Row-clustering of a Point Process-valued Matrix

Anonymous Author(s)
Affiliation
Address
email

Abstract

Structured point process data harvested from various platforms poses new challenges to the machine learning community. By imposing a matrix structure to repeatedly observed marked point processes, we propose a novel mixture model of multi-level marked point processes for identifying potential heterogeneity in the observed data. Specifically, we study a matrix whose entries are marked log-Gaussian Cox processes and cluster rows of such a matrix. An efficient semi-parametric Exponential-Solution (ES) algorithm combined with functional principal component analysis (FPCA) of point processes is proposed for model estimation. The effectiveness of the proposed framework is demonstrated through simulation studies and a real data analysis.

1 Introduction

Large-scale, high-resolution, and irregularly scattered event time data has attracted enormous research interest recently due to its ubiquity in practical applications, including medical visiting records [11], financial transaction ledgers [26] and server logs [9]. Given a collection of event time sequences, one research thread is to identify groups displaying similar patterns. In practice, the significance of this task emerges in multifarious scenarios. For example, matching users with similar activity patterns on social media platforms is beneficial to ads recommendations; clustering patients by their visiting records may help predict the course of the disease progression.

Our study is motivated by a dataset we collected from Twitter, which consists of posting times of 500 university's official accounts during April 15, 2021 to May 14th, 2021. Figure 1 displays posting time stamps of seven selected accounts in five consecutive days. While the daily posting patterns varied across different accounts, the date on which a posting was made seemed to also play an important role. Specifically, all accounts cascaded a barrage of postings on April 16th while few postings appeared on April 18th. Lastly, each posting was associated with a specific type of activity, namely, tweet, retweet, or reply. Our main interest is to cluster these multi-category, dynamic posting patterns into subgroups.

To characterize the highly complex posting patterns, we propose a mixture model of Multi-level Marked Point Processes (MM-MPP). We assume that the event sequences from each cluster are realizations of a multi-level log-Gaussian Cox process (LGCP) [13], which has been demonstrated useful for modelling repeatedly observed event sequences [26] . We here extend their work to the case of mixture models and propose a semiparametric Expectation-Solution algorithm to learn the underlying cluster structure. The proposed learning algorithm avoids iterative numerical optimizations within each ES step and hence is computationally efficient. In addition, we design an algorithm that can take advantage of array programming and GPU acceleration to further speed up computation.

In summary, our main contributions are two folds: (1) we propose an MM-MPP model for repeatedly observed multi-category event sequences; and (2) we develop a highly efficient semiparametric ES algorithm for event sequence clustering.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

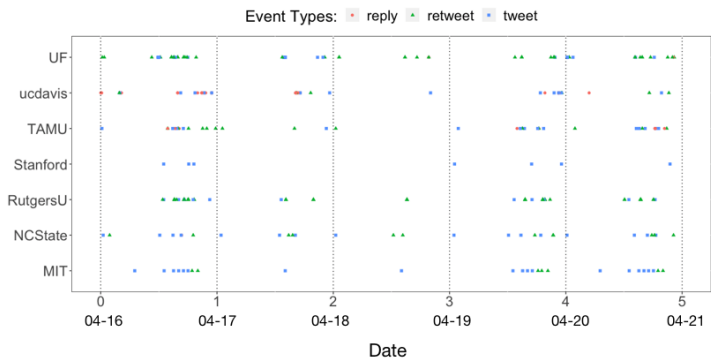

Figure 1: The activities on Twitter of selected accounts.

2 Related Work

Modelling of Event Sequences Point processes have been widely used to model event sequences [5], although most existing work rely on strong parametric assumptions. One prominent example is the Hawkes process [7], which accounts for temporal dependence among events by a self-triggering mechanism. However, existing Hawkes processes often assume that the triggering function only relies on the distance between two time points and hence are not suitable to model the data in our case that have multi-level variations. One way to account for variations from multiple sources is to exploit Cox process models, whose intensities are modeled by latent random functions. One popular class of Cox processes is the log-Gaussian Cox process (LGCP) [13], whose latent intensity functions are assumed to be Gaussian processes. Recently, [26] proposed a multi-level LGCP model to account for different sources of variations for repeatedly observed event data. However, clustering of repeatedly observed marked event time data was not considered in their work.

Clustering of Event Sequences. Extensive research has been done on this topic. To our knowledge, clustering models for point processes can be summarized into two major categories: distance-based clustering [3, 4, 15] and distribution-based clustering [27, 12]. The former measures the closeness between event sequences based on some extracted features and then uses classical distanced-based clustering algorithms such as k-means [4, 16] or EM algorithms [25]. The second approach, also referred to as model-based clustering, assumes that event sequences are derived from a parametric mixture model of point processes. One notable thread is the mixture model of the Hawkes point processes. For example, [27] proposed a Dirichlet mixtures of Hawkes processes (DMHP) under the Expectation-Maximization (EM) framework to identify clusters. However, existing EM algorithms for event sequence clustering have a common issue that they typically require iterative numerical optimizations within each M-step, which would drastically overburden the computation. This computational issue will be accentuated when event data are repeatedly observed and have marks.

3 Model-based Row-clustering for a Matrix of Marked Point Processes

Notation. Mathematically, suppose that we observe daily event sequences from n accounts during m days. For account i on day j, let $N_{i,j}$ denote the total number of events, $t_{i,j,l} \in (0, T]$ denote the l-th event time stamp, and $r_{i,j,l} \in \{1, \cdots, R\}$ denote the corresponding event types (marks). The activities of account i on day j can be summarized by a set $S_{i,j} = \{(t_{i,j,l}, r_{i,j,l})\}_{l=1}^{N_{i,j}}$, recording the time stamps and types for all $N_{i,j}$ events. This general notation can also describe other marked event sequences which are repeatedly observed on m non-overlapping time slots. We represent the collection of all marked daily event sequences as an $n \times m$ matrix $\mathcal{S}$, whose (i, j)th entry is a marked event sequence $S_{i,j}$. We aim to cluster the rows of $\mathcal{S}$ to identify potential heterogeneity in account activity patterns, while taking into account the dependence across rows and columns to characterize the complex event patterns and interactions among accounts, days, and event types.

3.1 A Mixture of Multi-level Marked LGCP Model

Given a matrix of daily event sequences $\mathcal{S}$, we can separate each matrix entry $S_{i,j}$ according to their marks (event types). Let $S^r_{i,j} = \{t_{i,j,l} | r_{i,j,l} = r\}$ record the time stamps of event type $r \in \{1, \cdots, R\}$. We model each $S^r_{i,j}$ by an inhomogeneous Poisson point process conditional on a latent intensity function: $\lambda^r_{i,j}(t|\Lambda^r_{i,j}) = \exp\{\Lambda^r_{i,j}(t)\}$, where $\Lambda^r_{i,j}(t) : [0,T] \mapsto \mathbb{R}$ is the random log intensity function on $[0,T]$. Following [26], we assume a multi-level model for $\Lambda^r_{i,j}(t)$:

$$\Lambda^r_{i,j}(t) = X^r_i(t) + Y^r_j(t) + Z^r_{i,j}(t), \quad t \in [0,T], \tag{1}$$

for $i = 1, \cdots, n$, $j = 1, \cdots, m$ and $r = 1, \cdots, R$. In model (1), $X^r_i(t)$, $Y^r_j(t)$ and $Z^r_{i,j}(t)$ are random functions on $[0,T]$, characterizing the variations of account i, day j and the residual deviation, respectively. In addition, we also take into account the dependence across event types when modelling $X^r_i(t)$, $Y^r_j(t)$ and $Z^r_{i,j}(t)$, while assuming independence across accounts, that is, for any (r, r'), $X^r_i(t)$ and $X^{r'}_{i'}(t)$ are independent when $i \neq i'$, $Y^r_j(t)$ and $Y^{r'}_{j'}(t)$ are independent when $j \neq j'$, and $Z^r_{i,j}(t)$ and $Z^{r'}_{i',j'}(t)$ are independent if $(i,j) \neq (i',j')$.

We assume that $\mathbf{X}_i(t) = \{X^r_i(t)\}^R_{r=1}$ is a mixture of multivariate Gaussian processes with C components in order to detect heterogeneous clusters. We introduce a binary vector $\boldsymbol{\omega}_i = \{\omega_{1,i}, \cdots, \omega_{C,i}\}'$ to encode the cluster membership for account i, where $\omega_{c,i} = 1$ if account i belongs to the c-th cluster and 0 otherwise. In analogy to other model-based clustering approaches, the unobserved cluster membership $\boldsymbol{\omega}_i$ are treated as missing data and assumed to follow a categorical distribution with parameter $\boldsymbol{\pi} = \{\pi_1, \cdots, \pi_C\}$, where π_c indicates the probability that an account belongs to the c-th cluster. Conditional on $\boldsymbol{\pi}$, we assume that $X^r_i(t)$'s in different clusters have heterogeneous behavioral patterns, characterized by their corresponding cluster-specific multivariate Gaussian processes with mean functions $\mu^r_{x,c}(t) = \mathbb{E}[X^r_i(t)|\omega_{c,i} = 1]$ and cross covariance functions $\Gamma^{r,r'}_{x,c}(s,t) = \text{Cov}[X^r_i(s), X^{r'}_i(t)|\omega_{c,i} = 1]$, for $s,t \in [0,T]$, and $r, r' = 1, \cdots, R$. Here, $\mu^r_{x,c}(t)$ characterizes the cluster-specific first-order intensity function, and $\Gamma^{r,r'}_{x,c}(s,t)$ describes the temporal dependence patterns within and across event types.

Similarly, we assume $\mathbf{Y}_j(t) = \{Y^r_j(t)\}^R_{r=1}$ and $\mathbf{Z}_{i,j}(t) = \{Z_{i,j}(t)^r\}^R_{r=1}$ are both mean-zero multivariate Gaussian processes to account for dependence of day-level and residual random effects within and across event types, respectively. The covariance functions take the forms: $\Gamma^{r,r'}_y(t) = \text{Cov}[Y^r_j(t), Y^{r'}_j(t)]$, and $\Gamma^{r,r'}_z(t) = \text{Cov}[Z^r_{i,j}(t), Z^{r'}_{i,j}(t)]$. As the heterogeneity patterns are assumed to be mainly explained by the account-level effect $\mathbf{X}$, both $\Gamma^{r,r'}_y(t)$ and $\Gamma^{r,r'}_z(t)$ are assumed to be homogeneous across all clusters.

A Single-level Special Case. When $m = 1$, our data matrix $\mathcal{S}$ only has one column of event sequences. The multi-level model in (1) reduces to a single-level model:

$$\lambda^r_{i,1}(t|\Lambda^r_{i,1}) = \exp\{\Lambda^r_{i,1}(t)\}, \quad \Lambda^r_{i,1} = X^r_i(t), \quad t \in [0,T] \tag{2}$$

where $\mathbf{X}_i(t) = \{X^r_i(t)\}^R_{r=1}$ has the same model specification as in the multi-level case described earlier. We remark that it is still of importance to consider this special case, as even in this simpler case limited work has been done for the clustering of repeatedly observed marked point processes.

3.2 The Likelihood Function

We denote the parameters concerning $\mathbf{X}_i(t)$ in cluster c as $\Theta_{x,c}$ and the parameters concerning $Y_j(t)$ and $Z_{i,j}(t)$ as Θ_y and Θ_z, respectively. Therefore, the parameters in model (1) consist of $\Omega = \{\boldsymbol{\pi}, \Theta_y, \Theta_z, \Theta_{x,c}, c = 1, \cdots, C\}$. When $m = 1$, $\Omega = \{\boldsymbol{\pi}, \Theta_{x,c}, c = 1, \cdots, C\}$ representing the parameters involved in model (2). The complete data $\mathcal{D}$ consists of the observed data $\mathcal{S}$ and the unobserved latent variables $\big\{\{\boldsymbol{\omega}_i\}^n_{i=1}, \mathcal{L}\big\}$, where $\mathcal{L} = \big\{\{\mathbf{X}_i(t)\}, \{\mathbf{Y}_i(t)\}, \{\mathbf{Z}_{i,j}(t)\}\big\}$ for model (1) and $\mathcal{L} = \{\{\mathbf{X}_i(t)\}$ for model (2). Let S_i be the i-th row of $\mathcal{S}$ representing activities of the i-th account. In our mixture model, the probability of the observed data $\mathcal{S}$ can be written as

$$p(\mathcal{S}; \Omega) = \mathbb{E}_\omega \mathbb{E}_\mathcal{L} \left[\prod_{i=1}^n \text{PP}(S_i|\mathcal{L}) \mid \{\boldsymbol{\omega}_i\}^n_{i=1}; \Omega \right], \tag{3}$$

where the expectations are taken with respect to the conditional distribution of latent variables $\mathcal{L}$ and $\boldsymbol{\omega}_i$'s, and $\text{PP}(S_i|\mathcal{L})$ is the conditional probability of a Poisson point process,

$$\text{PP}(S_i \mid \mathcal{L}) = \prod_{j=1}^{m}\prod_{r=1}^{R}\left\{\prod_{t\in S_{i,j}^r}\lambda_{i,j}^r(t \mid \Lambda_{i,j}^r)\exp\left[-\int_0^T \lambda_{i,j}^r(s \mid \Lambda_{i,j}^r)ds)\right]\right\}, \tag{4}$$

where, conditional on $\mathcal{L}$, $\Lambda_{i,j}^r(t)$ has the form as (1) for $m > 1$ and as (2) for $m = 1$.

4 Row-clustering Algorithms

Existing mixture model-based clustering methods typically rely on likelihood-based Expectation-Maximization algorithms [1] by treating unobserved latent variables, $\left\{\{\boldsymbol{\omega}_i\}_{i=1}^n, \mathcal{L}\right\}$ in our case, as missing data. However, standard EM algorithms are computationally intractable for the models we consider here. One computation bottleneck is the numerical optimizations involved in M-steps, which require many iterations due to the lack of close-form solutions when updating parameters. Moreover, the computation burden is severely aggravated by the fact that the expectations in E-step (see (3) for an example) involve an intractable multivariate integration.

In Section 4.1, we describe a novel efficient semi-parametric Expectation-Solution algorithm for the single-level model in (2) to bypass the computation challenges described above. We then show in Section 4.2 that the learning task of multi-level models in (1) can be transformed and solved by utilizing an algorithm similar to that of single-level models.

4.1 Learning of Single-level Models

The ES algorithm [6] is a general iterative approach to solving estimating equations involving missing data or latent variables. The algorithm proceeds by first constructing estimating equations based on a complete-data summary statistic, which may arise from a likelihood, a quasi-likelihood or other generalized estimating equations. Similar to the EM algorithm, the ES algorithm then iterates between an expectation (E)-step and a solution (S)-step until convergence to obtain parameter estimates. The detailed steps of a general ES algorithm framework are included in Supplementary S.2. The EM framework is a special case of ES when estimating equations are constructed from full likelihoods and using complete data as the summary statistic.

Due to the lack of closed-form for the likelihood function (3), we opt to design our algorithm under the more flexible and general ES framework for parameter estimations of the single-level models in (2), i.e., $m = 1$. The algorithm is summarized in Algorithm 1 and detailed below.

As a preliminary, we give the form of the expectation of the conditional intensity function given cluster memberships as follows:

$$\rho_c^r(t) = \mathbb{E}[\lambda_{i,1}^r(t) \mid \omega_{c,i} = 1] = \exp[\mu_{x,c}^r(t) + \Gamma_{x,c}^r(t,t)/2]. \tag{5}$$

The form of the second-order conditional intensity function is

$$\begin{aligned}\rho_{c,i}^{r,r'} &= \mathbb{E}[\lambda_i^r(s)\lambda_i^{r'}(t) \mid \omega_{c,i} = 1]\\ &= \exp[\mu_{x,c}^r(s) + \mu_{x,c}^{r'}(t)]\mathbb{E}\{\exp[X_i^r(s) + X_i^{r'}(t)|\omega_{c,i} = 1]\}\\ &= \rho_c^r(s)\rho_c^{r'}(t)\exp[\Gamma_{x,c}^{r,r'}(s,t)]\end{aligned} \tag{6}$$

for $i = 1, \cdots, n, r, r' = 1, \cdots, R$, where the last equality is derived following the moment generating function of a Gaussian random variable.

Estimating Equations. We carefully construct estimating equations of unknown parameters with three considerations in mind: (1) the expectation of the estimating equations over the complete data should be zero; (2) the conditional expectation of the estimating equation can be solved efficiently in the S-step; (3) the estimating equations should be fast to calculate.

Let $K_h(t) = h^{-1}K(t/h)$ be a kernel function with bandwidth h. We define

$$A_c^{r,r'}(s,t;h) = \sum_{i=1}^{n} \omega_{c,i} a_i^{r,r'}(s,t;h), \quad \text{where } a_i^{r,r'}(s,t;h) = \sum_{u\in S_i^r, v\in S_i^{r'}}^{u\neq v} \frac{K_h(s-u)K_h(t-v)}{ng(s;h)g(t;h)};$$

$$B_c^r(t;h) = \sum_{i=1}^{n} \omega_{c,i} b_i^r(t;h), \qquad \text{where } b_i^r(t;h) = \sum_{u\in S_i^r} \frac{K_h(t-u)}{ng(t;h)},$$

for $c = 1, ..., C$, and $r, r' = 1, ..., R$, where $g(x;h) = \int K_h(x-t)dt$. Using the Campbell's Theorem [14] and the moment generating function of the normal distribution, it is straightforward to show that $\mathbb{E}\left[A_c^{r,r'}(s,t;h)|\boldsymbol{\omega}\right] \approx \pi_c \rho_c^r(s)\rho_c^{r'}(t)\exp[\Gamma_{x,c}^{r,r'}(s,t)]$ and that $\mathbb{E}\left[B_c^r(t;h)|\boldsymbol{\omega}\right] \approx \pi_c\rho_c^r(t)$, provided that h is sufficiently small. This motivates us to consider the following estimating equations:

$$\begin{cases} A_c^{r,r'}(s,t;h) - \pi_c\rho_c^r(s)\rho_c^{r'}(t)\exp[\Gamma_{x,c}^{r,r'}(s,t)] = 0 \\ B_c^r(t;h) - \pi_c\rho_c^r(t) = 0 \\ \dfrac{\sum_{i=1}^n \omega_{c,i}}{n} - \pi_c = 0. \end{cases} \tag{7}$$

Expectation (E-step). Given an observed data $\mathcal{S}$ and a current parameter estimate Ω^*, we calculate the conditional expectation of the estimation equations in (7). Note that the three estimating equations are all linear with respect to $\{\omega_{c,i}, c = 1, \cdots, C, i = 1, \cdots, n\}$. Therefore, the conditional expectations of the estimating equations are obtained by replacing $w_{c,i}$ with its conditional expectation $\mathbb{E}_{\boldsymbol{\omega}}[\omega_{c,i}|\mathcal{S};\Omega^*]$, which has the following form:

$$\mathbb{E}_{\boldsymbol{\omega}}[\omega_{c,i}|\mathcal{S};\Omega^*] = \frac{\pi_c^* f(S_i|\omega_{c,i}=1;\Omega^*)}{\sum_{c=1}^C \pi_c^* f(S_i|\omega_{c,i}=1;\Omega^*)}, \tag{8}$$

where $f(S_i|\omega_{c,i}=1;\Omega^*) = \mathbb{E}_{\mathcal{L}}[\text{PP}(S_i|\mathcal{L})|\omega_{c,i}=1;\Omega^*]$. Here $\text{PP}(\cdot)$ is the conditional distribution function of S_i given $\omega_{c,i}$ and Ω^* as defined in (4). We propose to approximate $f(S_i|\omega_{c,i}=1;\Omega^*)$ by its Monte Carlo counterpart,

$$\hat{f}(S_i \mid \omega_{c,i}=1;\Omega^*) \approx \frac{1}{Q}\sum \hat{\text{PP}}(S_i \mid \boldsymbol{X}_c^{(g)}(t)), \tag{9}$$

where Q is the Monte Carlo sample size, $\boldsymbol{X}_c^{(q)}(t)$'s are independent samples from the multivariate Gaussian process with parameters $\Theta_{x,c}^*$, and $\hat{\text{PP}}(\cdot)$ is a numerical quadrature approximation of $\text{PP}(\cdot)$ following [2]:

$$\hat{\text{PP}}(S_i|\boldsymbol{X}(t)) = \exp\left\{\sum_{r=1}^{R}\sum_{u\in\tilde{S}_{i,1}^r} v_u[y_u X^r(u) - \exp X^r(u)]\right\}. \tag{10}$$

In the above, $\tilde{S}_{i,1}^r$ is the union of $S_{i,1}^r$ and a set of regular grid points, v_u is the quadrature weight corresponding to each u and $y_u = v_u^{-1}\Delta_u$, where Δ_u is an indicator of whether u is an observation ($\Delta_u = 1$) or a grid point ($\Delta_u = 0$).

Solution (S-step). In this step, we update the parameters by finding the solutions to the expected estimating equations from the E-step. For $c = 1, \cdots, C$, $r, r' = 1, \cdots, R$ and $r \neq r'$, the solutions take the following closed forms:

$$\pi_c^* = \frac{\sum_{i=1}^n \mathbb{E}[\omega_{c,i}|\mathcal{S};\Omega^*]}{n}, \tag{11}$$

$$\Gamma_{x,c}^{r,r'*} = \log\frac{\hat{\pi}_c\mathbb{E}_{\boldsymbol{\omega}}[A_c^{r,r'}(s,t;h)|\mathcal{S};\Omega^*]}{\mathbb{E}_{\boldsymbol{\omega}}[B_c^r(s;h)|\mathcal{S};\Omega^*]\mathbb{E}_{\boldsymbol{\omega}}[B_c^{r'}(t;h)|\mathcal{S};\Omega^*]}, \tag{12}$$

$$\mu_{x,c}^{r*}(t) = \log\frac{\mathbb{E}_{\boldsymbol{\omega}}[B_c^r(t;h)|\mathcal{S};\Omega^*]}{\hat{\pi}_c\exp[\hat{\Gamma}_{x,c}^r(t,t)/2]}. \tag{13}$$

Algorithm 1 Learning of the Single-level model in (2)

Input: $\mathcal{S} = \{S_i\}_{i=1}^n$, the number of clusters C, the bandwidth h;
Output: Estimates of model parameters, $\hat{\boldsymbol{\pi}}$, $\hat{\mu}_{x,c}^r(t)$, $\hat{\Gamma}_{x,c}^{r,r'}(s,t)$, for $c = 1, \cdots, C$, $r, r' = 1, \cdots, R$;
Calculate the components $a_i^{r,r'}$'s and c_i^r's;
Initialize $\Omega^* = \{\boldsymbol{\pi}^*, \Theta_{x,c}^*, c = 1, \cdots, C\}$ randomly;
Repeat:
E-Step:
Calculate $\mathbb{E}_{\boldsymbol{\omega}}[\omega_{c,i}|\mathcal{S}; \Omega^*]$ as (8);
Calculate $\mathbb{E}_{\boldsymbol{\omega}}[A_c^{r,r'}(s,t)|\mathcal{S}; \Omega^*]$ and $\mathbb{E}_{\boldsymbol{\omega}}[B_c^r(t)|\mathcal{S}; \Omega^*]$;
M-Step:
Update $\boldsymbol{\pi}^*$, $\mu_{x,c}^{r*}(t)$ and $\Gamma_{x,c}^{r,r'*}(s,t)$ according to (11),(12) and (13);
End:
Until: Reach the convergence criteria; $\hat{\boldsymbol{\pi}} = \boldsymbol{\pi}^*$, $\hat{\mu}_{x,c}^r = \mu_{x,c}^{r*}(t)$ and $\hat{\Gamma}_{x,c}^{r,r'}(s,t) = \Gamma_{x,c}^{r,r'*}(s,t)$

Sampling Strategies. The multi-dimensional functional form of $\boldsymbol{X}_c^{(g)}$ renders the sampling procedures in (9) intractable. Given the parameters $\Theta_{x,c}$, one solution is to find a low-rank representation of $\boldsymbol{X}_i$ with the functional principal components analysis (FPCA) [17]. Specifically, we approximate the latent Gaussian process $\boldsymbol{X}_i$ in cluster c nonparametrically using the Karhunen-Lòeve expansion [23] as: $X_i^r(t) = \boldsymbol{\mu}_c + < \boldsymbol{\xi}_i, \boldsymbol{\phi}(t) >$, for $r = 1, \cdots, R$, where $\boldsymbol{\xi}_i$ is a vector of normal random variables, and $\boldsymbol{\phi}(t)$ is a vector of orthogonal eigenfunctions. Using FPCA, we can obtain the sampling of $\boldsymbol{X}_i$ from the sampling of $\boldsymbol{\xi}_i$ indirectly. More detailed sampling procedure via FPCA can be seen in Supplementary S.2.

Model Selection. Our clustering procedures require choosing the proper number of clusters C and bandwidth h. In model-based clustering, one popular method for choosing the number of clusters is based on the Bayes information criterion (BIC) [19]. Since the probability $f(\mathcal{S}|\boldsymbol{\omega})$ is already calculated in each iteration, one can directly use this term for the BIC calculation. The choice of kernel bandwidth h also plays an important role for model stability. A small h may produce unstable clustering results while a large h would dampen the characteristics of each cluster. We use an adaptive h that maximizes the likelihood in each iteration. Specifically, we pre-calculate a series of $a_c^{r,r'}$ and b_c^r for different candidates of h. Then in each iteration, we select the one that gives the maximum likelihood.

Remarks. The most significant advantage of our method is that it avoids expensive iterations inside each E-step and S-step, unlike other EM algorithms for mixture point process models [27]. The elements $a_{i,h}^{r,r'}$ and $c_{i,h}^r$ in the estimating equations can be pre-calculated before E-S iterations to save computations. Moreover, the S-step is fast to execute thanks to the closed-form solutions. We will analyze the overall computation complexity of the learning algorithm in Section 4.3.

4.2 Learning of Multi-level Models

We now consider developing the learning algorithm of multi-level models in (1), assuming we repeatedly observe R types of events from n accounts on m days with $m > 1$. Below, we propose a method to transform the learning task of a multi-level model into a problem that can be solved by a two-step procedure, where the second step is mathematically equivalent to a single-level model and hence can be conveniently solved by a similar algorithm as in Algorithm 1.

For a given account i, consider the aggregated event sequence $\bar{S}_{i\cdot}^r = \cup_{j=1}^m S_{i,j}^r$ for each row of $\mathcal{S}$ and event type r. If we assume a multi-level model for each $S_{i,j}^r$ as in (1), conditional on latent variables $\mathcal{L}$, $\bar{S}_{i\cdot}^r$ is a superposition of m independent Poisson processes and hence can be viewed as a new Poisson process with intensity functional $\lambda_{i\cdot}^r(t|\mathcal{L}) = \sum_{j=1}^m \exp \Lambda_{i,j}^r(t)$. We approximate the distribution of $\bar{S}_{i\cdot}^r$ by a Poisson process with a marginal intensity function,

$$\bar{\lambda}_i^r(t) = \mathbb{E}_{YZ}\{\lambda_{i\cdot}^r(t|\mathcal{L})|X_i(t)\} = m \exp\{\tilde{X}_i^r(t)\} \tag{14}$$

where $\tilde{\boldsymbol{X}}_i = \{\tilde{X}_i^1, \cdots, \tilde{X}_i^R\}$ is a new multivariate mixture Gaussian process with mean function $\tilde{\mu}_{x,c}^r(t) = \mu_{x,c}^r + \Gamma_y^{r,r}(t,t)/2 + \Gamma_z^{r,r}(t,t)/2$ and covariance function $\tilde{\Gamma}_{x,c}^{r,r'}(s,t) = \Gamma_{x,c}^{r,r'}(s,t)$, if account i belongs to cluster c. When m is large, we expect the above approximation is accurate.

Note that the model in (14) for the aggregated event sequence $\bar{S}_{i\cdot}^r$ is inherently reduced to a single-level model. It allows us to separate the inference of the multi-level model in (1) into two steps: (*Step I*) learn the parameters in Θ_y and Θ_z and denote the estimated parameters as $\hat{\Gamma}_y^{r,r'}(s,t)$ and $\hat{\Gamma}_z^{r,r'}(s,t)$; (*Step II*) learn the clusters of the single-level model in (14) and estimate the parameters $\boldsymbol{\pi}$, $\tilde{\mu}_{x,c}^r$ and $\tilde{\Gamma}_{x,c}^{r,r'}$; Afterwards, the parameters involved in $\Theta_{x,c}$ can be obtained by,

$$\hat{\mu}_{x,c}^r(t) = \tilde{\mu}_{x,c}^r(t) - \hat{\Gamma}_y^{r,r}(t,t)/2 - \hat{\Gamma}_z^{r,r}(t,t)/2, \quad \hat{\Gamma}_{x,c}^{r,r'}(s,t) = \tilde{\Gamma}_{x,c}^{r,r'}(s,t).$$

For the learning task in Step I, [26] developed a semi-parametric algorithm to learn the repeatedly observed event sequences based on multi-level functional principal component analysis. In analogy to their work, we propose a similar inference framework to estimate Θ_y and Θ_x in our mixture multi-level model (1) and provide the details in Supplementary S.1. For step II, we resort to the single-level model algorithm described in Section 4.1.

4.3 Computational Complexity and Acceleration

Assume that the training event sequences belong to n accounts and C clusters and are repeatedly observed on m time slots. We also assume that the data contains R types of events and each sequence consists of I time stamps on average . Let Q be the sampling size used in the Monte Carlo integration in (9). In Step I, it requires $O(nmR)$ computation complexity to estimate Θ_y and Θ_z, according to [26]. Computation required to pre-calculate $a_i^{r,r'}$ and b_i^r is $O(nmIR^2)$ if we decomposition $a_c^{r,r'}$ as:

$$a_c^{r,r'} = \left[\sum_{u \in S_i^r} \frac{K_h(s-u)}{g(s;h)}\right]\left[\sum_{v \in S_i^{r'}} \frac{K_h(t-v)}{g(t;h)}\right] - \sum_{u \in S_i^r \cap S_i^{r'}} \frac{K_h(s-u)K_h(t-v)}{g(s;h)g(t;h)}.$$

In Step II, for each E-S iteration, we need $O(nCQR^3)$ for sampling and $O(nBCIQR^2)$ for other calculations. Therefore, the overall computational complexity is $O(nR^2(mI + CQR + CGI))$. To further reduce computation, we use array programming and GPU acceleration to calculate the high-dimensional integration in the Monte Carlo EM framework [24] to reduce the runtime of (8). The details are included in Supplementary S.2. In this study, we investigate situation when the value of R is small to moderate. When R is large, the low-rank or sparse properties of $\Gamma_{x,c}^{r,r'}$'s may be utilized to reduce the sampling complexity to $O(nCQR^2)$.

5 Numerical Examples

We examine the performance of our **MM-MPP** framework for clustering event sequences via synthetic data examples and a real-world application, and compare the performance between the proposed method with two other state-of-the-art methods. One competing method is a discrete Fréchet distance-based method (**DF**) by [15]. Unlike other distance-based clustering method, the DF cluster is able to characterize interactions among events. Another is a model-based clustering method based on Dirichlet mixture of Hawkes processes (**DMHP**) by [27]. DMHP is chosen as a competitor due to its capability of accounting for complex point patterns while performing clustering and making efficient variational Bayesian inference algorithm under a nested EM framework.

5.1 Synthetic Data

Setting. We generate the synthetic data from the proposed mixture model of log-Gaussian Cox processes in (1) and (2), in which there are $R = 5$ event types and daily time stamps reside in $[0, 2]$. We set the number of clusters C from 2 to 5 and set the number of accounts in each cluster to 500. We experiment with an increasing number of replicates ($m = 1$, 20 or 100), in order to check the convergence of our method. When $m = 1$, we generate event sequences from the single-level model in (2) without day-level variations. In this case, we compare the clustering results of DF, DMHP with those of the single-level model (MS-MPP). When $m = 20$ or 100, we generate data from the multi-level model in (1) and use the MM-MPP method to model the scenario where event sequences are repeatedly observed. However, the two competing methods, DF and DMHP, are not directly applicable for repeated event sequences. Therefore, in this case, we concatenate $\{S_{i,j}^r\}_{j=1}^m$

Table 1: Clustering Purity on Synthetic Data.

	$m = 1$			$m = 20$			$m = 100$		
C	DF	DMPH	MS-MPP	DF	DMPH	MM-MPP	DF	DMPH	MM-MPP
2	0.597	0.537	0.831	0.536	0.513	0.947	0.532	0.522	0.988
3	0.514	0.466	0.767	0.465	0.423	0.902	0.477	0.394	0.967
4	0.443	0.421	0.714	0.422	0.356	0.874	0.436	0.285	0.944
5	0.379	0.354	0.675	0.351	0.298	0.835	0.333	0.276	0.919

sequentially into a new event sequence $S^r_{i\cdot}$ on $[0, mT]$ and then apply DF and DMGP to this new sequence. The detailed settings of $X^r_i(t)$'s, $Y^r_j(t)$'s and $Z^r_{i,j}(t)$'s and other details of synthetic data examples are elaborated in Supplementary S.3.

Results. We evaluate the average clustering performance of each method over 100 repeated experiments under each setting, using *clustering purity* [18] as a evaluation metric. Table 1 reports the averaged clustering purity of each method on the synthetic data. When $m = 1$, MS-MPP obtains the best clustering result in terms of purity consistently across different numbers of clusters. Especially when C increases, which implies there is more overlap among clusters, the advantage of MS-MPP becomes more prominent. When $m = 20, 100$, MM-MPP still significantly outperforms the other two competitors. It is also noticeable that the performance of DF and DMPH in general deteriorates as m increases, despite the fact that more repeated event sequences offer more information for clustering. One explanation is that both DF and DMHP may incur bias due to ignoring different sources of variations for repeatedly observed event times. Another reason is that many existing Hawkes process models, such as DMHP, assume a constant triggering function over time, which may no be flexible enough to characterize general data generation patterns.

5.2 Real-world Data

we apply our method to a real-world Twitter dataset consisting of the postings of the official accounts of America's top 500 universities from April 15, 2021 to May 14, 2021. The data set was scraped from Twitter with the API `rtweet` [10]. It involves three categories of postings (tweet, retweet and reply), so that $R = 3$ in this study. As a result, the dataset contains $n = 500$ Twitter accounts for $m = 30$ consecutive days with a total of $233,465$ time stamps. We evaluate and compare clustering stability based on a measure called *clustering consistency* via K-trial cross validations [20, 21], as there is no ground truth clustering labels. The detailed definition of *clustering consistency* and other real data example details are included in Supplementary S.4.

Results. We compare the performance of DF, DMHP and MM-MPP in terms of clustering consistency using the Twitter dataset with $K = 100$ trials. The results in Table 2 suggest that MM-MPP outperforms its competitors notably, which demonstrates that our model can better characterize the postings patterns and offer a more stable and consistent clustering than other methods. Figure 2 shows the histograms of the number of learned clusters for each method. The median numbers of learned clusters are 3, 5 and 8 for MM-MPP, DMHP and DF respectively. Besides, the distribution of the number of clusters from our method seems to be the most concentrated, indicating its robustness in clustering. This phenomenon reflects the potential drawbacks of other distance-based clustering methods; they highly rely on the predefined features and hence are less robust. The robustness of our method may be also partly attributed to the flexibility of the latent conditional intensity functions that account for local multi-level deviations within each account. In contrast, other methods that fail to account local deviations may treat them as sources of heterogeneities and consequently result in more clusters.

More stories can be told by the estimated posting patterns. Given a predicted membership of account i by $c_i = \arg\max_c \mathbb{E}_\omega[\omega_{c,i}|\mathcal{S}; \hat{\Omega}]$, Figure 3 displays the estimated curves of $\hat{\mu}^r_{x,c}$ for tweet events ($r = 1$), retweet events ($r = 2$) and reply events ($r = 3$) respectively for $C = 3$. Recall $\hat{\mu}^r_{x,c}$ is interpreted as the baseline of intensity functions. This figure shows three different activity modes for the selected Twitter accounts. The universities in cluster 1 marked by red curves in Figure 3 in general have a lower frequency of posting retweets and reply, especially at daytime. This group includes the most top university in America, such as MIT, Harvard and Stanford. In contrast, the accounts in cluster 2 are relatively more active in all three types of postings. We further find that many accounts

Table 2: Clustering Consistency on Real-world Data.

Method	DF	DMHP	MM-MPP
Consistency	0.0961	0.275	0.394

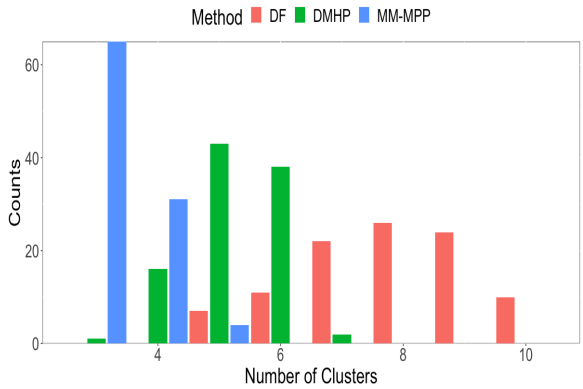

Figure 2: Histogram of the number of clusters.

in this cluster belong to the universities with middle ranks. More results and discussions with real data are included in our Supplementary file.

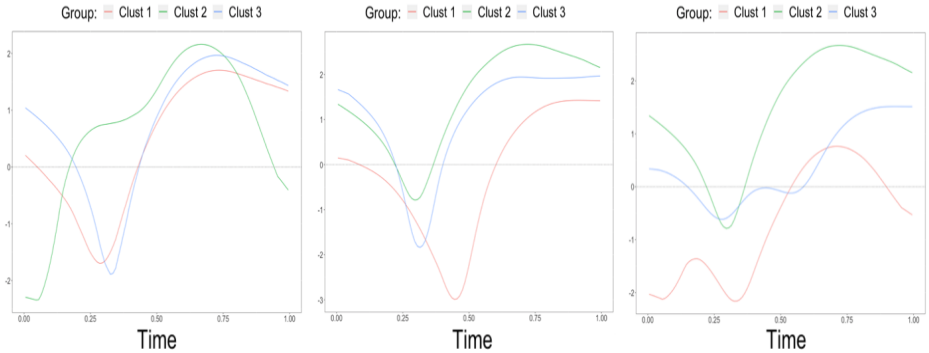

Figure 3: Curves of $\hat{\mu}^r_{x,c}(t)$. Left: tweet events; Mid: retweet events; Right: reply events;

6 Concluding Remarks

We propose a mixture model of multi-level marked point processes for the clustering of repeatedly observed marked event sequences. A novel and efficient learning algorithm is developed based on a semi-parametric ES method. We analyze the computational complexity of our learning algorithm. The proposed method is demonstrated to significantly outperform other competing methods in simulation experiments and in an application to Twitter data.

Currently we only focus on the events on temporal domains. However, clustering of spatial patterns on 2- or 3-dimensional domains has also attracted much research interests [8, 28]. Our current method involves a nonparametric covariance function estimation, which will encounter the curse of dimensionality when data reside in a higher p-dimensional domain. [22] proposed a low-rank covariance function estimation framework for functional data on high-dimensional space, which may serve as a potential remedy to the dimensionality issue.

This work has no foreseeable negative societal impacts, but users should be cautious when giving interpretation on clustering results to avoid any misleading conclusions.

References

[1] Murray Aitkin and Donald B Rubin. Estimation and hypothesis testing in finite mixture models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 47(1):67–75, 1985.

[2] Mark Berman and T Rolf Turner. Approximating point process likelihoods with glim. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 41(1):31–38, 1992.

[3] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994.

[4] Paul S Bradley and Usama M Fayyad. Refining initial points for k-means clustering. In *ICML*, volume 98, pages 91–99. Citeseer, 1998.

[5] Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: volume I: elementary theory and methods*. Springer, 2003.

[6] Michael Elashoff and Louise Ryan. An em algorithm for estimating equations. *Journal of Computational and Graphical Statistics*, 13(1):48–65, 2004.

[7] Alan G Hawkes and David Oakes. A cluster process representation of a self-exciting process. *Journal of Applied Probability*, pages 493–503, 1974.

[8] Anders Hildeman, David Bolin, Jonas Wallin, and Janine B Illian. Level set cox processes. *Spatial statistics*, 28:169–193, 2018.

[9] Husna Sarirah Husin, Lishan Cui, Herny Ramadhani Husny Hamid, and Norhaiza Ya Abdullah. Time series analysis of web server logs for an online newspaper. In *Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication*, pages 1–4, 2013.

[10] Michael W Kearney. rtweet: Collecting and analyzing twitter data. *Journal of Open Source Software*, 4(42):1829, 2019.

[11] Thomas A Lasko. Efficient inference of gaussian-process-modulated renewal processes with application to medical event data. In *Uncertainty in artificial intelligence: proceedings of the... conference. Conference on Uncertainty in Artificial Intelligence*, volume 2014, page 469. NIH Public Access, 2014.

[12] Dixin Luo, Hongteng Xu, Yi Zhen, Xia Ning, Hongyuan Zha, Xiaokang Yang, and Wenjun Zhang. Multi-task multi-dimensional hawkes processes for modeling event sequences. 2015.

[13] Jesper Møller, Anne Randi Syversveen, and Rasmus Plenge Waagepetersen. Log gaussian cox processes. *Scandinavian journal of statistics*, 25(3):451–482, 1998.

[14] Jesper Moller and Rasmus Plenge Waagepetersen. *Statistical inference and simulation for spatial point processes*. CRC Press, 2003.

[15] Tao Pei, Xi Gong, Shih-Lung Shaw, Ting Ma, and Chenghu Zhou. Clustering of temporal event processes. *International Journal of Geographical Information Science*, 27(3):484–510, 2013.

[16] Jie Peng, Hans-Georg Müller, et al. Distance-based clustering of sparsely observed stochastic processes, with applications to online auctions. *Annals of Applied Statistics*, 2(3):1056–1077, 2008.

[17] James O Ramsay and BW Silverman. Functional data analysis. *İnternet Adresi: http*, 2008.

[18] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

[19] Gideon Schwarz et al. Estimating the dimension of a model. *Annals of statistics*, 6(2):461–464, 1978.

[20] Robert Tibshirani and Guenther Walther. Cluster validation by prediction strength. *Journal of Computational and Graphical Statistics*, 14(3):511–528, 2005.

[21] Ulrike Von Luxburg. Clustering stability: an overview. 2010.

[22] Jiayi Wang, Raymond KW Wong, and Xiaoke Zhang. Low-rank covariance function estimation for multidimensional functional data. *Journal of the American Statistical Association*, pages 1–14, 2020.

[23] Satosi Watanabe. Karhunen-loeve expansion and factor analysis: theoretical remarks and application. In *Trans. on 4th Prague Conf. Information Theory, Statistic Decision Functions, and Random Processes Prague*, pages 635–660, 1965.

[24] Hong-Zhong Wu, Jun-Jie Zhang, Long-Gang Pang, and Qun Wang. Zmcintegral: A package for multi-dimensional monte carlo integration on multi-gpus. *Computer Physics Communications*, 248:106962, 2020.

[25] Weichang Wu, Junchi Yan, Xiaokang Yang, and Hongyuan Zha. Discovering temporal patterns for event sequence clustering via policy mixture model. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[26] Ganggang Xu, Ming Wang, Jiangze Bian, Hui Huang, Timothy Burch, Sandro Andrade, Jingfei Zhang, and Yongtao Guan. Semi-parametric learning of structured temporal point processes. *Journal of machine learning research*, 2020.

[27] Hongteng Xu and Hongyuan Zha. A dirichlet mixture model of hawkes processes for event sequence clustering. *arXiv preprint arXiv:1701.09177*, 2017.

[28] Fan Yin, Guanyu Hu, and Weining Shen. Analysis of professional basketball field goal attempts via a bayesian matrix clustering approach. *arXiv preprint arXiv:2010.08495*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code will be made publicly available upon request of revision or acceptance of the manuscript.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Supplementary S.2
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Supplementary S.4
   (b) Did you mention the license of the assets? [Yes] See Supplementary S.4
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]