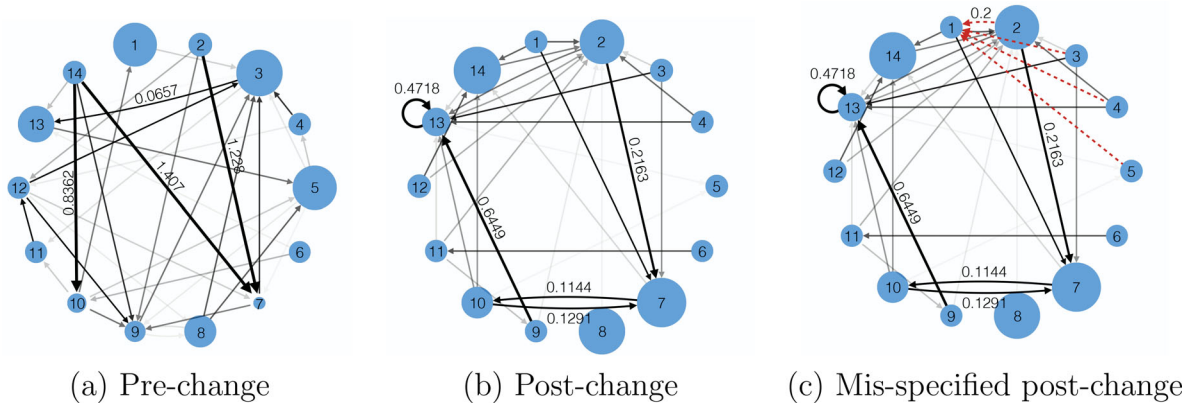

Figure 10. (a), (b) shows the estimated pre-change and post-change parameters. The size of the nodes are proportional to their background intensities ranging from 0 to 0.016. The opacity of the unlabeled edges are proportional to their weights, ranging from 0 to 0.06. (c) shows a case of post-change topology misspecification in CUSUM statistic.

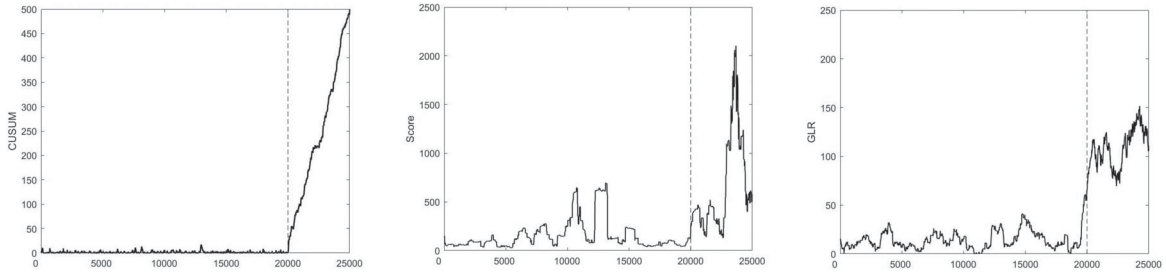

Figure 11. The CUSUM, score, and GLR statistics for the neuronal network application with $D = 14$ nodes. The true change-point is indicated by the dashed line.

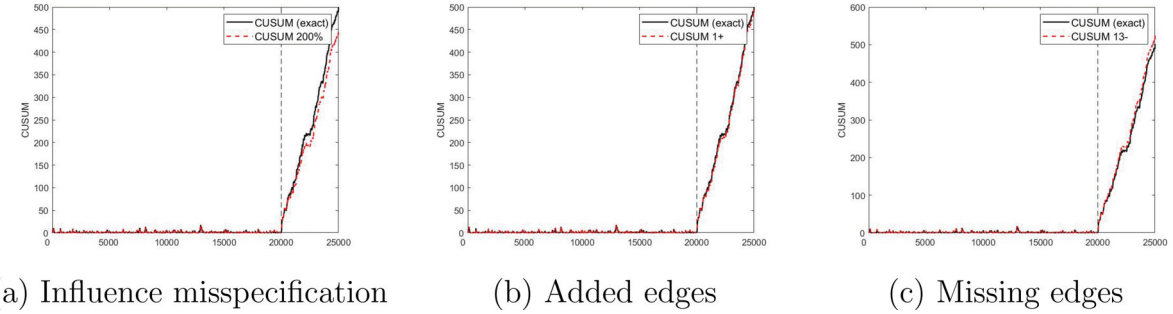

Figure 12. CUSUM statistics under different misspecifications of post-change parameters. The red line shows the CUSUM statistic under an "exact" specification of post-change parameters, and the dash line indicates the change-point.

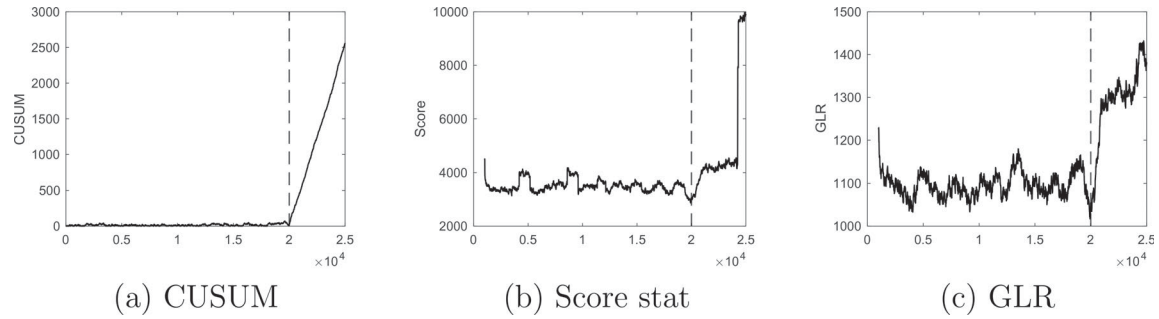

Figure 13. The CUSUM, score, and GLR statistics for the neuronal network application with $D = 80$ nodes. The true change-point is indicated by the dashed line.

mental sample size (Goense and Ratnam 2003): the experiment terminates after a change-point is detected, and does not need to run for the full testing period. This yields considerable cost savings for experiments, and speeds up the decision-making procedure. Continuous change-point detection is a capability vital to the success of neural interfaces, devices that monitor and decode the activity of a subject's brain. These neural interfaces being widely used in modern neuroengineering problems

to restore capabilities to patients, for example, manipulating a robotic hand or even typing on a virtual keyboard. Underlying the use of real-time neural interfaces is the ability to detect changes in population codes in real time.

While there have been significant advances in neuroimaging technology, it can still be quite costly to record fine-scale spiking data through in-vivo (i.e., *physical*) experiments. To illustrate the proposed method, we instead simulate the spike train data using the PyNN package with the NEURON simulator, which implements the neuronal model in Brette and Gerstner (2005). We build off previous work in neural simulation and use a network of exponential integrate-and-fire neurons with spike triggered and sub-threshold adaptation currents. This can be viewed as a *computer* experiment surrogate for the expensive physical experiments, which we cannot obtain due to high costs.

The simulation set-up is as follows. We first simulate several small networks of neurons in a balance of 80–20 excitatory to inhibitory neurons, with network size fixed at $D = 14$ neurons. From this, we obtain a continuous readout of each neuron spiking data. Each neuron then receives a small Gaussian noise current, representing random external influence on the network. In addition, a select few neurons receive inputs from an external source, which represents the phenomenon of sparse population coding. The neurons that spike at higher rates form a distributed representation of the network state.

We then randomly selected two such subsets of neurons, representing two different states. We simulate the network in the first state for a long time (from $t = 0$–20,000 ms) to learn network dynamics and structure under the first population code. The pre-change Hawkes process parameters are obtained via maximum likelihood estimation (MLE) on the pre-change spike train data. After $\kappa = 20{,}000$ ms, we then simulate a change from the first to the second state. The goal is to quickly detect the change-point κ in a sequential fashion from the simulated spike trains. For CUSUM, the post-change Hawkes process parameters are estimated via MLE on the post-change spike trains. For the score statistic, the pre-change Fisher information matrix I_0 is highly ill-conditioned when estimated from spike trains, so we instead use a slightly regularized estimate $I_0 + \lambda I$, where $\lambda = 1$ and I is an identity matrix. The estimated pre-change and post-change models are shown in Figure 10.

Figure 11 shows the CUSUM, GLR, and score statistics, respectively, with the dashed line indicating the change-point in population code. Both the score and GLR statistics use a window size of 1000 and an update rate of $\gamma = 5$. As in numerical experiments, we see that the CUSUM statistic increases rapidly after the change-point, which shows it is quite effective at detecting the underlying neuronal network changes. The score and GLR statistics are also noticeably larger after the change-point, with the increase in GLR more prominent than the increase for the score statistic. The increases in GLR and score statistics are noticeably lower than that for the proposed CUSUM procedure, which suggests that our method can better detect population code changes in neuronal networks.

Next, we consider the case where post-change parameter estimates are misspecified for the CUSUM statistic. This may arise, for example, when there is a lack of spike train data for post-change parameter estimation. We consider three scenarios for misspecification: (a) the post-change topology is correct, but the influence parameters are scaled at 200%, (b) the influence parameters are correct, but there are spurious edges on neuron 1 for the topology (see Figure 12(b)), (c) the post-change influence parameters are correct, but all the edges to neuron 13 are missing for the topology. Figure 12 shows the CUSUM statistics for these three scenarios, along with the "exact" CUSUM statistics, which use exact post-change MLEs. We see that the CUSUM is quite robust: its CUSUM statistics are quite close to the exact CUSUM for both influence and topology misspecifications. Hence, our method appears to efficiently detect population code changes, even under uncertainties in post-change parameter estimation.

Finally, we investigate the scalability of these methods by increasing the network size to $D = 80$ neurons, with all simulation and experimental settings fixed as before. Due to the inaccuracies in estimating the pre-change parameters, the CUSUM procedure has a positive drift in the pre-change scenario, which disagrees with the theoretical property needed for CUSUM—before the change, the expected drift should be negative; otherwise, there will be constant false alarms raised. To address this issue, as a common practice, we subtract a positive constant 0.2 from the increment when forming the CUSUM procedure from all $\ell_{n\gamma,\cdot}$, which will ensure the drift term has a negative expected value. Figure 13 shows the corresponding CUSUM, GLR, and score statistics, with the dashed line indicating the change-point in population code. We see similar observations as before: the CUSUM statistic increases rapidly after the change-point, whereas the increases in the GLR and score statistics are much more subtle, thus, indicating our approach can provide better detection of population codes. As in simulation experiments, the computation time favors our method: the proposed CUSUM procedure requires 0.44 sec on the 80-node network, whereas the GLR and score procedures require 32.48 and 4.63 sec, respectively. This shows the improved performance and efficiency of our recursive CUSUM approach.

7. Conclusion and Discussions

We have presented a new sequential CUSUM procedure for detecting change-point in the multi-dimensional self- and mutual-exciting point processes, that is, network Hawkes processes. By tackling the complex and long-term dependence between event times, we develop the CUSUM procedure that enjoys efficient recursive computation and memory efficiency if we employ truncation. Using numerical experiments, we showed that the CUSUM procedure yields improved performance over existing detection procedures (Shewhart-type) based on score statistics and generalized likelihood ratio (GLR) statistics. Moreover, we found that, although the CUSUM procedure requires specifying the post-change distribution parameters, it is fairly robust to parameter misspecification when it is possible to estimate the topology and magnitude of a potential abrupt change and outperforms existing methods in that setting. This can be partly explained by that these alternative methods are the Shewhart-type approaches (based on evaluating a detection statistic using a sliding window), which does not accumulate information from the past. We also demonstrated a realistic neuroengineering application of our procedure for neuronal network change-point detection.

Supplementary Materials

supplementary.pdf: Proofs and theoretical properties of the CUSUM procedure. (PDF file)

code.zip: Code reproducing detecting statistics examples and ARL versus EDD plots in the numerical experiments (Figures 5, 6, and Table 1.) (MATLAB code)

Funding

The work of Haoyun Wang, Liyan Xie, and Yao Xie were partially supported by an NSF CAREER CCF-1650913, and NSF DMS-1830210.

References

Achab, M., Bacry, E., Gaïffas, S., Mastromatteo, I., and Muzy, J.-F. (2017), "Uncovering Causality from Multivariate Hawkes Integrated Cumulants," *The Journal of Machine Learning Research*, 18, 6998–7025. [48]

Brette, R., and Gerstner, W. (2005), "Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity," *Journal of Neurophysiology*, 94, 3637–3642. [55]

Chiang, W.-H., Liu, X., and Mohler, G. (2021), "Hawkes Process Modeling of Covid-19 with Mobility Leading Indicators and Spatial Covariates," *International Journal of Forecasting*, 38, 505–520. [45]

Daley, D. J., and Vere-Jones, D. (2003), *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods*, New York: Springer. [46]

Eliasmith, C., and Anderson, C. H. (2003), *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*, Cambridge, MA: MIT Press. [44]

Goense, J., and Ratnam, R. (2003), "Continuous Detection of Weak Sensory Signals in Afferent Spike Trains: The Role of Anti-correlated Interspike Intervals in Detection Performance," *Journal of Comparative Physiology A*, 189, 741–759. [53,54]

Hawkes, A. G. (1971), "Spectra of Some Self-exciting and Mutually Exciting Point Processes," *Biometrika*, 58, 83–90. [44,46]

——— (2018), "Hawkes Processes and their Applications to Finance: A Review," *Quantitative Finance*, 18, 193–198. [44]

Lambert, R. C., Tuleau-Malot, C., Bessaih, T., Rivoirard, V., Bouret, Y., Leresche, N., and Reynaud-Bouret, P. (2018), 'Reconstructing the Functional Connectivity of Multiple Spike Trains Using Hawkes Models," *Journal of Neuroscience Methods*, 297, 9–21. [44,52]

Li, S., Xie, Y., Farajtabar, M., Verma, A., and Song, L. (2017), "Detecting Changes in Dynamic Events Over Networks," *IEEE Transactions on Signal and Information Processing over Networks*, 3, 346–359. [45,50,52]

Lorden, G. (1971), "Procedures for Reacting to a Change in Distribution," *Annals of Mathematical Statistics*, 42, 1897–1908. [49]

Mohler, G. O., Short, M. B., Brantingham, P. J., Schoenberg, F. P., and Tita, G. E. (2011), "Self-exciting Point Process Modeling of Crime," *Journal of the American Statistical Association*, 106, 100–108. [44,45]

Ogata, Y. (1978), "The Asymptotic Behaviour of Maximum Likelihood Estimators for Stationary Point Processes," *Annals of the Institute of Statistical Mathematics*, 30, 243–261. [49]

——— (1988), "Statistical Models for Earthquake Occurrences and Residual Analysis for Point Processes," *Journal of the American Statistical Association*, 83, 9–27. [44]

——— (1998), "Space-time Point-process Models for Earthquake Occurrences," *Annals of the Institute of Statistical Mathematics*, 50, 379–402. [44]

Page, E. S. (1954), "Continuous Inspection Schemes," *Biometrika*, 41, 100–115. [46,47]

Pollak, M. (1985), "Optimal Detection of a Change in Distribution," *Annals of Statistics*, 13, 206–227. [49]

Rambaldi, M., Filimonov, V., and Lillo, F. (2018), "Detection of Intensity Bursts Using Hawkes Processes: An Application to High-Frequency Financial Data," *Physical Review E*, 97, 032318. [45]

Rasmussen, J. G. (2011), "Temporal Point Processes: The Conditional Intensity Function," *Lecture Notes*, Jan. [45]

Reinhart, A. (2018), "A Review of Self-exciting Spatio-Temporal Point Processes and their Applications," *Statistical Science*, 33, 299–318. [45]

Reynaud-Bouret, P., Rivoirard, V., and Tuleau-Malot, C. (2013), "Inference of Functional Connectivity in Neurosciences via Hawkes Processes," in *IEEE Global Conference on Signal and Information Processing*, pp. 317–320. IEEE. [44]

Reynaud-Bouret, P., and Roy, E. (2007), "Some Non asymptotic Tail Estimates for Hawkes Processes," *Bulletin of the Belgian Mathematical Society-Simon Stevin*, 13, 883–896. [44]

Rizoiu, M.-A., Mishra, S., Kong, Q., Carman, M., and Xie, L. (2018), "SIR-Hawkes: Linking Epidemic Models and Hawkes Processes to Model Diffusions in Finite Populations," in *Proceedings of the 2018 World Wide Web Conference*, pp. 419–428. [44]

Shewhart, W. A. (1925), "The Application of Statistics as an Aid in Maintaining Quality of a Manufactured Product," *Journal of the American Statistical Association*, 20, 546–548. [51]

Shewhart, W. A. (1931), *Economic Control of Quality of Manufactured Product*, Milwaukee, WI: American Society for Quality Control. [51]

Tang, S., Zhang, Y., Li, Z., Li, M., Liu, F., Jiang, H., and Lee, T. S. (2018), "Large-Scale Two-Photon Imaging Revealed Super-Sparse Population Codes in the V1 Superficial Layer of Awake Monkeys," *Elife*, 7, e33370. [44]

Tartakovsky, A., Nikiforov, I., and Basseville, M. (2015), *Sequential Analysis: Hypothesis Testing and Changepoint Detection*. ser. Monographs on Statistics and Applied Probability 136. Boca Raton, FL: Chapman and Hall/CRC Press. [47]

Tartakovsky, A. G. (2020), *Sequential Change Detection and Hypothesis Testing: General Non-iid Stochastic Models and Asymptotically Optimal Rules*. ser. Monographs on Statistics and Applied Probability 165. Boca Raton, FL: Chapman and Hall/CRC Press. [47]

Wang, D., Yu, Y., and Willett, R. (2020a), "Detecting Abrupt Changes in High-Dimensional Self-exciting Poisson Processes," arXiv preprint arXiv:2006.03572. [45]

Wang, H., Xie, L., Cuozzo, A., Mak, S., and Xie, Y. (2020b), "Uncertainty Quantification for Inferring Hawkes Networks," *in Advances in Neural Information Processing Systems*, 33. [44,52]

Xie, L., Xie, Y., and Moustakides, G. V. (2020), "Sequential Subspace Change Point Detection," *Sequential Analysis*, 39, 307–335. [52]

Xie, L., Zou, S., Xie, Y., and Veeravalli., V. V. (2021), "Sequential Change Detection: Classical Results and New Directions," *IEEE Journal on Selected Areas in Information Theory*, 2, 494–514. [47]

Xie, Y., and Siegmund, D. (2012), "Spectrum Opportunity Detection with Weak and Correlated Signals," in *Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pp. 128–132. IEEE. [49]

Yang, S.-H., and Zha, H. (2013), Mixture of Mutually Exciting Processes for Viral Diffusion," in *International Conference on Machine Learning*, pp. 1–9. PMLR. [44]

Zhou, F., Li, Z., Fan, X., Wang, Y., Sowmya, A., and Chen, F. (2020), "Fast Multi-Resolution Segmentation for Nonstationary Hawkes Process Using Cumulants," *International Journal of Data Science and Analytics*, 10, 321–330. [45]