Deep Neyman-Scott Processes

Chengkuan Hong
UC Riverside
chong009@ucr.edu

Christian R. Shelton
UC Riverside
cshelton@cs.ucr.edu

Abstract

A Neyman-Scott process is a special case of a Cox process. The latent and observable stochastic processes are both Poisson processes. We consider a deep Neyman-Scott process in this paper, for which the building components of a network are all Poisson processes. We develop an efficient posterior sampling via Markov chain Monte Carlo and use it for likelihood-based inference. Our method opens up room for the inference in sophisticated hierarchical point processes. We show in the experiments that more hidden Poisson processes brings better performance for likelihood fitting and events types prediction. We also compare our method with state-of-the-art models for temporal real-world datasets and demonstrate competitive abilities for both data fitting and prediction, using far fewer parameters.

1 INTRODUCTION

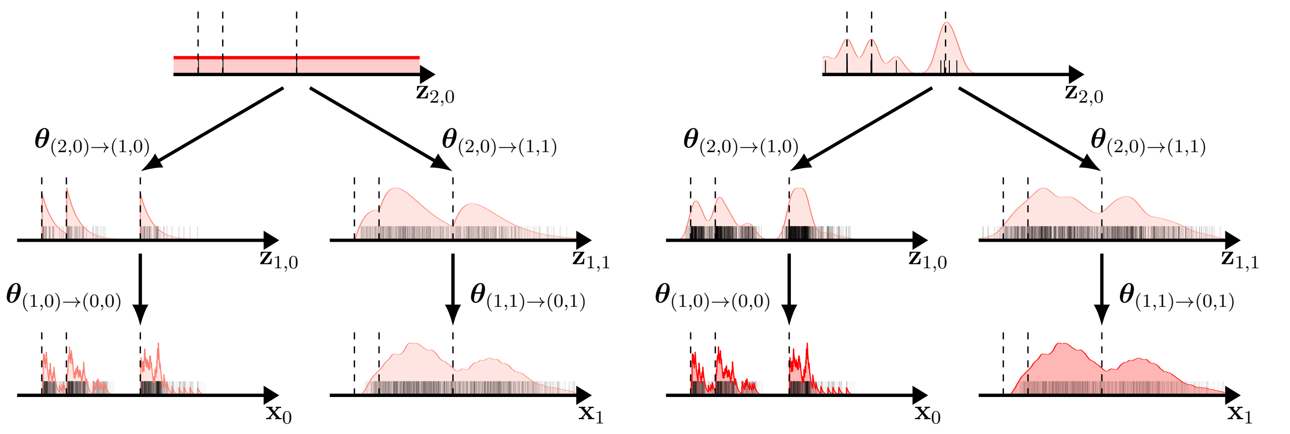
Point processes have attracted attention due to their ability to model the temporal and spatial patterns of event data. They have been applied to various fields, *e.g.*, finance (Bauwens & Hautsch, 2009), neuroscience (Perkel et al., 1967), and cosmology (Stoica et al., 2014). The Poisson process is one of the most commonly used models. The intensity of a Poisson process describes the expected number of events per unit time or space and is independent of the events elsewhere.

We usually do not have any prior knowledge of the functional form of the intensity function. A common strategy is to assume the intensity function itself is a latent stochastic process. By introducing this latent process, we introduce dependence between the expected number of events at different times or places. With this strategy, we have a class of point processes called Cox processes (Cox, 1955). Typical examples for the latent stochastic process are Poisson processes (Neyman & Scott, 1958), Cox processes (Adams, 2009), Strauss processes (Yau & Loh, 2012), and Gaussian processes (Adams et al., 2009).

In this paper, we consider a special class of Cox processes, Neyman-Scott processes (N-SPs) of order larger than one (Neyman & Scott, 1958), which we call deep Neyman-Scott processes (DN-SPs). Neyman and Scott first described univariate DN-SPs in 1958; they were trying to model the distribution of galaxies in the Universe. Each building block of the univariate DN-SPs is a Poisson process. Taking univariate DN-SPs of order two as an example, one Poisson process generates the centers of some superclusters, consisting of galaxy clusters. In turn, each galaxy cluster generates its own set of points, galaxies, from a Poisson process centered the cluster center. The advantage of the DN-SPs compared to the other Cox processes is that they have the ability to build deep hierarchical models where each component in the system is a point process. Further, unlike Hawkes processes (Hawkes, 1971) and similar temporal processes, DN-SP processes do not enforce a temporal or causal "direction," and thus also encompass spatial and spatio-temporal processes.

N-SPs of order of one have been well studied in the literature. Møller & Waagepetersen (2003) discuss posterior sampling algorithms and inference methods for univariate N-SPs based on the Metropolis-Hastings (M-H) (Metropolis et al., 1953; Norman & Filinov, 1969; Hastings, 1970) and spatial birth-and-death (SB&D) (Kelly & Ripley, 1976; Ripley, 1977; Baddeley & Møller, 1989). Linderman et al. (2017) consider a sequential Monte Carlo sampler for a multivariate (not deep) N-SP, but without giving a functional form for the proposal. Williams et al. (2020) give a collapsed Gibbs posterior sampling algorithm for a univariate (not deep) N-SP. However, little attention has been given to DN-SPs (order greater than one) due to the difficulty in the

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s).



$\mathbf{z}_{2,0}$
$\boldsymbol{\theta}_{(2,0)\to(1,0)}$
$\boldsymbol{\theta}_{(2,0)\to(1,1)}$
$\mathbf{z}_{1,0}$
$\mathbf{z}_{1,1}$
$\boldsymbol{\theta}_{(1,0)\to(0,0)}$
$\boldsymbol{\theta}_{(1,1)\to(0,1)}$
$\mathbf{x}_0$
$\mathbf{x}_1$

Figure 1: Illustration of Model Sampling: (left) forward sampling & (right) posterior sampling.

complex posterior sampling and inference. The only existing work we have found is from Andersen et al. (2018), which employs a moment-based inference for photoactivated localization microscopy. Although the moment-based inference is computationally easy, some constant parameters for the estimation of the moments have to be tuned manually, and it is not able to estimate the posterior intensity surface of the unobservable point processes.

To the best of our knowledge, we are the first to give a posterior sampling algorithm and a likelihood-based inference method for multivariate DN-SPs with experiments for real-world data. In this paper, we first construct a temporal multivariate DN-SP in one dimension in Sec. 2. Then in Sec. 3, we present an efficient posterior sampling via Markov chain Monte Carlo (MCMC) and use it for inference. We give competitive experiments results for real-world data compared with MTPP (Lian et al., 2015), the neural Hawkes process (Mei & Eisner, 2017), the self-attentive Hawkes process (Zhang et al., 2020a), and the transformer Hawkes process (Zuo et al., 2020) in Sec. 4. While our experiments are for temporal point processes, our inference method can also be applied to general spatial point processes as discussed in Sec. 3.1.

2 DEEP NEYMAN-SCOTT PROCESSES

In this section, we review temporal point processes and introduce our DN-SPs. Different from the N-SPs given by Neyman & Scott (1958), our DN-SPs are multivariate (events can be marked or labeled).

Temporal Point Processes A temporal point process (TPP) (Daley & Vere-Jones, 2003) is a random process with a realization as a sequence of events $\{t_i\}_{i=1}^m$, where each time point $t_i \in \mathbb{R}_{\geq 0}$ and $t_i < t_{i+1}, \forall i$.

Definition 1 (conditional intensity function). *The* conditional intensity function *(CIF) of a TPP is defined as*

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{\Pr(\text{One event in } [t, t+\Delta t] \mid \mathcal{H}_t)}{\Delta t},$$

where $\mathcal{H}_t$ is an element in a filtration $\{\mathcal{H}_t\}_{t\geq 0}$.

TPPs are completely characterized by their CIFs as they determine the distributions of the numbers of events as Bernoulli random variables at each infinitesimal interval. As a special case for TPPs, a homogeneous Poisson process (HPP) has a constant CIF $\lambda_0 > 0$. At each infinitesimal interval $[t, t+\Delta t]$, the probability of one event is $\lambda_0 \cdot \Delta t$.

Deep Neyman-Scott Processes To build DN-SPs, we will stack TPPs in a hierarchical manner *s.t.* the distributions of the CIFs as random processes are controlled by the TPPs on higher layers.

Each observed data point $\mathbf{x}$ is a collection of sequences $\{\mathbf{x}_k\}_{k=1}^{K_0} = \{\{t_{0,k,j}\}_{j=1}^{m_{0,k}}\}_{k=1}^{K_0}$, where $\mathbf{x}_k$ is the sequence of the kth type of event (or events with the kth mark in a discrete-marked TPP) and $t_{0,k,j}$ is the time of the jth event of this type. (The 0 indicates this is an observation event at the bottom layer of the model.) For each data point $\mathbf{x}$, there are L hidden layers of TPPs $\mathbf{Z} = \{\mathbf{Z}_1, \ldots, \mathbf{Z}_L\}$, with $\mathbf{Z}_\ell = \{Z_{\ell,k}\}_{k=1}^{K_\ell}$. K_ℓ is the number of hidden processes at level ℓ, and $Z_{\ell,k}$ is a TPP. $Z_{\ell,k}$ and $Z_{\ell+1,j}$ are connected by a kernel function $\phi_{\boldsymbol{\theta}_{(\ell+1,k)\to(\ell,j)}}(\cdot)$, whose functional form will be given later.

Generative Model Semantics We first draw samples from the TPPs on the top layer. These TPPs are assumed to be HPPs. The CIF for $Z_{L,k}$ is a constant function

$$\lambda_{L,k}(t) = \mu_k, \text{ where } \mu_k > 0.$$

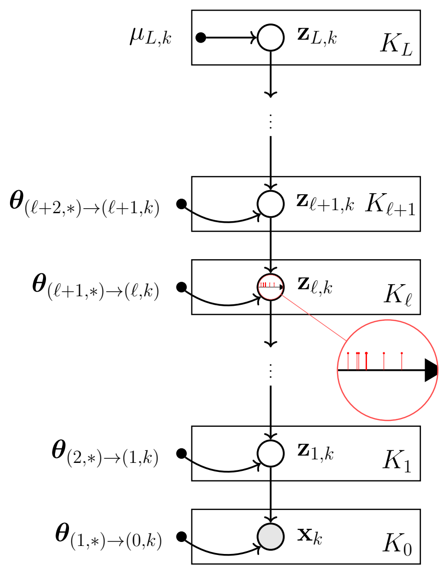

$\mu_{L,k}$
$\mathbf{z}_{L,k}$
K_L
$\boldsymbol{\theta}_{(\ell+2,*)\to(\ell+1,k)}$
$\mathbf{z}_{\ell+1,k}$
$K_{\ell+1}$
$\boldsymbol{\theta}_{(\ell+1,*)\to(\ell,k)}$
$\mathbf{z}_{\ell,k}$
K_ℓ
$\boldsymbol{\theta}_{(2,*)\to(1,k)}$
$\mathbf{z}_{1,k}$
K_1
$\boldsymbol{\theta}_{(1,*)\to(0,k)}$
$\mathbf{x}_k$
K_0

Figure 2: The Structure of A DN-SP

Next, we draw samples for each hidden TPPs conditional on the TPPs in the layer immediately above. The CIF for $Z_{\ell,k}$ conditional on $\mathbf{Z}_{\ell+1}$ is

$$\lambda_{\ell,k}(t) = \sum_{i=1}^{K_{\ell+1}} \sum_{t_{\ell+1,i,j}} \phi_{\boldsymbol{\theta}_{(\ell+1,i)\to(\ell,k)}}(t - t_{\ell+1,i,j}), \quad (1)$$

where $\mathbf{z}_{\ell+1,i} = \{t_{\ell+1,i,j}\}_{j=1}^{m_{\ell+1,i}}$ is a realization for $Z_{\ell+1,i}$.

As an example, Fig. 1 demonstrates the distributions of the events for a DN-SP with two hidden layers. The left figure is for forward sampling and the right one is for posterior sampling. For forward sampling, three events are drawn from $Z_{2,0}$. The dashed lines indicate the positions of the three events on the top layer. The plots for the other TPPs are the densities and rug plots of the samples drawn conditioned on $\mathbf{z}_{2,0}$. For posterior sampling, the samples for $\mathbf{x}_0$ and $\mathbf{x}_1$ are collected and fixed from the forward sampling. Then we draw posterior samples for the other TPPs conditional on $\mathbf{x}_0$ and $\mathbf{x}_1$ and plot their densities and rug plots as well. Note the modes of the posterior distribution for $\mathbf{z}_{2,0}$ recover the positions of the prior events for $\mathbf{z}_{2,0}$ in forward sampling.

Fig. 2 gives the general structure of a DN-SP similar to a graphical model, where each node represents a TPP, $\boldsymbol{\theta}_{(\ell+1,*)\to(\ell,k)} = [\boldsymbol{\theta}_{(\ell+1,i)\to(\ell,k)}]_{i=1}^{K_{\ell+1}}$, and a sample node for $\mathbf{z}_{\ell,k}$ is enlarged.

Conditioned on $\mathbf{Z}_{\ell+1}$, $Z_{\ell,k}$ is a Poisson process whose CIF at each time interval is independent of the CIF at the other time intervals, and the number of events at each time interval follows the Poisson distribution. The time interval for each hidden TPP is set to be the same as the evidence and no edge effects are considered in our model.

Evidence Likelihood The observed data are assumed to be drawn conditioned on the hidden TPPs on the lowest hidden layer in a manner analogous to those of the layers above. That is, the intensity for $\mathbf{x}_k$, $\lambda_{0,k}(t)$, is as in Eq. 1 where $\ell = 0$.

The Kernel Function For a TPP, it is reasonable to make the kernel function be zero for negative inputs (*i.e.*, to be "causal") and converge to 0 as t goes to infinity (the influence of the events will eventually disappear). The sole constrain is that the sum appearing in the intensity function (Eq. 1) should always be nonnegative. We also want the kernel function to be as flexible as possible with only a few parameters, so we choose a gamma kernel such that

$$\phi_{\boldsymbol{\theta}}(x) = \begin{cases} p \cdot \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, \text{ for } x > 0,\ p, \alpha, \beta > 0, \\ 0, \text{ for } x \le 0, \end{cases}$$

where $\boldsymbol{\theta} = \{p, \alpha, \beta\}$ and $\Gamma(\alpha)$ is the gamma function. The gamma kernel asymptotes to 0 as $x \to \infty$. Moreover, with the varying parameters, the gamma kernel can either be monotonically decreasing or have a unimodal shape, which provides flexibility.

3 INFERENCE

Typical ways to estimate the parameters include Bayesian inference and maximum likelihood estimation (MLE). Bayesian inference requires the posterior sampling for the parameters (see Appx. D for more details), but in our case it becomes too slow to be feasible if we have many parameters. For more efficient inference, we adopt Monte Carlo expectation-maximization (MCEM) (Wei & Tanner, 1990) as an indirect way to maximize the marginal likelihood. Different from the ordinary EM, the expected value of the log-likelihood is approximated by a Monte Carlo method in the expectation steps. Posterior samples of the hidden TPPs are required for the estimation of the expected values.

The posterior sampling for a hidden TPP is not trivial as it involves the sampling for an unbounded number of variables and the posterior stochastic processes of TPPs are not TPPs anymore. Instead, the posterior stochastic processes of TPPs are spatial point processes (SPPs). Unlike TPPs, SPPs do not have a natural ordering in space to define a filtering, hence the CIFs are not well-defined. In order to better describe a SPP, we need to provide a different type of conditional intensity function for an SPP:

Definition 2 (Papangelou conditional intensity function (Papangelou, 1974))**.** *The* Papangelou conditional intensity function *(PCIF) is defined as*

$$\lambda_{\mathcal{P}}(t) = \lim_{\Delta t \to 0} \frac{\Pr(\text{One event in } B_{\Delta t}(t) \mid [\mathbf{N} \backslash B_{\Delta t}(t)])}{\Delta t},$$

where $B_{\Delta t}(t) = [t, t+\Delta t]$, $\mathbf{N} = [\mathbf{Z}, \mathbf{x}]$ is the whole set of hidden TPPs and the evidence, and $[\mathbf{N} \backslash B_{\Delta t}(t)]$ is the information of the point processes $\mathbf{N}$ outside $B_{\Delta t}(t)$.

Different from a CIF, which is only conditional on the history, a PCIF is conditional on the whole space. For a Poisson process, PCIF is equivalent to CIF as the probability to have an event at any infinitesimal interval is independent of the time or location.

The PCIF for our model is given in Prop. 1. See Appx. A for more details.

Proposition 1. *The PCIF for the posterior point process of $Z_{\ell,k}$ is*

$$\lambda_{\mathcal{P};\ell,k}(t) = \lambda_{\ell,k}(t) \prod_{i=1}^{K_{\ell-1}} \left(\exp\left(-\Phi_{(\ell,k)\to(\ell-1,i)}(T-t)\right) \prod_{t_{\ell-1,i,j}>t} \frac{\lambda'_{\ell-1,i}(t_{\ell-1,i,j}, t)}{\lambda_{\ell-1,i}(t_{\ell-1,i,j})} \right), \quad (2)$$

where $\lambda'_{\ell-1,i}(x,t) = \lambda_{\ell-1,i}(x) + \phi_{(\ell,k)\to(\ell-1,i)}(x-t)$ and $\Phi_{(\ell,k)\to(\ell-1,i)}(x) = \int_0^x \phi_{\boldsymbol{\theta}_{(\ell,k)\to(\ell-1,i)}}(\tau)d\tau$.

From Prop. 1, we can see that the PCIF at time t is not only controlled by the events on layers $\ell+1$ through $\lambda_{\ell,k}(\cdot)$, but controlled by the events on layers ℓ and $\ell-1$ through $\{\lambda_{\ell-1,i}(\cdot)\}_{i=1}^{K_{\ell-1}}$.

3.1 MCMC

As the PCIF has complex correlations both within and between nodes, it is hard to directly do posterior sampling for the hidden TPPs. Considering the simplest case, if we only have one hidden layer, our model just becomes a multivariate N-SP. In the SPPs community, SB&D is more popular than M-H possibly due to its simplicity and efficiency (Geyer & Møller, 1994; Clifford & Nicholls, 1994; Møller & Waagepetersen, 2003). Unfortunately, the sufficient condition for the convergence of SB&D is not satisfied in our case, and it is an open question of whether SB&D converges for our model. We provide more details in Appx. E. Thus, we devised a different Markov chain equipped with auxiliary variables that converges quickly to the true posterior distribution. Compared with a naive MCMC sampler which just re-samples all the hidden events from homogeneous Poisson processes every time as the proposal, our posterior sampling is much more efficient due to the help of auxiliary variables.

Remark. *Our MCMC can be applied to non-casual kernels (i.e., the kernels can be functions which have non-zero values in the whole space, like a Gaussian function) and SPPs with any dimensions, not only temporal DN-SPs. No matter how we choose the kernel, the detailed balance and ergodicity conditions are still satisfied, and thus the MCMC sampler still converges to the posterior distribution.*

3.1.1 Virtual Events

Similar to prior work (Rao & Teh, 2011a, 2013; Qin & Shelton, 2015; Shelton et al., 2018), we add virtual events as auxiliary variables for our MCMC sampler. They work by providing the candidates for the real events of the hidden TPPs, and they do not contribute to any intensity functions. With the help of the virtual events, we only need to search for the real events where the virtual events appear, instead of the whole space. We explore all possible real event locations by resampling virtual events.

For each data point $\mathbf{x}$, there are L layers of virtual TPPs (VPPs) $\tilde{\mathbf{Z}} = \{\tilde{\mathbf{Z}}_1, \ldots, \tilde{\mathbf{Z}}_L\}$ aligning with the hidden real point processes (RPPs) $\mathbf{Z}$, where $\tilde{\mathbf{Z}}_\ell = \{\tilde{Z}_{\ell,k}\}_{k=1}^{K_\ell}$ with $\tilde{Z}_{\ell,k}$ as a virtual TPP. The CIF for $\tilde{Z}_{\ell,k}$ conditioned on $\mathbf{Z}_{\ell-1}$ is

$$\tilde{\lambda}_{\ell,k}(\tilde{t}) = \tilde{\mu}_{\ell,k} + \sum_{i=1}^{K_{\ell-1}} \sum_{t_{\ell-1,i,j}} \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(t_{\ell-1,i,j} - \tilde{t}) \quad (3)$$

where $\tilde{\mu}_{\ell,k} \geq 0$ is the base rate, $\tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(\cdot)$ is the virtual kernel function, which we assume is a gamma kernel. Note that $\tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(t_{\ell-1,i,j} - \tilde{t})$ evolves in the **opposite direction** to $\phi_{\boldsymbol{\theta}_{(\ell+1,i)\to(\ell,k)}}(t - t_{\ell+1,i,j})$, which we will rationalize later. However, the intensity of virtual events at layer ℓ depends on the **real events** at layer $\ell-1$, not the virtual events there.

3.1.2 Complete Likelihood

The complete likelihood for the joint RPPs and VPPs for a data point is

$$p(\mathbf{x}, \mathbf{Z}{=}\mathbf{z}, \tilde{\mathbf{Z}}{=}\tilde{\mathbf{z}}) = p(\mathbf{z}_L) \prod_{\ell=0}^{L-1} p(\mathbf{z}_\ell \mid \mathbf{z}_{\ell+1}) \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}_\ell), \quad (4)$$

where

$$p(\mathbf{z}_L) = \textstyle\prod_{k=1}^{K_L} p(\mathbf{z}_{L,k}),$$
$$p(\mathbf{z}_\ell \mid \mathbf{z}_{\ell+1}) = \textstyle\prod_{k=1}^{K_\ell} p(\mathbf{z}_{\ell,k} \mid \mathbf{z}_{\ell+1}),$$
$$\tilde{p}(\tilde{\mathbf{z}}_\ell \mid \mathbf{z}_{\ell-1}) = \textstyle\prod_{k=1}^{K_\ell} \tilde{p}(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1}) \ .$$

The likelihood of $\mathbf{z}_{L,k}$ is

$$p(\mathbf{z}_{L,k}) = \exp\left(-\mu_{L,k} T\right) \mu_{L,k}^{m_{L,k}},$$

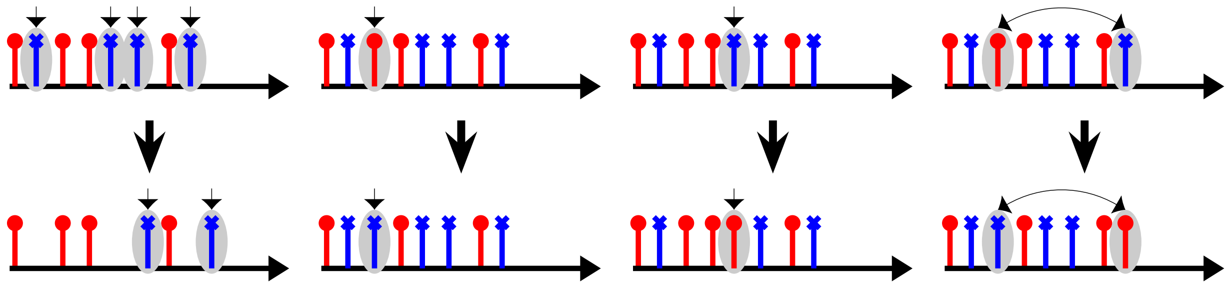


Figure 3: Examples For Sampler Moves. ● Represents a Real Event. ✗ Represents a Virtual Event.

where $m_{L,k}$ is the number of events drawn from $Z_{L,k}$.

The likelihood of $\mathbf{z}_{\ell,k}$ for $0 \leq \ell \leq L-1$ is

$$p(\mathbf{z}_{\ell,k}|\mathbf{z}_{\ell+1}) = \exp\left(\sum_{t_{\ell,k,j}\leq T} \log \lambda_{\ell,k}(t_{\ell,k,j}) - \int_0^T \lambda_{\ell,k}(t)\,dt\right).$$

(Here and for the rest of the paper, we let $\mathbf{z}_{0,k}$ be $\mathbf{x}_k$.)

The likelihood of $\tilde{\mathbf{z}}_{\ell,k}$ for $1 \leq \ell \leq L$ is

$$\tilde{p}(\tilde{\mathbf{z}}_{\ell,k}|\mathbf{z}_{\ell-1}) = \exp\left(\sum_{\tilde{t}_{\ell,k,j}\leq T} \log \tilde{\lambda}_{\ell,k}(\tilde{t}_{\ell,k,j}) - \int_0^T \tilde{\lambda}_{\ell,k}(t)\,dt\right).$$

3.1.3 Sampler Moves

During the sampling process, we first select a hidden TPP uniformly and then apply a move selected randomly from the following three types with a predetermined probability distribution. See Fig. 3 for illustration. More details can be found in Appx. B.

Move 1: Re-sample virtual events. This move re-samples the virtual events for a VPP. The dimensionality of the sample space is changed after each re-sampling. But the determinant of the Jacobian matrix, introduced as the correction for the changes of variables in reversible-jump MCMC (Green, 1995), is 1, since the new variables are independent of the current virtual events of the Markov chain. The acceptance probability is always 1 for this move.

Move 2: Flip. We uniformly pick an event from the union of the samples from the RPP and the VPP, and then propose to change the type for that event. If the type of the picked event is real, we propose to flip it to be a virtual event, and vice versa.

Move 3: Swap. One event is picked uniformly from the samples for each of the RPP and the VPP. Then we propose to swap the types of these two events, *i.e.*, the real event becomes a virtual event and vice versa.

Suppose the proposals for the changes are to adjust the events in $\mathbf{z}_{\ell,k}$ and $\tilde{\mathbf{z}}_{\ell,k}$ to become the events in $\mathbf{z}'_{\ell,k}$ and $\tilde{\mathbf{z}}'_{\ell,k}$. Then the likelihood ratio is

$$\mathcal{P} = \frac{p(\mathbf{z}'_{\ell,k}|\mathbf{z}_{\ell+1})\tilde{p}(\tilde{\mathbf{z}}'_{\ell,k}|\mathbf{z}_{\ell-1})}{p(\mathbf{z}_{\ell,k}|\mathbf{z}_{\ell+1})\tilde{p}(\tilde{\mathbf{z}}_{\ell,k}|\mathbf{z}_{\ell-1})} \cdot \frac{p(\mathbf{z}_{\ell-1}|\mathbf{z}'_{\ell})\tilde{p}(\tilde{\mathbf{z}}_{\ell+1}|\mathbf{z}'_{\ell})}{p(\mathbf{z}_{\ell-1}|\mathbf{z}_{\ell})\tilde{p}(\tilde{\mathbf{z}}_{\ell+1}|\mathbf{z}_{\ell})},$$

where $p(\mathbf{z}_{\ell-1} \mid \mathbf{z}_\ell) = \prod_{k=1}^{K_{\ell-1}} p(\mathbf{z}_{\ell-1,k} \mid \mathbf{z}_\ell)$, and $\tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}_\ell) = \prod_{k=1}^{K_{\ell+1}} \tilde{p}(\tilde{\mathbf{z}}_{\ell+1,k} \mid \mathbf{z}_\ell)$.

The ratio for the proposal probability is 1 for both *Move 2* and *Move 3*. So the acceptance probability for *Move 2* and *Move 3* is $\min(1, \mathcal{P} \cdot 1)$.

It is necessary to have *Move 3* to help accelerate mixing even though *Move 2* seems to already include the ability to swap through two consecutive flips. However, often there is a real event that has large positive contribution to the likelihood in Eq. 4. If we propose to flip this event to a virtual event, the likelihood ratio $\mathcal{P}$ would be very small, hence this real event would stay in a same place for a long time. Similarly, flipping a virtual event first would be unlikely.

With these three sampler moves, we address the form of the VPP intensity functions in Eq. 3:

1. The intensity functions for the VPPs are only controlled by the RPPs on the lower layers. This allows us to sample the VPPs directly and efficiently, using the inversion method (Çinlar, 2013) in *Move 1*, to push the information "upwards."
2. The virtual events tend to appear more at the places where the probability is high to have a real event. In particular, events at higher levels tend to proceed those at lower levels. So it is natural to have virtual kernel functions that evolve with time in the reverse direction to the real kernel functions. The true posterior of the upper level (conditioned on the lower level) is not so simple, but this is a good approximation for the proposal.
3. The base rates $\tilde{\mu}_{\ell,k}$ help accelerate the mixing for the point processes not directly connected to the

evidence. Without them, an upper layer can be "starved" for virtual events (necessary to allow the addition and movement of real points) if a lower layer has few real events.

3.1.4 Update Virtual Kernels Parameters

Because we want the distribution of the proposed virtual events to be as close as possible to the posterior distribution of the real events, we would like to maximize the likelihood of the parameters for the VPPs assuming the posterior samples for the real events are drawn from the VPPs. That is, we adjust the sampler to make it more efficient by tuning the parameters of the VPPs (as the sampler is valid across different VPP parameters).

As the number of events usually varies significantly for different evidence samples, we assume the base rates $\tilde{\mu}_{n,L,i}$ and $\mu_{n,i}$ on the top layer are different for each data point $\mathbf{x}_n$, where n represents the data index.

Let $\mathbf{z}_n^{(1)}, \ldots, \mathbf{z}_n^{(\mathcal{S})}$ be the posterior samples from the distribution $p(\mathbf{Z}_n \mid \mathbf{x}_n; \boldsymbol{\mu}_n, \boldsymbol{\theta})$, where $\boldsymbol{\mu}_n = [\mu_{n,i}]_{i=1}^{K_L}$ are the parameters of the HPPs on the top layer and $\boldsymbol{\theta} = \left[\boldsymbol{\theta}_{(\ell+1,*)\to(\ell,*)}\right]_{\ell=0}^{L-1}$ are the parameters of the kernel functions. Given the log-likelihood of the parameters of the VPPs *w.r.t* the real events

$$\tilde{\mathrm{llh}}\left(\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\mu}}_n; \mathbf{x}_n, \mathbf{z}_n\right) = \sum_{\ell=1}^{L} \tilde{\mathrm{llh}}_{(\ell-1,*)\to(\ell,*);\mathbf{x}_n,\mathbf{z}_n}$$

where

$$\tilde{\mathrm{llh}}_{(\ell-1,*)\to(\ell,*);\mathbf{x}_n,\mathbf{z}_n} = \sum_{k=1}^{K_\ell} \left(\sum_{t_{n,\ell,k,j} \le T} \log \tilde{\lambda}_{n,\ell,k}(t_{n,\ell,k,j}) - \int_0^T \tilde{\lambda}_{n,\ell,k}(t)dt \right).$$

The update rules are

$$\tilde{\boldsymbol{\mu}}_n \leftarrow \tilde{\boldsymbol{\mu}}_n + \frac{\tilde{r}}{\mathcal{S}N} \sum_{s=1}^{\mathcal{S}} \nabla_{\tilde{\boldsymbol{\mu}}_n} \tilde{\mathrm{llh}}\left(\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\mu}}_n; \mathbf{x}_n, \mathbf{z}_n^{(s)}\right), \quad (5)$$

$$\tilde{\boldsymbol{\theta}} \leftarrow \tilde{\boldsymbol{\theta}} + \frac{\tilde{r}}{\mathcal{S}N} \sum_{n=1}^{N} \sum_{s=1}^{\mathcal{S}} \nabla_{\tilde{\boldsymbol{\theta}}} \tilde{\mathrm{llh}}\left(\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\mu}}_n; \mathbf{x}_n, \mathbf{z}_n^{(s)}\right), \quad (6)$$

where

$$\tilde{\boldsymbol{\mu}}_n = \left[[\tilde{\mu}_{n,\ell,k}]_{\ell=1}^{L} \right]_{k=1}^{K_\ell},$$

$$\tilde{\boldsymbol{\theta}} = \left[\left[\left[\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,j)} \right]_{i=1}^{K_{\ell-1}} \right]_{j=1}^{K_\ell} \right]_{\ell=1}^{L},$$

N is the number of data points, and $\tilde{r} > 0$ is the step size for optimizing. See Appx. C.1 for the gradient.

3.2 MCEM

MCEM iteratively applies the following two-step process until the parameters for the RPPs converge.

Step 1. Generate posterior samples $\mathbf{z}_n^{(1)}, \ldots, \mathbf{z}_n^{(\mathcal{S})}$ from the distribution $p(\mathbf{Z}_n \mid \mathbf{x}_n; \boldsymbol{\mu}_n, \boldsymbol{\theta})$.

Step 2. Update the parameters. Given the log-likelihood function of the parameters of the RPPs *w.r.t* the real events

$$\mathrm{llh}(\boldsymbol{\theta}, \boldsymbol{\mu}_n; \mathbf{x}_n, \mathbf{z}_n) = \log \left(\prod_{i=1}^{K_L} p(\mathbf{z}_{n,L,i}) \cdot \prod_{\ell=0}^{L-1} \prod_{i=1}^{K_\ell} p(\mathbf{z}_{n,\ell,i} \mid \mathbf{z}_{n,\ell+1}) \right), \quad (7)$$

the update rules for the parameters are

$$\boldsymbol{\mu}_n \leftarrow \arg\max_{\boldsymbol{\mu}_n} \left\{ \frac{1}{\mathcal{S}} \sum_{s=1}^{\mathcal{S}} \mathrm{llh}(\boldsymbol{\theta}, \boldsymbol{\mu}_n; \mathbf{x}_n, \mathbf{z}_n^{(s)}) \right\}, \quad (8)$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \frac{r}{\mathcal{S}N} \sum_{n=1}^{N} \sum_{s=1}^{\mathcal{S}} \nabla_{\boldsymbol{\theta}} \mathrm{llh}(\boldsymbol{\theta}, \boldsymbol{\mu}_n; \mathbf{x}_n, \mathbf{z}_n^{(s)}), \quad (9)$$

where N is the number of data points and $r > 0$ is the step size for optimizing. See Appx. C.2 for the gradient and maximization formula.

Full maximization is utilized for $\boldsymbol{\mu}_n$ in Eq. 8, and ascent-based MCEM (Caffo et al., 2005) is used for $\boldsymbol{\theta}$ in Eq. 9. When the sample size $\mathcal{S}$ goes to infinity and we update the parameters as in Eq. 9, the expected value of the log-likelihood in Eq. 7 increases at each iteration with probability converging to 1 (Caffo et al., 2005). The general theory for the convergence of MCEM has not been well-established. Different senses and different approaches for the convergence analysis are given by Neath et al. (2013). We tried various settings for MCEM and choose the one that is stable and computationally efficient. Adam (Kingma & Ba, 2015) was utilized for the optimization of the parameters of the kernel and virtual kernel functions. Notice that we are not changing the parameters for the real or virtual kernels during MCMC sampling. The posterior samples for the real events are collected from the MCMC sampler after it reaches convergence. Pseudo-code is given in Appx. F.

4 EXPERIMENTS

The code is available online at `https://github.com/hongchengkuan/Deep-Neyman-Scott-Processes`.

4.1 Architectures

We constructed hierarchical models as in Fig. 4. Black horizontal arrows are point processes. Gray arrows are

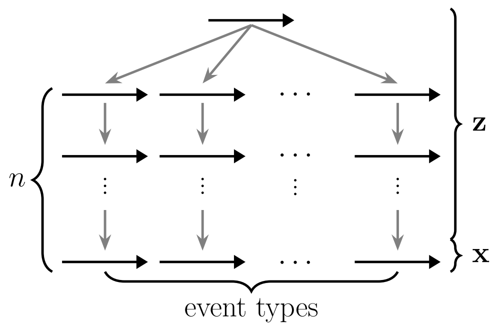


Figure 4: n-hidden

model connections. For the 1-hidden model, we only have one layer of hidden TPPs and there is only one TPP in total on the top layer. The hidden TPP is connected to all the types of events in the evidence.

For the 2-hidden model, we have the one hidden TPP on layer 1 for each type in the evidence with a single connection between matched hidden- and observed-TPPs, *i.e.*, $\phi_{\boldsymbol{\theta}_{(1,i)\to(0,j)}}(\cdot) = 0$ if $i \neq j$. There is also only one hidden TPP on the top layer, which is connected to all hidden TPPs on layer 1.

The n-hidden model can be constructed by adding more hidden layers for each type similarly. More details about training, testing, and prediction are given in Appx. G. For each dataset, the **best** result is shown in bold and underlined, the **runner-up** in bold.

4.2 Synthetic Data Experiments

We use synthetic data to illustrate the modeling power of multiple layers. The synthetic data are generated from DN-SPs with differing numbers of hidden layers and 2 event types. They are divided into training and test sets. Then we apply our models with different number of hidden layers to train and test the synthetic datasets. The log-likelihood per event, time prediction root mean squared error (RMSE) and type prediction accuracy for the test set are shown in Table 1. The left-most column represents the different depths we use to generate synthetic data and the right 3 columns are the results when we use different model depths for training and testing. For log-likelihood (which our model is trained for), increasing depth helps, particularly up to the depth of the data. Accuracy and RMSE have similar behavior.

4.3 Real-world Data Experiments

We compare our DN-SP to four currently popular continuous-time event modeling methods: MTPP (Lian et al., 2015), the neural Hawkes process (NHP) (Mei & Eisner, 2017), the self-attentive Hawkes process (SAHP) (Zhang et al., 2020a), and the transformer Hawkes process (THP) (Zuo et al., 2020). MTPP combines Gaussian processes and piecewise-constant intensity models (Gunawardana et al., 2011), while the other baselines are based on neural networks. The datasets we use are retweets, earthquakes, and homicides. The details for these datasets can be found in Appx. G.1. Each dataset is split into training, validation, and test sets randomly five times. We report the mean values with the standard deviations shown in parentheses. Validation is used to tune the hyperparameters and early stopping for all the datasets for NHP, SAHP and THP. For DN-SPs, validation is only used for early stopping for retweets dataset, as we only use mini-batch gradient ascent for retweets. Batch gradient ascent and termination based on training are used for earthquakes and homicides. For MTPP, each task corresponds to a type in our experiments. The model parameters of MTPP are fixed across different sequences and each sequence has their own variational parameters. We use forward sampling to do the prediction for MTPP in a similar way as in Appx. G.4.

The results in Table 2 indicate that our model achieves competitive results compared to the baselines. We have the best RMSE for earthquakes and homicides, the best accuracy for earthquakes, and the best likelihood for retweets and homicides. The likelihood for the earthquakes dataset is only a little worse than SAHP and THP. Note that *our model only needs to fit tens of parameters during training, compared to the hundreds of thousands parameters needed for the neural networks.*

It is also notable that the 2-hidden is better than 1-hidden in terms of likelihood and accuracy while, for RMSE, 2-hidden is worse than 1-hidden consistently. It indicates that separate hidden TPPs for each type does increase the power to fit the evidence. The RMSE of the 1-hidden model is not maintained when increasing model capacity to the 2-hidden model, because we are fitting to likelihood which is related but different than RMSE. Note that on the homicides data, our RMSE error for both 2-hidden and 1-hidden are significantly better than those of prior work.

We also ran the real-world experiments for both a fully connected model and a model with more than 2 hidden layers. The likelihood, RMSE, and accuracy were not improved, because the single top layer is sufficient to correlate events across the different types. The time complexity analysis can be found in Appx. G.5.

5 RELATED WORK

Neural Networks Recent work uses neural networks to directly model the CIF for a TPP, *e.g.*, recurrent neural networks (Du et al., 2016; Mei & Eisner,

Table 1: Synthetic Experimental Results

Synthetic Datasets	Model	Log-likelihood	RMSE	Accuracy
2-hidden	1-hidden	−0.006	1.048	0.722
	2-hidden	0.286	**0.942**	**0.760**
	3-hidden	**0.301**	**1.010**	**0.762**
	4-hidden	**0.304**	1.189	0.757
3-hidden	1-hidden	0.528	0.731	0.782
	2-hidden	**0.822**	**0.611**	**0.812**
	3-hidden	**0.835**	**0.613**	**0.814**
	4-hidden	0.831	0.670	0.809
4-hidden	1-hidden	1.177	0.411	0.820
	2-hidden	1.458	0.409	0.845
	3-hidden	**1.464**	**0.391**	**0.846**
	4-hidden	**1.466**	**0.391**	**0.846**

Table 2: Real-world Experimental Results

Datasets	Model	Log-likelihood	RMSE($\times 10^4$)	Accuracy
Retweets	MTPP	−13.44(0.18)	1.64(0.03)	0.36(0.00)
	NHP	−6.12(0.03)	1.64(0.03)	0.48(0.01)
	SAHP	−4.38(0.17)	**1.60(0.04)**	0.51(0.03)
	THP	−4.63(0.03)	**1.54(0.03)**	**0.61(0.00)**
	1-hidden	**−4.15(0.07)**	**1.60(0.03)**	0.48(0.00)
	2-hidden	**−3.54(0.15)**	1.69(0.07)	**0.57(0.00)**
Earthquakes	MTPP	−10.45(0.09)	0.20(0.01)	0.52(0.00)
	NHP	−8.70(0.08)	0.20(0.01)	0.39(0.01)
	SAHP	**−8.29(0.25)**	**0.16(0.01)**	0.56(0.03)
	THP	**−8.13(0.05)**	0.20(0.01)	**0.61(0.01)**
	1-hidden	−8.45(0.05)	**0.15(0.01)**	**0.61(0.01)**
	2-hidden	−8.43(0.06)	**0.16(0.01)**	**0.60(0.01)**
Homicides	MTPP	−18.15(1.40)	23.14(3.41)	0.20(0.02)
	NHP	−21.93(0.93)	23.13(3.42)	0.18(0.03)
	SAHP	−14.02(0.33)	23.13(3.42)	0.19(0.02)
	THP	−14.87(0.63)	23.13(3.42)	**0.26(0.03)**
	1-hidden	**−11.68(0.20)**	**17.40(2.68)**	**0.25(0.01)**
	2-hidden	**−10.78(0.11)**	**17.73(2.84)**	**0.25(0.02)**

2017) and self-attention mechanism (Zuo et al., 2020; Zhang et al., 2020a). Omi et al. (2019) model the cumulative hazard function. And, some others attempt to model the conditional distribution of the inter-event times (Mehrasa et al., 2019; Shchur et al., 2020). All of these methods assume the latent space is deterministically identified by a neural network, which lacks the natural flexibility induced by the randomness of a stochastic process. Most critically, a large number of parameters is generally required.

Gaussian Processes (GP) Modulated Point Processes Others have constructed GP-modulated point processes with the assumption that the intensity functions are smooth (*e.g.* Møller et al., 1998; Kottas, 2006; Kottas & Sansó, 2007; Adams et al., 2009; Rao & Teh, 2011b; Gunter et al., 2014; Samo & Roberts, 2015; Lloyd et al., 2015; Donner & Opper, 2018; Aglietti et al., 2019). However, the intensity functions can experience sudden changes upon events arrivals, *e.g.*, a big earthquake can dramatically increase the probability to have a small earthquake in the near future. And the integral of the intensity function is not available in a closed form, requiring approximations in its computation. Others have imposed GP priors on the CIFs of Hawkes processes (Zhou et al., 2019; Zhang et al., 2019, 2020b; Zhou et al., 2020), which assume the sudden changes can only happen at the times of the observed events. All of these GP-modulated models are not able to be stacked to construct a deep model built solely with point processes.

Piecewise-constant Intensity Model (PCIM) Another line of work it to use a PCIM (Gunawardana et al., 2011) or other variants (Weiss & Page, 2013; Lian et al., 2015). The history of events is embedded into a piecewise-constant function, which is the intensity function itself or some other function that can output the intensity. The size of time windows for the calculation of the piecewise-constant function needs to be pre-determined.

6 CONCLUSION

We build a deep Neyman-Scott process (DN-SP) and use it to model real-world event sequences. Different from the existing methods for Cox processes, we are able to stack point processes in a hierarchical manner and do not assume the intensity function is smooth in the space. We propose and test an efficient MCMC posterior sampling algorithm for DN-SPs. Virtual events as auxiliary variables help accelerate the mixing of our Markov chains. With the fast posterior sampling, we are able to do inference for large datasets. Our encouraging experiments results suggest that it is promising to build a deep model with point processes. Future research directions include constructing more complicated kernels, designing variational inference algorithm, and generalizing the DN-SPs into more general spaces.

Acknowledgments

Computational resources were provided through Office of Naval Research grant N00014-18-1-2252.

Data for this study came from the Berkeley Digital Seismic Network (BDSN), doi:10/7932/BDSN; the High Resolution Seismic Network (HRSN), doi:10.7932/HRSN; and the Bay Area Regional Deformation Network (BARD), doi:10.7932/BARD, all operated by the UC Berkeley Seismological Laboratory and archived at the Northern California Earthquake Data center (NCEDC), doi: 10/7932/NCEDC.

References

Adams, R. P. *Kernel methods for nonparametric Bayesian inference of probability densities and point processes.* PhD thesis, University of Cambridge, 2009.

Adams, R. P., Murray, I., and MacKay, D. J. Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 9–16, 2009.

Aglietti, V., Bonilla, E. V., Damoulas, T., and Cripps, S. Structured variational inference in continuous Cox process models. In *Advances in Neural Information Processing Systems*, pp. 12458–12468, 2019.

Andersen, I. T., Hahn, U., Arnspang, E. C., Nejsum, L. N., and Jensen, E. B. V. Double Cox cluster processes—with applications to photoactivated localization microscopy. *Spatial statistics*, 27:58–73, 2018.

Baddeley, A. and Møller, J. Nearest-neighbour Markov point processes and random sets. *International Statistical Review/Revue Internationale de Statistique*, pp. 89–121, 1989.

BARD. Bay area regional deformation network. UC Berkeley Seismological Laboratory. dataset., 2014.

Bauwens, L. and Hautsch, N. Modelling financial high frequency data using point processes. In *Handbook of financial time series*, pp. 953–979. Springer, 2009.

BDSN. Berkeley digital seismic network. UC Berkeley Seismological Laboratory. dataset., 2014.

Caffo, B. S., Jank, W., and Jones, G. L. Ascent-based Monte Carlo expectation-maximization. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):235–251, 2005.

Çinlar, E. *Introduction to stochastic processes.* Dover Publications, 2013.

Clifford, P. and Nicholls, G. Comparison of birth-and-death and Metropolis-Hastings Markov chain Monte Carlo for the Strauss process. 1994.

COC. City of Chicago, Crimes - 2001 to present. `https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-Present/ijzp-q8t2`.

Cox, D. R. Some statistical methods connected with series of events. *Journal of the Royal Statistical Society: Series B (Methodological)*, 17(2):129–157, 1955.

Daley, D. and Vere-Jones, D. *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods.* Springer-Verlag New York, 2nd edition, 2003.

Donner, C. and Opper, M. Efficient Bayesian inference of sigmoidal Gaussian Cox processes. *The Journal of Machine Learning Research*, 19(1):2710–2743, 2018.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1555–1564, 2016.

Geyer, C. J. and Møller, J. Simulation procedures and likelihood inference for spatial point processes. *Scandinavian journal of statistics*, pp. 359–373, 1994.

Green, P. J. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

Gunawardana, A., Meek, C., and Xu, P. A model for temporal dependencies in event streams. In *Advances in Neural Information Processing Systems*, pp. 1962–1970, 2011.

Gunter, T., Lloyd, C., Osborne, M. A., and Roberts, S. J. Efficient Bayesian nonparametric modelling of structured point processes. In *Uncertainty in Artificial Intelligence (UAI)*, 2014.

Hastings, W. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Hawkes, A. G. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1): 83–90, 1971.

HRSN. High resolution seismic network. UC Berkeley Seismological Laboratory. dataset., 2014.

Kallenberg, O. An informal guide to the theory of conditioning in point processes. *International Statistical Review/Revue Internationale de Statistique*, pp. 151–164, 1984.

Kelly, F. P. and Ripley, B. D. A note on Strauss's model for clustering. *Biometrika*, pp. 357–360, 1976.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Kottas, A. Dirichlet process mixtures of beta distributions, with applications to density and intensity estimation. In *Workshop on Learning with Nonparametric Bayesian Methods, 23rd International Conference on Machine Learning (ICML)*, volume 47. Citeseer, 2006.

Kottas, A. and Sansó, B. Bayesian mixture modeling for spatial Poisson process intensities, with applications to extreme value analysis. *Journal of Statistical Planning and Inference*, 137(10):3151–3163, 2007.

Lian, W., Henao, R., Rao, V., Lucas, J., and Carin, L. A multitask point process predictive model. In *International Conference on Machine Learning*, pp. 2030–2038, 2015.

Lieshout, V., M., M. N., and Baddeley, A. J. Extrapolating and interpolating spatial patterns. In Lawson, A. B. and Denison, D. G. (eds.), *Spatial Cluster Modelling*, chapter 4, pp. 61–86. Chapman & Hall/CRC, 2002.

Linderman, S. W., Wang, Y., and Blei, D. M. Bayesian inference for latent Hawkes processes. In *Advances in Approximate Bayesian Inference*, 2017.

Lloyd, C., Gunter, T., Osborne, M., and Roberts, S. Variational inference for Gaussian process modulated Poisson processes. In *International Conference on Machine Learning*, pp. 1814–1822, 2015.

Mehrasa, N., Jyothi, A. A., Durand, T., He, J., Sigal, L., and Mori, G. A variational auto-encoder model for stochastic point processes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3165–3174, 2019.

Mei, H. and Eisner, J. M. The neural Hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, pp. 6754–6764, 2017.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Møller, J. On the rate of convergence of spatial birth-and-death processes. *Annals of the Institute of Statistical Mathematics*, 41(3):565–581, 1989.

Møller, J. and Waagepetersen, R. P. *Statistical inference and simulation for spatial point processes*. CRC Press, 2003.

Møller, J., Syversveen, A. R., and Waagepetersen, R. P. Log Gaussian Cox processes. *Scandinavian journal of statistics*, 25(3):451–482, 1998.

NCEDC. Northern California earthquake data center. UC Berkeley Seismological Laboratory. dataset., 2014.

Neath, R. C. et al. On convergence properties of the Monte Carlo EM algorithm. In *Advances in modern statistical theory and applications: a Festschrift in Honor of Morris L. Eaton*, pp. 43–62. Institute of Mathematical Statistics, 2013.

Neyman, J. and Scott, E. L. Statistical approach to problems of cosmology. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(1): 1–29, 1958.

Norman, G. and Filinov, V. Investigations of phase transitions by a Monte-Carlo method. *High Temperature*, 7(2):216, 1969.

Omi, T., Aihara, K., et al. Fully neural network based model for general temporal point processes. In *Advances in Neural Information Processing Systems*, pp. 2122–2132, 2019.

Papangelou, F. The conditional intensity of general point processes and an application to line processes. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 28(3):207–226, 1974.

Perkel, D. H., Gerstein, G. L., and Moore, G. P. Neuronal spike trains and stochastic point processes: I. the single spike train. *Biophysical journal*, 7(4): 391–418, 1967.

Preston, C. Spatial birth and death processes. *Bulletin of the International Statistical Institute*, 46:371–391, 1977.

Qin, Z. and Shelton, C. R. Auxiliary Gibbs sampling for inference in piecewise-constant conditional intensity models. In *UAI*, pp. 722–731, 2015.

Rao, V. and Teh, Y. W. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 619–626, 2011a.

Rao, V. and Teh, Y. W. Fast MCMC sampling for Markov jump processes and extensions. *The Journal of Machine Learning Research*, 14(1):3295–3320, 2013.

Rao, V. A. and Teh, Y. W. Gaussian process modulated renewal processes. In *NIPS*, pp. 2474–2482, 2011b.

Ripley, B. D. Modelling spatial patterns. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(2):172–192, 1977.

Samo, Y.-L. K. and Roberts, S. Scalable nonparametric Bayesian inference on point processes with Gaussian processes. In *International Conference on Machine Learning*, pp. 2227–2236. PMLR, 2015.

Shchur, O., Bilos, M., and Günnemann, S. Intensity-free learning of temporal point processes. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HygOjhEYDH.

Shelton, C. R., Qin, Z., and Shetty, C. Hawkes process inference with missing data. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Stoica, R., Tempel, E., Liivamägi, L., Castellan, G., and Saar, E. Spatial patterns analysis in cosmology based on marked point processes. *European Astronomical Society Publications Series*, 66:197–226, 2014. doi: 10.1051/eas/1466013.

Wei, G. C. and Tanner, M. A. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American statistical Association*, 85(411):699–704, 1990.

Weiss, J. C. and Page, D. Forest-based point process for event prediction from electronic health records. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 547–562. Springer, 2013.

Williams, A., Degleris, A., Wang, Y., and Linderman, S. Point process models for sequence detection in high-dimensional neural spike trains. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 14350–14361. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/a5481cd6d7517aa3fc6476dc7d9019ab-Paper.pdf.

Yau, C. Y. and Loh, J. M. A generalization of the Neyman-Scott process. *Statistica Sinica*, pp. 1717–1736, 2012.

Zhang, Q., Lipani, A., Kirnap, O., and Yilmaz, E. Self-attentive Hawkes process. In *International Conference on Machine Learning*, 2020a.

Zhang, R., Walder, C., Rizoiu, M.-A., and Xie, L. Efficient non-parametric Bayesian Hawkes processes. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 4299–4305. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/597. URL https://doi.org/10.24963/ijcai.2019/597.

Zhang, R., Walder, C., and Rizoiu, M.-A. Variational inference for sparse Gaussian process modulated Hawkes process. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6803–6810, 2020b.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., and Leskovec, J. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1513–1522, 2015.

Zhou, F., Li, Z., Fan, X., Wang, Y., Sowmya, A., and Chen, F. Efficient EM-variational inference for Hawkes process. *arXiv preprint arXiv:1905.12251*, 2019.

Zhou, F., Li, Z., Fan, X., Wang, Y., Sowmya, A., and Chen, F. Efficient inference for nonparametric Hawkes processes using auxiliary latent variables. *Journal of Machine Learning Research*, 21(241):1–31, 2020.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. Transformer Hawkes process. In *International Conference on Machine Learning*, 2020.

Supplementary Material: Deep Neyman-Scott Processes

A POSTERIOR PCIF

Lemma 1 (Kallenberg (1984))**.** *The PCIF for a point process Ξ defined on S with density f is*

$$\lambda_{\mathcal{P}}(t) = \frac{f(\xi \cup \{t\})}{f(\xi)}, \ t \in S \backslash \xi,$$

with ξ as a realization of Ξ and $a/0 = 0$ for $a \geq 0$.

Proposition 1. *The PCIF for the posterior point process of $Z_{\ell,k}$ is*

$$\lambda_{\mathcal{P};\ell,k}(t) = \lambda_{\ell,k}(t) \prod_{i=1}^{K_{\ell-1}} \left(\exp\left(-\Phi_{(\ell,k)\to(\ell-1,i)}(T-t)\right) \prod_{t_{\ell-1,i,j}>t} \frac{\lambda'_{\ell-1,i}(t_{\ell-1,i,j}, t)}{\lambda_{\ell-1,i}(t_{\ell-1,i,j})} \right), \tag{10}$$

where $\lambda'_{\ell-1,i}(x,y) = \lambda_{\ell-1,i}(x) + \phi_{(\ell,k)\to(\ell-1,i)}(x-y)$ and $\Phi_{(\ell,k)\to(\ell-1,i)}(x) = \int_0^x \phi_{\boldsymbol{\theta}_{(\ell,k)\to(\ell-1,i)}}(\tau)d\tau$.

Proof. The $p.d.f$ to have an event at time t for $Z_{\ell,k}$ is

$$p(t, \mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}, \mathbf{z}_\ell/\{t\}, \mathbf{z}_{\ell+1}, \ldots, \mathbf{z}_L) = p(\mathbf{z}_L)p(\mathbf{z}_{L-1} \mid \mathbf{z}_L) \cdots p(\mathbf{x} \mid \mathbf{z}_1),$$

where $\mathbf{z}_\ell/\{t\}$ means there is no event at time t for $Z_{\ell,k}$.

If there is no event at time t, the $p.d.f$ conditional on the information of the point processes except time t is

$$p(\mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}, \mathbf{z}_\ell/\{t\}, \mathbf{z}_{\ell+1}, \ldots, \mathbf{z}_L) = p(\mathbf{z}_L)p(\mathbf{z}_{L-1} \mid \mathbf{z}_L) \cdots p(\mathbf{z}_\ell/\{t\} \mid \mathbf{z}_{\ell+1}) \cdots p(\mathbf{x} \mid \mathbf{z}_1).$$

Thus, according to Lemma 1, the posterior PCIF to have an event at time t is

$$\begin{aligned}\lambda_{\mathcal{P};\ell,k}(t) &= \frac{p(t, \mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}, \mathbf{z}_\ell/\{t\}, \mathbf{z}_{\ell+1}, \ldots, \mathbf{z}_L)}{p(\mathbf{x}, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}, \mathbf{z}_\ell/\{t\}, \mathbf{z}_{\ell+1}, \ldots, \mathbf{z}_L)} \\ &= \frac{p(\mathbf{z}_\ell/\{t\} \cup \{t\} \mid \mathbf{z}_{\ell+1})}{p(\mathbf{z}_\ell/\{t\} \mid \mathbf{z}_{\ell+1})} \\ &= \lambda_{\ell,k}(t) \prod_{i=1}^{K_{\ell-1}} \left(\exp\left(-\Phi_{(\ell,k)\to(\ell-1,i)}(T-t)\right) \prod_{t_{\ell-1,i,j}>t} \frac{\lambda'_{\ell-1,i}(t_{\ell-1,i,j}, t)}{\lambda_{\ell-1,i}(t_{\ell-1,i,j})} \right).\end{aligned}$$

□

B SAMPLER MOVES

Recall

$$\mathcal{P} = \frac{p(\mathbf{x}, \mathbf{z}', \tilde{\mathbf{z}}')}{p(\mathbf{x}, \mathbf{z}, \tilde{\mathbf{z}})} = \frac{p(z'_{\ell,k} \mid \mathbf{z}_{\ell+1}) \cdot \tilde{p}(\tilde{z}'_{\ell,k} \mid \mathbf{z}_{\ell-1})}{p(z_{\ell,k} \mid \mathbf{z}_{\ell+1}) \cdot \tilde{p}(\tilde{z}_{\ell,k} \mid \mathbf{z}_{\ell-1})} \cdot \frac{p(\mathbf{z}_{\ell-1} \mid \mathbf{z}'_\ell) \cdot \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}'_\ell)}{p(\mathbf{z}_{\ell-1} \mid \mathbf{z}_\ell) \cdot \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}_\ell)},$$

where

$$p(\mathbf{z}_{\ell-1} \mid \mathbf{z}_\ell) = \prod_{k=1}^{K_{\ell-1}} p(\mathbf{z}_{\ell-1,k} \mid \mathbf{z}_\ell),$$

$$\tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}_\ell) = \prod_{k=1}^{K_{\ell+1}} \tilde{p}(\tilde{\mathbf{z}}_{\ell+1,k} \mid \mathbf{z}_\ell).$$

B.1 Re-sample Virtual Events

Suppose we want to re-sample the virtual events for $\tilde{Z}_{\ell,k}$, the proposal probability is

$$q(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1}) = \tilde{p}(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1}),$$

and the integrated detailed balance equation is

$$\begin{aligned}&\int_A \int_{A'} p(\mathbf{x}, \mathbf{z}, \tilde{\mathbf{z}}) q(\tilde{\mathbf{z}}'_{\ell,k} \mid \mathbf{z}_{\ell-1}) \alpha(\tilde{\mathbf{z}}_{\ell,k}, \tilde{\mathbf{z}}'_{\ell,k}) d\tilde{\mathbf{z}}'_{\ell,k} d\tilde{\mathbf{z}}_{\ell,k}\\ =&\int_A \int_{A'} p(\mathbf{x}, \mathbf{z}', \tilde{\mathbf{z}}') q(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1}) \alpha(\tilde{\mathbf{z}}'_{\ell,k}, \tilde{\mathbf{z}}_{\ell,k}) d\tilde{\mathbf{z}}'_{\ell,k} d\tilde{\mathbf{z}}_{\ell,k},\end{aligned}$$

where $\tilde{\mathbf{z}}_{\ell,k} \in A$ and $\tilde{\mathbf{z}}'_{\ell,k} \in A'$.

The proposal probability ratio is

$$\mathcal{Q} = \frac{q(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1})}{q(\tilde{\mathbf{z}}'_{\ell,k} \mid \mathbf{z}_{\ell-1})} = \frac{\tilde{p}(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1})}{\tilde{p}(\tilde{\mathbf{z}}'_{\ell,k} \mid \mathbf{z}_{\ell-1})}.$$

The likelihood ratio is

$$\mathcal{P} = \frac{p(\mathbf{x}, \mathbf{z}', \tilde{\mathbf{z}}')}{p(\mathbf{x}, \mathbf{z}, \tilde{\mathbf{z}})} = \frac{\tilde{p}(\tilde{\mathbf{z}}'_{\ell,k} \mid \mathbf{z}_{\ell-1})}{\tilde{p}(\tilde{\mathbf{z}}_{\ell,k} \mid \mathbf{z}_{\ell-1})}.$$

The absolute value of the determinant of the Jacobian is

$$\mathcal{J} = \left| \frac{\partial(\tilde{\mathbf{z}}'_{\ell,k}, \tilde{\mathbf{z}}_{\ell,k})}{\partial(\tilde{\mathbf{z}}'_{\ell,k}, \tilde{\mathbf{z}}_{\ell,k})} \right| = 1.$$

So the acceptance probability is

$$\alpha(\tilde{\mathbf{z}}_{\ell,k}, \tilde{\mathbf{z}}'_{\ell,k}) = \mathcal{P} \cdot \mathcal{Q} \cdot \mathcal{J} = 1.$$

B.2 Flip a Virtual Event to a Real Event

Suppose we want to flip a virtual event $\tilde{t}_{\ell,k,j}$ to a real event $t_{\ell,k,m_{\ell,k}+1}$.

The number of events for the current state and the proposed state is $m_{\ell,k} + \tilde{m}_{\ell,k}$. The probabilities to pick $\tilde{t}_{\ell,k,j}$ and $t_{\ell,k,m_{\ell,k}+1}$ are both $1/(m_{\ell,k} + \tilde{m}_{\ell,k})$, as the total number of events does not change. Hence, the proposal probability ratio is

$$\mathcal{Q} = \frac{1/(m_{\ell,k} + \tilde{m}_{\ell,k})}{1/(m_{\ell,k} + \tilde{m}_{\ell,k})} = 1.$$

The likelihood ratio is

$$\begin{aligned}\mathcal{P}_v =& \frac{\lambda_{\ell,k}(\tilde{t}_{\ell,k,j})}{\tilde{\lambda}_{\ell,k}(\tilde{t}_{\ell,k,j})} \cdot \frac{p(\mathbf{z}_{\ell-1} \mid \mathbf{z}'_{\ell}) \cdot \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}'_{\ell})}{p(\mathbf{z}_{\ell-1} \mid \mathbf{z}_{\ell}) \cdot \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}_{\ell})}\\ =& \frac{\lambda_{\ell,k}(\tilde{t}_{\ell,k,j})}{\tilde{\lambda}_{\ell,k}(\tilde{t}_{\ell,k,j})}\\ &\cdot \prod_{i=1}^{K_{\ell-1}} \left(\exp\left(-\Phi_{\boldsymbol{\theta}_{(\ell,k)\to(\ell-1,i)}}(T - \tilde{t}_{\ell,k,j})\right) \frac{\prod_{t_{\ell-1,i,j} > \tilde{t}_{\ell,k,j}} \lambda'_{\ell-1,i}(t_{\ell-1,i,j}, \tilde{t}_{\ell,k,j})}{\prod_{t_{\ell-1,i,j} > \tilde{t}_{\ell,k,j}} \lambda_{\ell-1,i}(t_{\ell-1,i,j})} \right)\\ &\cdot \prod_{i=1}^{K_{\ell+1}} \left(\exp\left(-\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,k)\to(\ell+1,i)}}(\tilde{t}_{\ell,k,j})\right) \frac{\prod_{\tilde{t}_{\ell+1,i,j} < \tilde{t}_{\ell,k,j}} \tilde{\lambda}'_{\ell+1,i}(\tilde{t}_{\ell+1,i,j}, \tilde{t}_{\ell,k,j})}{\prod_{\tilde{t}_{\ell+1,i,j} < \tilde{t}_{\ell,k,j}} \tilde{\lambda}_{\ell+1,i}(\tilde{t}_{\ell+1,i,j})} \right),\end{aligned}$$

where $\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,k)\to(\ell+1,i)}}(x) = \int_0^x \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,k)\to(\ell+1,i)}}(\tau) d\tau$, $\lambda'_{\ell-1,i}(x,y) = \lambda_{\ell-1,i}(x) + \phi_{\boldsymbol{\theta}_{(\ell,k)\to(\ell-1,i)}}(x-y)$, and $\tilde{\lambda}'_{\ell+1,i}(x,y) = \tilde{\lambda}_{\ell+1,i}(x) + \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,k)\to(\ell+1,i)}}(y-x)$.

So the acceptance probability is

$$\alpha = \min(1, \mathcal{P}_v).$$

B.3 Flip a Real Event to a Virtual Event

Suppose we want to flip a real event $t_{\ell,k,j}$ to a virtual event $\tilde{t}_{\ell,k,\tilde{m}_{\ell,k}+1}$.

Similar to Section B.2, the proposal probability ratio is $\mathcal{Q} = 1$.

The likelihood ratio is

$$\begin{aligned}\mathcal{P}_r =& \frac{\tilde{\lambda}_{\ell,k}(t_{\ell,k,j})}{\lambda_{\ell,k}(t_{\ell,k,j})} \cdot \frac{p(\mathbf{z}_{\ell-1} \mid \mathbf{z}'_\ell) \cdot \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}'_\ell)}{p(\mathbf{z}_{\ell-1} \mid \mathbf{z}_\ell) \cdot \tilde{p}(\tilde{\mathbf{z}}_{\ell+1} \mid \mathbf{z}_\ell)} \\ =& \frac{\tilde{\lambda}_{\ell,k}(t_{\ell,k,j})}{\lambda_{\ell,k}(t_{\ell,k,j})} \\ & \cdot \prod_{i=1}^{K_{\ell-1}} \left(\exp\left(\Phi_{\boldsymbol{\theta}_{(\ell,k)\to(\ell-1,i)}}(T - t_{\ell,k,j})\right) \frac{\prod_{t_{\ell-1,i,j} > t_{\ell,k,j}} \lambda'_{\ell-1,i}(t_{\ell-1,i,j}, t_{\ell,k,j})}{\prod_{t_{\ell-1,i,j} > t_{\ell,k,j}} \lambda_{\ell-1,i}(t_{\ell-1,i,j})} \right) \\ & \cdot \prod_{i=1}^{K_{\ell+1}} \left(\exp\left(\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,k)\to(\ell+1,i)}}(t_{\ell,k,j})\right) \frac{\prod_{\tilde{t}_{\ell+1,i,j} < t_{\ell,k,j}} \tilde{\lambda}'_{\ell+1,i}(\tilde{t}_{\ell+1,i,j}, t_{\ell,k,j})}{\prod_{\tilde{t}_{\ell+1,i,j} < t_{\ell,k,j}} \tilde{\lambda}_{\ell+1,i}(\tilde{t}_{\ell+1,i,j})} \right),\end{aligned}$$

where $\lambda'_{\ell-1,i}(x, y) = \lambda_{\ell-1,i}(x) - \phi_{\boldsymbol{\theta}_{(\ell,k)\to(\ell-1,i)}}(x - y)$, and $\tilde{\lambda}'_{\ell+1,i}(x, y) = \tilde{\lambda}_{\ell+1,i}(x) - \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,k)\to(\ell+1,i)}}(y - x)$.

So the acceptance probability is

$$\alpha = \min(1, \mathcal{P}_r).$$

B.4 Swap a Real Event and a Virtual Event

Suppose we want to flip a virtual event $\tilde{t}_{\ell,k,j}$ to a real event $t_{\ell,k,m_{\ell,i}+1}$ and we also want to flip a real event $t_{\ell,k,j}$ to a virtual event $\tilde{t}_{\ell,k,\tilde{m}_{\ell,k}+1}$.

The probability to pick $\tilde{t}_{\ell,k,j}$ and $t_{\ell,k,j}$ is $1/m_{\ell,k} \cdot 1/\tilde{m}_{\ell,k}$, the same for the probability to pick the real and virtual events in the reverse move. Thus, the proposal probability ratio is $\mathcal{Q} = 1$.

The likelihood ratio is

$$\mathcal{P} = \mathcal{P}_v \cdot \mathcal{P}_r \;.$$

So the acceptance probability is

$$\alpha = \min(1, \mathcal{P}).$$

C OPTIMIZATION

C.1 Derivatives *w.r.t* the Parameters for VPPs

For simplicity, we omit the data index n. The likelihood of the parameters for the VPPs $w.r.t$ the real events is

$$\tilde{\text{llh}} = \sum_{\ell=1}^{L} \sum_{k=1}^{K_\ell} \left(\sum_{j=1}^{\tilde{m}_{\ell,k}} \log \tilde{\lambda}_{\ell,k}(t_{\ell,k,j}) - \int_0^T \tilde{\lambda}_{\ell,k}(t) dt \right),$$

where

$$\int_0^T \tilde{\lambda}_{\ell,k}(t) dt = \tilde{\mu}_{\ell,k} T + \sum_{i=1}^{K_{\ell-1}} \sum_{j=1}^{m_{\ell-1,i}} \tilde{\Phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(t_{\ell-1,i,j}).$$

The integral of the virtual kernel function is

$$\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(t) = \int_0^t \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell,i)\to(\ell+1,k)}}(\tau) d\tau = \int_0^t \tilde{p} \frac{\tilde{\beta}^{\tilde{\alpha}}}{\Gamma(\tilde{\alpha})} \tau^{\tilde{\alpha}-1} e^{-\tilde{\beta}\tau} d\tau = \tilde{p} \frac{1}{\Gamma(\tilde{\alpha})} \gamma(\tilde{\alpha}, \tilde{\beta} t)$$

with $\tilde{p} = \tilde{p}_{(\ell-1,i)\to(\ell,k)}$, $\tilde{\alpha} = \tilde{\alpha}_{(\ell-1,i)\to(\ell,k)}$, $\tilde{\beta} = \tilde{\beta}_{(\ell-1,i)\to(\ell,k)}$.

The partial derivative of $\tilde{\text{llh}}$ $w.r.t$ $\tilde{\mu}_{\ell,k}$ is

$$\partial_{\tilde{\mu}_{\ell,k}}\tilde{\text{llh}} = \sum_{j=1}^{\tilde{m}_{\ell,k}} \frac{1}{\tilde{\lambda}_{\ell,k}(t_{\ell,k,j})} - T.$$

In the following section, we use $\tilde{\boldsymbol{\theta}}$ to denote $\tilde{\boldsymbol{\theta}}_{(\ell-1,*)\to(\ell,k)}$.

The partial derivative of $\tilde{\text{llh}}$ $w.r.t$ $\tilde{p}$ is

$$\begin{aligned}\partial_{\tilde{p}}\tilde{\text{llh}} =&\partial_{\tilde{p}}\left(\sum_{j=1}^{\tilde{m}_{\ell,k}} \log\tilde{\lambda}_{\ell,k}(t_{\ell,k,j}) - \sum_{i=1}^{K_{\ell-1}}\sum_{j=1}^{m_{\ell-1,i}} \tilde{\Phi}_{\tilde{\boldsymbol{\theta}}}(t_{\ell-1,i,j})\right)\\ =&\sum_{j=1}^{\tilde{m}_{\ell,k}} \frac{\partial_{\tilde{p}}\phi_{\tilde{\boldsymbol{\theta}}_{\tilde{\boldsymbol{\theta}}}}(t_{\ell,k,j})}{\tilde{\lambda}_{\ell,k}(t_{\ell,k,j})} - \sum_{i=1}^{K_{\ell-1}}\sum_{j=1}^{m_{\ell-1,i}} \partial_{\tilde{p}}\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}_{\tilde{\boldsymbol{\theta}}}}(t_{\ell-1,i,j}),\end{aligned}$$

where

$$\partial_{\tilde{p}}\phi_{\tilde{\boldsymbol{\theta}}}(t) = \frac{\tilde{\beta}^{\tilde{\alpha}}}{\Gamma(\tilde{\alpha})}t^{\tilde{\alpha}-1}e^{-\tilde{\beta}t},\ \partial_{\tilde{p}}\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}}(t) = \frac{1}{\Gamma(\tilde{\alpha})}\gamma(\tilde{\alpha},\tilde{\beta}t).$$

The partial derivative of $\tilde{\text{llh}}$ $w.r.t$ $\tilde{\alpha}$ is

$$\partial_{\tilde{\alpha}}\tilde{\text{llh}} = \sum_{j=1}^{\tilde{m}_{\ell,k}} \frac{\partial_{\tilde{\alpha}}\phi_{\tilde{\boldsymbol{\theta}}}(t_{\ell,k,j})}{\tilde{\lambda}_{\ell,k}(t_{\ell,k,j})} - \sum_{i=1}^{K_{\ell-1}}\sum_{j=1}^{m_{\ell-1,i}} \partial_{\tilde{\alpha}}\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}}(t_{\ell-1,i,j}),$$

where

$$\begin{aligned}\partial_{\tilde{\alpha}}\phi_{\tilde{\boldsymbol{\theta}}}(t) =&\tilde{p}\frac{(\tilde{\beta}t)^{\tilde{\alpha}-1}\ln(\tilde{\beta}t)\Gamma(\tilde{\alpha}) - (\tilde{\beta}t)^{\tilde{\alpha}-1}\Psi(\tilde{\alpha})\Gamma(\tilde{\alpha})}{\Gamma^2(\tilde{\alpha})}\tilde{\beta}e^{-\tilde{\beta}t} = \tilde{p}(\tilde{\beta}t)^{\tilde{\alpha}-1}\frac{\ln(\tilde{\beta}t) - \Psi(\tilde{\alpha})}{\Gamma(\tilde{\alpha})}\tilde{\beta}e^{-\tilde{\beta}t},\\ \partial_{\tilde{\alpha}}\Phi_{\tilde{\boldsymbol{\theta}}}(t) =&\tilde{p}\left(-\frac{\Psi(\tilde{\alpha})}{\Gamma(\tilde{\alpha})}\gamma(\tilde{\alpha},\tilde{\beta}t) + \frac{1}{\Gamma(\tilde{\alpha})}\frac{\partial\gamma(\tilde{\alpha},\tilde{\beta}t)}{\partial\tilde{\alpha}}\right)\\ =&\tilde{p}\left(-\frac{\Psi(\tilde{\alpha})}{\Gamma(\tilde{\alpha})}\gamma(\tilde{\alpha},\tilde{\beta}t) + \frac{1}{\Gamma(\tilde{\alpha})}\frac{\partial(\Gamma(\tilde{\alpha}) - \Gamma(\tilde{\alpha},\tilde{\beta}t))}{\partial\tilde{\alpha}}\right)\\ =&\tilde{p}\Bigg(- \frac{\Psi(\tilde{\alpha})}{\Gamma(\tilde{\alpha})}\gamma(\tilde{\alpha},\tilde{\beta}t) + \frac{1}{\Gamma(\tilde{\alpha})}\cdot\Big(\Psi(\tilde{\alpha})\Gamma(\tilde{\alpha}) - \ln(\tilde{\beta}t)\Gamma(\tilde{\alpha},\tilde{\beta}t) - \tilde{\beta}t\cdot T(3,\tilde{\alpha},\tilde{\beta}t)\Big)\Bigg),\end{aligned}$$

and $T(m,s,x)$ is a special case of the Meijer G-function

$$T(m,s,x) = G_{m-1,m}^{m,0}\left(\left.\begin{matrix}\underbrace{0,0,\cdots,0}_{m-1}\\ \underbrace{s-1,-1,\cdots,-1}_{m}\end{matrix}\right| x\right).$$

However, it is expensive and numerically unstable to directly calculate Meijer G-function, so we use the first order finite difference to approximate the derivative.

The partial derivative of $\tilde{\text{llh}}$ $w.r.t$ $\tilde{\beta}$ is

$$\partial_{\tilde{\beta}}\tilde{\text{llh}} = \sum_{j=1}^{\tilde{m}_{\ell,k}} \frac{\partial_{\tilde{\beta}}\phi_{\tilde{\boldsymbol{\theta}}}(t_{\ell,k,j})}{\tilde{\lambda}_{\ell,k}(t_{\ell,k,j})} - \sum_{i=1}^{K_{\ell-1}}\sum_{j=1}^{m_{\ell-1,i}} \partial_{\tilde{\beta}}\tilde{\Phi}_{\tilde{\boldsymbol{\theta}}}(t_{\ell-1,i,j}),$$

where

$$\begin{aligned}\partial_{\tilde{\beta}}\phi_{\tilde{\boldsymbol{\theta}}}(t) &= \tilde{p}/\Gamma(\tilde{\alpha})t^{\tilde{\alpha}-1}(\tilde{\alpha}\tilde{\beta}^{\tilde{\alpha}-1}e^{-\tilde{\beta}t} - \tilde{\beta}^{\tilde{\alpha}}e^{-\tilde{\beta}t}t) = \tilde{p}/\Gamma(\tilde{\alpha})t^{\tilde{\alpha}-1}\tilde{\beta}^{\tilde{\alpha}-1}e^{-\tilde{\beta}t}(\tilde{\alpha} - \tilde{\beta}t)\ ,\\ \partial_{\tilde{\beta}}\Phi_{\tilde{\boldsymbol{\theta}}}(t) &= \tilde{p}\frac{1}{\Gamma(\tilde{\alpha})}\frac{\partial\gamma(\tilde{\alpha},\tilde{\beta}t)}{\partial\tilde{\beta}} = \tilde{p}\frac{1}{\Gamma(\tilde{\alpha})}(\tilde{\beta}t)^{\tilde{\alpha}-1}e^{-\tilde{\beta}t}\cdot t\ .\end{aligned}$$

We use softplus function to make sure the parameters are all positive.

C.2 Maximization and Derivatives *w.r.t* the Parameters for RPPs

The likelihood of the parameters for $\mathbf{Z}_n$ *w.r.t* the real events is

$$\text{llh} = \sum_{\ell=0}^{L}\sum_{k=1}^{K_\ell}\left(\sum_{j=1}^{m_{n,\ell,k}} \log \lambda_{n,\ell,k}(t_{n,\ell,k,j}) - \int_0^T \lambda_{n,\ell,k}(t)dt\right),$$

where

$$\int_0^T \lambda_{n,L,k}(t)dt = \mu_{n,k}T \ ,$$
$$\int_0^T \lambda_{n,\ell,k}(t)dt = \sum_{i=1}^{K_{\ell+1}}\sum_{j=1}^{m_{n,\ell+1,i}} \Phi_{\boldsymbol{\theta}_{(\ell+1,i)\to(\ell,k)}}(t_{n,\ell+1,i,j}) \text{ for } 0 \le \ell \le L-1 \ .$$

The maximizing value for $\mu_{n,L,k}$ is

$$\mu_{n,L,k} = \frac{m_{n,L,k}}{T}.$$

The functional forms for the derivatives of the other parameters of the kernel functions are the same as the forms for VPPs.

D FULL BAYESIAN INFERENCE

For simplicity, we use $\boldsymbol{\theta}$ to denote all of the hyperparameters. We need sample $\boldsymbol{\theta}$ from the posterior distribution

$$p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{z}) = \frac{p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta})}{p(\mathbf{x}, \mathbf{z})}.$$

Each time we uniformly select a hyperparameter θ from $\boldsymbol{\theta}$ and the acceptance ratio is

$$\mathcal{A} = \min(\alpha, 1),$$

where

$$\alpha = \frac{p(\theta' \mid \mathbf{x}, \mathbf{z})q(\theta' \to \theta)}{p(\theta \mid \mathbf{x}, \mathbf{z})q(\theta \to \theta')} = \frac{p(\theta')p(\mathbf{x}, \mathbf{z} \mid \theta')q(\theta' \to \theta)}{p(\theta)p(\mathbf{x}, \mathbf{z} \mid \theta)q(\theta \to \theta')},$$

θ' is the proposal, and θ is the current value.

We assume the prior distributions for each parameter is Gamma distribution, and use h to denote the shape, c to denote the scale, then

$$p(\theta) = \text{Gamma}(\theta; h, c),$$
$$q(\theta' \to \theta) = \text{Gamma}(\theta; h, \theta'/h),$$
$$q(\theta \to \theta') = \text{Gamma}(\theta'; h, \theta/h),$$

and

$$\begin{aligned}\alpha =& \frac{\cancel{1/(\Gamma(h)c^h)(\theta')^{h-1}} \exp(-\theta'/c) \cdot 1/\left(\cancel{\Gamma(h)}(\theta'/h)^h\right) \cancel{\theta^{h-1}} \exp(-\theta/(\theta'/h)) \cdot p(\mathbf{x}, \mathbf{z} \mid \theta')}{\cancel{1/(\Gamma(h)c^h)\theta^{h-1}} \exp(-\theta/c) \cdot 1/\left(\cancel{\Gamma(h)}(\theta/h)^h\right) \cancel{(\theta')^{h-1}} \exp(-\theta'/(\theta/h)) \cdot p(\mathbf{x}, \mathbf{z} \mid \theta)} \\ =& \frac{\theta^h \exp(-\theta'/c - \theta/(\theta'/h)) \cdot p(\mathbf{x}, \mathbf{z} \mid \theta')}{(\theta')^h \exp(-\theta/c - \theta'/(\theta/h)) \cdot p(\mathbf{x}, \mathbf{z} \mid \theta)}.\end{aligned}$$

E SPATIAL BIRTH-AND-DEATH ALGORITHM

We attempt to construct a spatial birth-death process to simulate the posterior distribution of hidden TPPs. We need to construct two proposals: birth and death. We use a birth to add an event and a death to delete an event.

Spatial birth-and-death (Preston, 1977) is a continuous-time Markov process. The detailed balance equation is

$$p(\mathbf{z}_{\ell,k} \mid \mathbf{x})b(\mathbf{z}_{\ell,k}, t) = p(\mathbf{z}_{\ell,k} \cup \{t\} \mid \mathbf{x})d(\mathbf{z}_{\ell,k} \cup \{t\}, t), \tag{11}$$

where $b(\mathbf{z}_{\ell,k}, t)$ is the birth rate to add a new event at time t to the current hidden events set $\mathbf{z}_{\ell,k}$ and $d(\mathbf{z}_{\ell,k}, t)$ is the death rate for removing an event with time t.

A common way to determine the birth rate $b(\mathbf{z}_{\ell,k}, t)$ is to make it proportional to the PCIF as in Eq. 2 and the death rate to be a constant number (Ripley, 1977; Baddeley & Møller, 1989; Møller, 1989). However, it would be very hard to calculate the total birth rate exactly, as it would require an integral of the product terms in Eq. 2. If, instead, we try to find an upper bound for the PCIF and then use thinning to get the samples for birth process, it would have far too many attempted jumps rejected for the birth stage.

Similar to Lieshout et al. (2002), we can make the birth rate to be

$$b(\mathbf{z}_{\ell,k}, t) = \tilde{\mu}_{\ell,k} + \sum_{i=1}^{K_{\ell-1}} \sum_{t_{\ell-1,i,j} > t} \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(t_{\ell-1,i,j} - t). \tag{12}$$

The only difference from Lieshout et al. (2002) is we divide the birth rate by $\lambda_{\ell,k}(\cdot)$ and the birth rate not only depends on the evidence, but also depends on the posterior samples.

To satisfy the detailed balance Eq. 11, the death rate $d(\mathbf{z}_{\ell,k} \cup t, t)$ needs to be

$$\begin{aligned} d(\mathbf{z}_{\ell,k} \cup t, t) =& \frac{b(\mathbf{z}_{\ell,k}, t)}{\lambda_{\mathcal{P};\ell,k}(t)} \\ =& \frac{\tilde{\mu}_{\ell,k} + \sum_{i=1}^{K_{\ell-1}} \sum_{t_{\ell-1,i,j} > t} \tilde{\phi}_{\tilde{\boldsymbol{\theta}}_{(\ell-1,i)\to(\ell,k)}}(t_{\ell-1,i,j} - t)}{\lambda_{\ell,k}(t) \prod_{i=1}^{K_{\ell-1}} \left(\exp\left(-\Phi_{(\ell,k)\to(\ell-1,i)}(T-t)\right) \prod_{t_{\ell-1,i,j} > t} \frac{\lambda'_{\ell-1,i}(t_{\ell-1,i,j}, t)}{\lambda_{\ell-1,i}(t_{\ell-1,i,j})} \right)}, \end{aligned} \tag{13}$$

where $\lambda'_{\ell-1,i}(x, y) = \lambda_{\ell-1,i}(x) + \phi_{\theta_{(\ell,k)\to(\ell-1,i)}}(x - y)$.

The total birth rate is

$$\beta(\mathbf{z}) = \sum_{\ell,k} \int_0^T b(\mathbf{z}_{\ell,k}, t)dt = \sum_{\ell,k} \left(\tilde{\mu}_{\ell,k} T + \sum_{i=1}^{K_{\ell-1}} \sum_{t_{\ell-1,i,j}} \tilde{\Phi}_{\tilde{\theta}_{(\ell-1,i)\to(\ell,k)}}(t_{\ell-1,i,j}) \right), \tag{14}$$

and the total death rate is

$$\delta(\mathbf{z}) = \sum_{\ell,k} \left(\sum_j d(\mathbf{z}_{\ell,k}, t_{\ell,k,j}) \right).$$

Open problem: *Does the SB&D with the birth rate as in Eq. 12 and the death rate as in Eq. 13 converge to an invariant distribution?*

The SB&D given in Lieshout et al. (2002) is guaranteed to converge to an invariant distribution as it satisfies a sufficient condition that the total birth rate is bounded from above and the total death rate is bounded from below. However, the total birth rate in Eq. 14 is finite but not bounded in our case, which violates the sufficient condition satisfied in Lieshout et al. (2002). Please refer to Møller & Waagepetersen (2003) for more details.

Therefore, we cannot guarantee this method will converge, and instead use the MCMC method in the main section of this paper.

F INFERENCE ALGORITHM

Algorithm 1 MCEM for DN-SPs

```
Input: data $\{\mathbf{x}_n\}$, model $\mathcal{M}$
Initialization: base rates $\{\boldsymbol{\mu}_n\}$, kernel parameters $\boldsymbol{\theta}$, virtual base rates $\{\tilde{\boldsymbol{\mu}}_n\}$, virtual kernel parameters $\tilde{\boldsymbol{\theta}}$,
initial states for Markov chains.
repeat
    $\mathbf{z}_n^{(1)}, \dots, \mathbf{z}_n^{(\mathcal{S})} \sim p(\mathbf{Z}_n \mid \mathbf{x}_n; \boldsymbol{\mu}_n, \boldsymbol{\theta})$ by MCMC
    Maximize base rates based on Eq. 8
    Use Adam to optimize Eqs. 9, 5, and 6
    Record $\mathbf{z}_n^{(\mathcal{S})}$ as the initial state for the Markov chain
until the expected log-likelihood in Eq. 7 get converged
```

G EXPERIMENTS DETAILS

G.1 Datasets

Retweets (Zhao et al., 2015) The retweets dataset collected sequences of tweets streams. Each sequence contains the times and types for some follow-up retweets. We use the same dataset as used in NHP. The retweets are grouped into three types (small, medium and large) according to the number of followers of the users who owned the retweets.

Earthquakes (NCEDC, 2014; BDSN, 2014; HRSN, 2014; BARD, 2014) We collected the times and magnitudes for earthquakes between 01/01/2014 00:00:00 and 01/01/2020 00:00:00 in the region spanning between 34.5° and 43.2° latitude and between −126.00° and −117.76° longitude. If the magnitude of a earthquake is smaller than 1, we classify it as a small earthquake, otherwise a large earthquake.

Homicides (COC) This dataset contains the times for homicides that occurred at five contiguous districts (007-011) with the most homicides in Chicago from 01/01/2001 00:00:00 to 01/01/2020 00:00:00. The type for an event is the district where the homicide occurred. The terms of use can be found at `https://www.chicago.gov/city/en/narr/foia/data_disclaimer.html`.

The number of types, the number of total events and the number of sequences for each dataset are summarized in Table 3.

Table 3: Datasets Statistics

Datasets	# types	# events	# sequences		
			train	validation	test
Retweets	3	2610102	20000	2000	2000
Earthquakes	2	156743	209	53	53
Homicides	5	3956	6	2	2

G.2 Training and Testing

The probabilities of the moves for *Move 1*, *Move 2*, and *Move 3* are 0.2, 0.6, and 0.2 respectively. We train the models using Algorithm 1 to get the parameters of the kernel and virtual kernel functions.

During testing, the parameters of the kernel functions and virtual kernel functions are fixed. We update the base rates according to Eqs. 8 and 5. The posterior samples for RPPs are collected to calculate the expectation of the log-likelihood per event. For each sequence in the evidence, there is an event at the end. To better capture the last event, we assume there is a synthetic real event for each hidden TPP at the end. This synthetic real event is only involved with the calculation of the intensity functions for the VPPs.

G.3 Likelihood

For a neural network, the CIF has a parametric form $\lambda_{f_\Theta(\mathbf{x})}(t)$ determined by a function f_Θ and the data $\mathbf{x}$. The parameters Θ for f_Θ are learned during training. For testing, the CIF $\lambda_{f_\Theta(\mathbf{x})}(t)$ is determined by the testing data $\mathbf{x}$ and f_Θ. The log-likelihood is $\log P(\mathbf{x} \mid \lambda_{f_\Theta(\mathbf{x})}(t))$.

For a DN-SP, Θ, the parameters for the kernels, are learned during training, and fully specify $P(\mathbf{Z} \mid \mathbf{x})$. We use a random function f_Θ to determine the CIF. The distribution of $f_\Theta(\mathbf{x})$ is fully determined by $P(\mathbf{Z} \mid \mathbf{x})$, and thus is fully determined by Θ and $\mathbf{x}$. We calculate the expected value of the log-likelihood $\mathbb{E}_{f_\Theta(\mathbf{x})}[\log P(\mathbf{x} \mid \lambda_{f_\Theta(\mathbf{x})}(t))] = \mathbb{E}_{\mathbf{Z}\sim P(\mathbf{Z}|\mathbf{x})}[\log P(\mathbf{x} \mid \mathbf{Z})]$ to compare with the baselines.

We do not compare the marginal log-likelihood $\log \mathbb{E}_{\mathbf{Z}\sim P(\mathbf{Z})}[P(\mathbf{x} \mid \mathbf{Z})]$ of our model with the log-likelihood of the neural-network-based models because there are no efficient methods for estimating this expectation, as "forward sampling" with $P(\mathbf{Z})$ is prohibitively inefficient and samples from $P(\mathbf{Z} \mid \mathbf{x})$, which our method is designed to generate, cannot be used (as the importance sampling ratio weight cannot be calculated).

Similarly, it is difficult to calculate the marginal likelihood for MTPP and the log-likelihood for MTPP is the evidence lower bound per event.

To address the difference in likelihood calculations across methods, we also provide two additional metrics: predictive accuracy and root mean squared error, which are handled in exactly the same fashion for all methods (see Appx. G.4).

G.4 Prediction

For prediction, each method predicts the time and type of the next event, conditioned on all events prior to it. This process is repeated for every event in the testing data (slowly increasing the conditioning set for each new prediction).

Instead of calculating an infinite integral as in NHP and SAHP, or using an additional layer of a neural network as in THP, we simply do forward sampling to predict the time and type for the next future event, as we have an explicit formula for the intensity.

We predict the time for the next future event from the beginning to the end. After we get to the convergence of our Markov chain, we draw the samples for the next future event $e_i = (t_i, k_i)$ conditional on the current state of the Markov chain. Suppose the samples for the next future event e_i conditional on the history $\mathcal{H}_{i-1} = \{(t_1, k_1), (t_2, k_2), (t_3, k_3), \cdots, (t_{i-1}, k_{i-1})\}$ are $(t_i^1, k_i^1), (t_i^2, k_i^2), \ldots, (t_i^{\mathcal{S}}, k_i^{\mathcal{S}})$, where $\{t_i^j\}_{j=1}^{\mathcal{S}}$ are the times of the samples for the future event e_i and $\{k_i^j\}_{j=1}^{\mathcal{S}}$ are the types of the future event e_i. The prediction for the future event time is $\hat{t}_i = \frac{1}{\mathcal{S}} \sum_{j=1}^{\mathcal{S}} t_i^j$ and the type prediction is $\hat{k}_i = \arg\max_{k\in\{1,\ldots,K_0\}} \sum_{j=1}^{\mathcal{S}} \mathbb{1}_k(k_i^j)$, where $\mathbb{1}_k(k_i^j)$ is the indicator function which is equal to 1 *iff* $k = k_i^j$. Then we calculate the root mean squared error (RMSE) for the time prediction and accuracy for the type prediction.

After the prediction for e_i, we use the current state of the Markov chain for $\mathcal{H}_{i-1}$ as the initial state for the Markov chain for $\mathcal{H}_i$ and run the Markov chain until convergence. Then we predict the time and type for the event e_{i+1} conditional on $\mathcal{H}_i$ the same as how we predict the event e_i.

G.5 Time Complexity

The time complexities for flip, swap, and resampling for a single event in MCMC are $\mathcal{O}(1)$, constant irrespective of any values (amount of data, parameter values, length of time, etc). A full analysis of the MCMC time complexity would require bounding the number of steps necessary (by mixing time arguments, usually). As with almost all other non-trivial MCMC inference methods, we do not have such a bound.

Intuitively, a model with more hidden layers has much larger search space for the events from a posterior distribution and it should take more time to get fully mixed. And we have observed experimental results to support our intuition. As shown in Table 4, the 2-hidden models take more time to get fully mixed than 1-hidden models.

The experiments are trained on a cluster with multiple CPU cores. The number of CPU cores ranges from 8 to 64.

Table 4: Training Time in Hours

Datasets	Model	Time
Retweets	1-hidden	12.0
	2-hidden	108.5
Earthquakes	1-hidden	0.8
	2-hidden	22.3
Homicides	1-hidden	1.1
	2-hidden	14.0

H LIMITATIONS

- The current gamma kernel function is simple, restricting the ability to model more complex data. In the future, we can try to design more complex kernels.
- We only have experiments for temporal point processes and no edge effects are taken into account here. However, a SPP is not constrained to one dimensional space. SPPs can exist in Euclidean spaces with any finite dimensions and non-Euclidean spaces, like spheres and torus. Our next step would be to apply our method to more general SPPs.

I NEGATIVE SOCIETAL IMPACTS

Our model may be used to predict people's behavior from the data tracked by smart phones or other wearable devices. Such predictions regarding when people are away from their residence, visiting a medical facility, or at work could be used for theft, medical rate increases, or overbearing workplace monitoring. Anonymization and privacy-preserving algorithms could be used to mitigate such risks.