FAST HIGH-DIMENSIONAL TEMPORAL POINT PROCESSES WITH APPLICATIONS

by

Ali Caner Türkmen

B.S., Industrial Engineering, Boğaziçi University, 2010

M.S., Software Engineering, Boğaziçi University, 2014

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Computer Engineering
Boğaziçi University
2020

—to my family,

and in fond memory of Yılmaz Can Akbulut

who was always fascinated by “data science”

ACKNOWLEDGEMENTS

I must begin by thanking my advisor, Prof. Taylan Cemgil, not only for exceptional mentorship and supervision, but also for extending his warm friendship over the course of my studies. It has been, and will continue to be, a true privilege. I would also like to thank Profs. Refik Güllü and Sinan Yıldırım for their invaluable guidance over the course of the last three years.

I was fortunate to work with an extraordinary group of researchers at Amazon through two internships, where parts of this thesis were written. I would like to thank Drs. Bernie Wang, David Salinas, Tim Januschowski, Prof. Alex Smola, members of the ML Forecasting team and AWS AI Labs.

This work would not be possible without the contributions of my chief partner-in-crime Gökhan Çapan, İlker Gündoğdu, Melih Barsbey, and many other Bayesians of PiLab past and present. Cheering me on were a great group of friends with whom I shared the many joys and woes of life. A shout out to Ömer, Yiğit, Gülçin, Kartal, Can, Mahmut, Kuzey, Göktuğ, Buğra, their families and other members of the extended BU IE crowd whose companionship have been my great fortune.

I feel blessed to be a part of a loving family, my sister Cansu—on whose couch I crashed for months and who still chose to pursue a graduate degree, my mother Nuray—whose endless love and patience took me through every school I went to since kindergarten, and my father Ahmet Türkmen—who taught me (apart from writing a DOS batch file at age five) that a smooth sea never made a skilled sailor. To my grandparents Ayhan—for taking care of me through too long a university career, and Macit Erarslan—who served as the single greatest motivator for this work. Ultimately, this thesis is a tribute to them.

I would like to acknowledge the support of the Scientific and Technological Re-

search Council of Turkey (TÜBİTAK), who partially supported this thesis under Grant No. 116E580.

ABSTRACT

FAST HIGH-DIMENSIONAL TEMPORAL POINT PROCESSES WITH APPLICATIONS

Large sets of continuous-time discrete event streams are often in the focus of seismology, neuroscience, finance, behavioral science among other scientific and engineering disciplines. In this work, we explore a set of novel models and algorithms to learn from such data at scale, in the presence of a large number of events and event types. First, we develop two algorithms for estimating high-dimensional multivariate Hawkes processes with a low-rank parameterization. The first approach leverages a novel connection to nonnegative matrix factorization, which we use to propose a stochastic gradient descent algorithm. We then demonstrate, via a moment-based approach, that we can reduce the parameter estimation problem to a single low-rank approximation. Notably, both approaches require only a few scans of the data, feature well-known matrix decompositions as subroutines, and yield fast parameter estimation. We also propose global-local temporal point processes (TPP), multidimensional TPP models that model self- and mutual-excitation patterns at different scales of time. One such model, FastPoint, relies on deep recurrent neural networks to approximate the mutual excitation pattern, and results in several orders of magnitude faster learning. Global-local TPPs also allow for substantially faster sequential Monte Carlo sampling, greatly accelerating the current state of the art in simulating temporal point patterns. Finally, we propose a novel application area for TPPs, applying ideas from renewal processes and deep learning to intermittent demand forecasting. Our contributions aim to remove both of the main challenges—scalable learning and inference—facing the adoption of high-dimensional TPP models.

ÖZET

HIZLI YÜKSEK BOYUTLU ZAMANSAL NOKTA SÜREÇLERİ VE UYGULAMALARI

Büyük ölçekli sürekli-zamanlı ayrık olay akışları deprembilim, sinirbilim, finans, davranış bilimi ve birçok diğer bilim ve mühendislik disiplininde sıkça odak konusudur. Bu çalışmada, bu tür verilerden ölçeklenebilir bir şekilde—yüksek sayıda olay verisi ve olay türü altında—öğrenme sağlayacak yeni bir dizi model ve algoritma incelenmektedir. Öncelikle yüksek-boyutlu çok-değişkenli Hawkes sürecinde düşük-ranklı parametre kestirimi için iki algoritma önerilmiştir. İlk olarak, negatif olmayan matris ayrışımı ile yeni bir bağlantı üzerine kurulu bir rassal eğim iniş algoritması verilmiştir. İkinci olarak, moment-tabanlı bir yaklaşımla kestirim probleminin tek bir düşük ranklı ayrışıma indirgenebileceği gösterilmektedir. İki yaklaşımda da verinin birkaç kez taranması yeterlidir, yaygınca bilinen matris ayrışımları alt yordam olarak kullanılmaktadır, ve hızlı ve yüksek başarımlı parametre kestirimi sağlanmaktadır. Ayrıca, öz-uyarım ve türler arası uyarım davranışlarını farklı zaman ölçeklerinde tarif eden, global-yerel zamansal nokta süreçleri (ZNS) adıyla yeni bir ZNS sınıfı tanımlanmıştır. Bu sınıfın bir örneği, FastPoint, türler arası uyarım örüntülerini bir derin özyineli sinirsel ağ ile kestirerek eşdeğerlerinden yüzlerce kat daha hızlı öğrenme sağlamaktadır. Global-yerel ZNS modelleri, sıralı Monte Karlo yöntemleri ile çok daha hızlı örnek çekilmesini sağlamakta ve bilinen nokta süreci benzetimi yöntemlerinin tümünden daha verimli sonuç üretmektedir. Son olarak, ZNS modellerinin uygulama alanları, seyrek talep tahmini problemine yenileme süreçleri ve derin öğrenme yöntemlerinin uygulanmasıyla genişletilmiştir. Çalışmamız, yüksek boyutlu ZNS modellerinin yaygın kullanımının önündeki iki büyük engeli—öğrenme ve çıkarımı ölçeklemenin zorluğunu—gidermeyi amaçlamaktadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

A, B, C	Sets
a, b, γ, δ	Scalars
$\mathbf{a}, \mathbf{b}, \mathbf{v}, \lambda$	Vectors
$\mathbf{A}, \mathbf{B}, \mathbf{\Gamma}, \mathbf{\Lambda}$	Matrices
d	A positive integer—the number of dimensions (marks)
$\mathcal{E}$	Exponential distribution
g	Delay density of a Hawkes process
$\mathcal{G}$	Geometric distribution
$\mathcal{H}_t$	A filtration (up to time t)
$\mathbf{I}$	The identity matrix
$\mathcal{I}_k$	Index set of events of mark k, $\mathcal{I}_k \subset \mathbb{N}$
i, j, k, l, m, n, d, r	Positive integers reserved for indices or set cardinalities
$\mathbf{L}$	A graph Laplacian
$M(t), M_t$	Stochastic process M, $t \in \mathbb{R}_{\geq 0}$
n	A positive integer—to denote the number of events
$N(t), N_t$	Counting process N, $t \in \mathbb{R}_{\geq 0}$
$\mathcal{N}$	Gaussian distribution
$\mathcal{NB}$	Negative binomial distribution
$\mathcal{PO}$	Poisson distribution
t	A positive variable—to denote “time”
$\mathcal{U}$	Uniform distribution
$Y(t), Y_t$	Martingale Y, $t \in \mathbb{R}_{\geq 0}$
β	Hawkes process delay density rate
Θ	A set of model parameters
Π	A random point set
$\Pi_{(s,t]}$	A random (a.s. finite) point set over the interval $(s, t]$
$\Pi_{(s,t]}^{(k)}$	A random set of points of mark k over the interval $(s, t]$

ϕ	Hawkes process infectivity (triggering) kernel
$\mathbf{\Phi}$	Hawkes process infectivity (triggering) matrix
$\mathbf{\Psi}$	Hawkes process inverse kernel

$\mathbb{P}\{A\}$	Probability measure (of event A)
$\mathbb{E}\left[X\right]$	Mathematical expectation of a random variable X
$\mathbb{V}[X]$	Variance of a random variable X
$\lvert A\rvert$	Cardinality (of set A)
$[\![A]\!]$	Indicator function of event A (Iverson bracket)
$[n]$	Set of positive integers up to n, $\{1, 2, \cdots, n\}$
$[\mathbf{A}]_{ij} = a_{ij}$	The i-th row, j-th column of matrix $\mathbf{A}$
$\lVert\mathbf{A}\rVert_p$	p-norm (operator norm), $p = 2$ by default
$\lVert\mathbf{A}\rVert_F$	Frobenius norm
$\lVert\mathbf{v}\rVert_p$	p-norm of vector $\mathbf{v}$, $p = 2$ by default
$[\mathbf{v}]_i = v_i$	The i-th element of vector $\mathbf{v}$
$\triangleq$	Equality by definition
$\mathbf{A}^T$	Matrix transpose
$\mathbf{A} \geq 0$	Matrix $\mathbf{A}$ is elementwise greater than or equal to 0
$\mathbf{A} \succeq 0$	Matrix $\mathbf{A}$ is positive semidefinite, (if $\succ$, positive definite)

LIST OF ACRONYMS/ABBREVIATIONS

a.s.	almost surely
AR	Autoregressive
cdf	Cumulative Distribution Function
DTRP	Discrete-time Renewal Process
EM	Expectation Maximization
EVD	Eigenvalue Decomposition
EWMA	Exponentially Weighted Moving Average
GAN	Generative Adversarial Network
GEVD	Generalized Eigenvalue Decomposition
GLMTPP	Global-Local Multidimensional Temporal Point Process
GPU	Graphical Processing Unit
HP	Hawkes Process
IDF	Intermittent Demand Forecasting
IMA	Integrated Moving Average
KL	Kullback-Leibler (Divergence)
LSTM	Long Short-Term Memory
MHP	Multidimensional Hawkes Process
MLP	Multilayer Perceptron
MM	Majorization Minimization
MRP	Markov Renewal Process
MTPP	Multidimensional (Multivariate) Temporal Point Process
NMF	Nonnegative Matrix Factorization
NN	Neural Network
ODE	Ordinary Differential Equation
pd	Positive Definite
pdf	Probability Density Function
psd	Positive Semi-definite
RMTPP	Recurrent Marked Temporal Point Process

RNN	Recurrent Neural Network
SDE	Stochastic Differential Equation
SGD	Stochastic Gradient Descent
SMC	Sequential Monte Carlo
SSM	State Space Model
SSOE	Single Source of Error
TPP	Temporal Point Process

1. INTRODUCTION

1.1. Motivation

Many real-world phenomena can be represented as instantaneous events in continuous time. Often, one is concerned with making sense of the underlying process that gives rise to these events, usually when many records belonging to multiple types of events are available. We give some examples in the next few paragraphs.

Much of quantitative financial analysis focuses on price movements in aggregated frames of time, and analysts build models of how these time series move in sequence. However, in modeling actual market dynamics that jointly give rise to prices, the artifact of interest is the so-called *trade-and-quote* data that contains records of billions of buy/sell orders belonging to thousands of assets, happening in continuous time [1–3]. For example, see the top row of Figure 1.1, where the timestamps of price-shifting trades in an interbank currency market are illustrated as points in time [4].

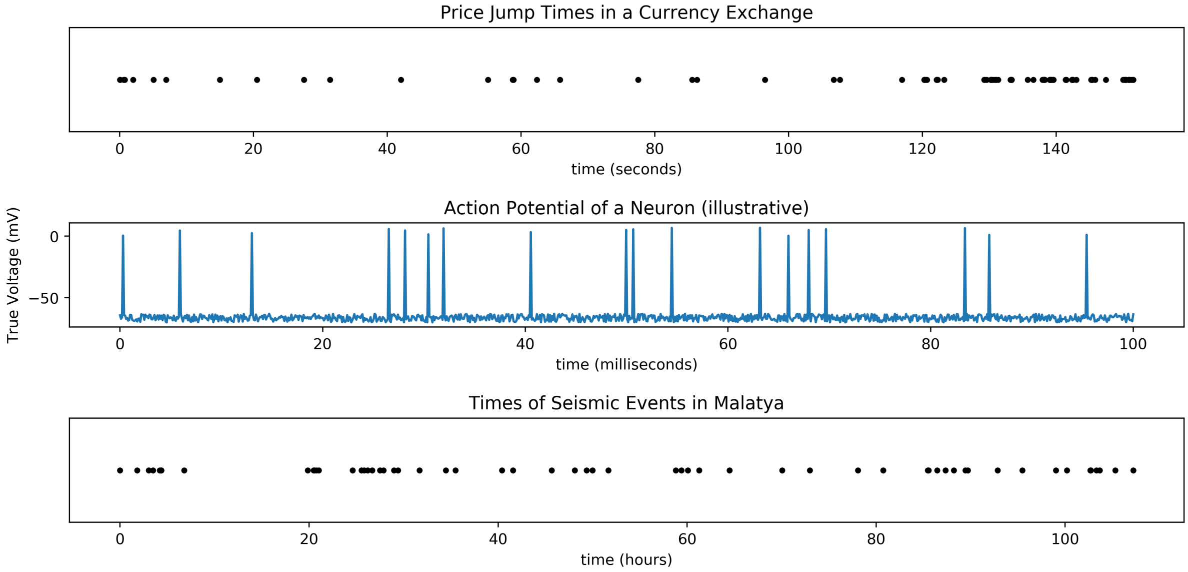


Figure 1.1. Examples of point processes.

The circuitry that commands human behavior is a neuronal network of 10^{11} cells, in constant communication with each other. Neurons “fire” asynchronously, generating “spikes,” as they encode and convey information downstream. The middle row of Figure 1.1 illustrates voltage readings with spikes on a neuronal membrane. Even for model microorganisms, understanding this circuitry requires inference over millions of such impulses, all observed in continuous time, across thousands of different neurons. Similarly to finance, understanding the most granular level of data demands interpretation of continuous time events— “spike trains”—across many entities that collectively amass enormous data sets [5–9].

Increasingly, social media is a key source of data for studies of how people interact and influence each other. Identifying how information, news, and opinions disseminate through online social media can require many forms of analysis such as looking at statistics of the network over time, or carrying out a static analysis of the social graph structure. Again, however, zooming in and taking the full data into account requires that one is able to model individual social interactions, happening asynchronously, across millions of entities, and at web scale [10, 11].

Finally, accurate models of seismic systems account for relationships of individual events across time and types (*e.g.*, the fault lines they occurred on). In the last row of Figure 1.1 earthquakes in and around Malatya, Turkey are shown with their timestamps on the x-axis [12].

In this work, we deal with models aimed at learning from such data. Specifically, we build stochastic models for *discrete events*—events that occur instantaneously as a point in continuous time.

1.2. Challenges Addressed

We seek to understand the dynamics of large systems where individual phenomena of interest occur *asynchronously*, *i.e.*, they are best cast in a continuous time formalism.

We will see that these systems are best modeled with *temporal point processes* (TPP), models of random point configurations on the real line—interpreted as time. Starting from elementary building blocks in modern machine learning and stochastic process literatures, we will explore several directions in building new models and inference methods.

Our first challenge was already highlighted in the previous section. Taking full account of the granular temporal data instead of "aggregating" time, the scale of data to be processed multiplies by orders of magnitude, greatly exacerbating the computational challenge. As such, our work will propose approaches where scalability in the number of events is a key concern.

We will specifically focus on cases where, beyond the continuous time formalism, one is faced with data where events are endowed with *marks*, *i.e.*, additional features. In our examples, we referred to spikes of different neurons or buy/sell orders on multiple assets. Indeed, a common problem is scaling TPP inference when events can be identified as one of too many such entities. Especially when relationships between these entities are non-trivial, learning is greatly hindered by the computational requirement, as is sampling from learned models for predictive inference. This problem, also referred to as *high-dimensionality*, is the main issue which our models and inference methods address.

Having introduced some of the key concerns of our work that appear in the title, let us focus on the two remaining claims. Often, applications dictate that inference occurs in the same scale of time in which observations are made, *e.g.*, in the stock market. As such, despite the large scale of data at hand, the importance of having fast computational routines for processing point data is paramount. Here, we work towards "fast" high-dimensional TPP algorithms. Finally, we will explore experiments in a wide span of potential application domains, ending the final chapter with a rather unusual application to retail demand forecasting.

1.3. Objectives and Contributions

The initial research direction of this study was focused on probabilistic learning and inference algorithms for high-frequency finance, just as in the beginning of the previous section. Although TPP ideas had been explored in-depth in quantitative finance, many of the techniques developed in this literature were applicable to common application domains of data mining and machine learning. We also noted, in the light of a series of recent papers that started in 2017, that many ideas from machine learning could be applied in contexts traditionally addressed with TPPs. As such, our research was greatly affected by recent developments and changed tack several times. Finally, however, it resulted in several distinct contributions in the intersection of machine learning, TPPs, and applications. Our contributions can be summarized as follows.

(i) We explore accelerated learning of high-dimensional multivariate mutually-exciting point processes, or multivariate Hawkes processes (MHP). Specifically, we explore a low rank Hawkes process (HP) where we enforce low-rankness on the infectivity matrix. We show that this model is naturally related to nonnegative matrix factorization (NMF), which leads to a set of fast learning algorithms for general low-rank MHP.

(ii) We propose a new algorithm for learning a low-rank MHP with a symmetric kernel. We show that a pass-efficient and fast learning algorithm can be attained by invoking the moment-matching approach to parameter estimation. Our algorithm requires a single scan of the data set, and reduces parameter estimation to a symmetric low-rank approximation problem. Interpreting the MHP kernel as parameterizing network structure that governs how different marks excite further events from each other, we show that our approach is closely linked to spectral clustering, and results in a natural way to cluster large sets of asynchronous event streams.

(iii) We propose a deep learning-driven approach to high-dimensional TPP learning. Our model, FastPoint, leverages observations made in recent papers that fuse deep learning and TPPs. However, instead of approximating an intensity function directly with a recurrent neural network, we use neural networks to approximate mutual-excitation structures while leaving "local" TPPs to be simpler processes. This is the first application of a "global-local" time series modeling approach to temporal point processes, and results in two orders of magnitude faster learning and improved predictive accuracy.

(iv) We focus on a largely underexplored area in TPP research, fast and scalable simulation (sampling). To that end, we explore sequential Monte Carlo (SMC) in the context of multivariate TPPs. Apart from being one of the first to apply SMC in the backdrop of TPP modeling, we show for the first time that it results in far superior speeds of inference while keeping the quality of predictions (*i.e.*, the *effective sample size*) constant. We observe that certain SMC routines are naturally parallelizable, making predictive inference significantly simpler—conceptually and computationally—than the only approach in sampling-based inference in TPPs, Ogata's thinning method.

(v) We finally focus on a novel application domain for TPPs: retail demand forecasting. Inspired by neural TPPs, we propose several models to tackle the "intermittent demand forecasting" problem studied in the forecasting and inventory control literature. Specifically, we show that discrete analogues of self-modulated renewal processes, which we call discrete-time deep renewal processes, outperform previously proposed probabilistic models. Moreover, we also explore scenarios where "timestamps" are available for each demand, casting the problem in a continuous time formalism for the first time.

(vi) A number of open-source libraries [12] or contributions [13,14] have resulted from our work, expanding the somewhat limited availability of tools for learning with TPPs.

1.4. Organization of The Thesis

In Chapter 2, we start by giving general background on TPPs and their variants, the principal theoretical devices that we use throughout this thesis. Namely, we set up the notation through an introduction of some of the elementary TPP theory, with

Poisson processes and renewal processes. We then introduce Hawkes processes (HP), one of the most popular models for modeling temporal interdependencies of individual occurrences, and its multidimensional variant, the MHP. We introduce general notation for self-modulating processes, and moment (cumulant) representations of general weakly stationary multidimensional TPPs (MTPP). We finish by reviewing the latest advances in the machine learning literature, giving a brief introduction to deep learning and a review of neurally-modulated TPPs.

In Chapter 3, we focus on low-rank MHPs. After introducing previous approaches in the same direction, we draw links between low-rank factorization methods, graph clustering approaches, and MHPs.

In Chapter 4, we introduce FastPoint, and global-local TPPs in general. Here we also explore predictive inference in MTPPs, and work on how SMCs apply both for TPPs in general and in the specific context of FastPoint. We test our model and algorithm on several popular TPP data sets.

In Chapter 5, we apply ideas from earlier chapters to intermittent demand forecasting. To that end, we formalize two new classes of TPP models based on deep learning, and give experimental results on widely-used benchmark data sets. We conclude this thesis, discussing potential next steps, in Chapter 6.

2. BACKGROUND

This chapter aims to equip the reader with a general understanding of the theory on which our contributions build. Specifically, we aim to give a tutorial introduction to TPPs, at a level that suffices only for introducing the rest of the material. We then introduce HPs, discuss some common properties and uses for them. Finally, we introduce deep (or "neurally-modulated") TPPs, recent advancements in the machine learning community.

2.1. Temporal Point Processes

Stochastic processes are collections of random variables $\{X_t\}$ where t is some suitable index set [15, 16]. Often, the index set in which t takes values obeys a total ordering, and is interpreted as *time*. If the set of t is finite or countable, the process is referred to as a *discrete time* process. When X_t are indexed by the set of real numbers, the index set is interpreted as *continuous time*.

A point process, on the other hand, refers to random *configurations* of points on general measurable sets, with mild assumptions such as having finitely many points on bounded subsets. Spatial point processes, for example, determine distributions of point sets on a geodetic coordinate system. Determinantal point processes describe random configurations of "points" on finite-dimensional spaces.

A *temporal* point process (TPP) obeys both definitions. Defined naively, we will interpret TPPs as distributions of randomly allocated points on the real line [17]. The real line, in turn, will be used to model time (hence, *temporal*), and the points to model some *occurrence*,[1] such as the arrivals of customers to a queue or order arrivals to a stock market.

[1] We will interchangeably use the term *event*, not to be confused with the events of the underlying sigma field.

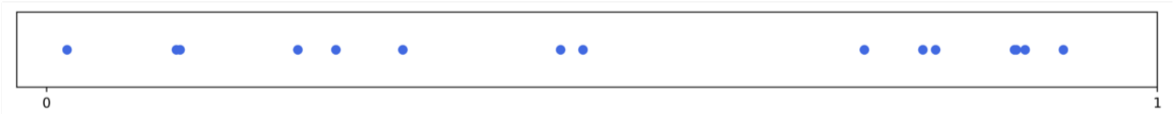


Figure 2.1. A draw from a TPP.

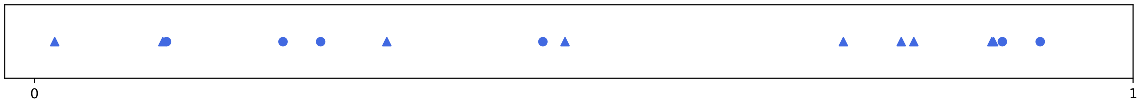


Figure 2.2. A draw from an MTPP.

Just as the random variate, *i.e.*, a draw, from a discrete probability distribution can be represented with an integer; a draw from a TPP can be represented as a finite set of points on bounded measurable subsets of $\mathbb{R}$. See Figure 2.1 for an illustration of one such draw. More concretely, a point configuration on a bounded set $(0, T]$ is a set of points $\{t_i\}_{i=1}^n$, $0 < t_1 < \cdots < t_n \leq T$. Without significant loss of generality, we will require that our point processes are *simple*. That is, we will require that strict inequality holds in the timestamps of individual events, *i.e.*, that no two points will land on each other a.s.

Events at times t_i may be equipped with *marks*, or features, $y_i \in \mathcal{Y}$. Then, a *marked* TPP determines the distribution over random point sets each identified as an ordered pair, *i.e.*, $\{(t_i, y_i)\}_{i=1}^n$. When $\mathcal{Y}$ is a finite set, indexed by $k \in [d]$, an equivalent formalism is to refer to *multidimensional* (or multivariate) TPP (MTPP)—*i.e.*, a set of d point processes that are not necessarily independent. For example, if k were to index users, an MTPP can be used to jointly model timestamps of their tweeting activity. $\mathcal{Y}$ can also be a general feature space, e.g. $\mathbb{R}_+$ to denote the size of limit orders on the stock market, or $\mathbb{R}^2$ to model coordinates of earthquakes.

We can now formally reiterate the focus of our work. Until the final chapter, we will be concerned with MTPPs in this thesis, focusing on learning and inference problems for large number of events n, and large number of marks (types) d.

There are several equivalent ways of characterizing a TPP. Before moving to an introduction of some popular TPPs, let us introduce an equivalent mathematical device. A *counting process* is a continuous-time stochastic process, denoted $\{N(t)\}_{t \in \mathbb{R}_+}$, or N_t, with two important properties: $N(t)$ take values in nonnegative integers, and that draws from the process are non-decreasing, *i.e.*, $N(t) \leq N(s)$ for all $t < s$. [2] Note that a counting process equivalently defines a TPP, and vice versa, since $N(t)$ could be used to represent the *count* of points until time t. As such, we will use the two notations, of random point sets and counting processes, interchangeably.

2.1.1. Poisson Process

We start with the "archetypal" TPP [17], the Poisson process [18].

Definition 2.1. *(Poisson process on the real line) A Poisson process on the real line is a random countable subset* $\Pi \subset \mathbb{R}$ *such that for any disjoint subsets* A_i *of* $\mathbb{R}$, $N(A_i) = |A_i \cap \Pi|$ *are independent random variables, elements of* Π *are distinct and* $N(A_i)$ *finite for bounded subsets* A_i*.*

Our definition captures the essence of Poisson processes, described before as complete randomness [18], or complete independence. It also omits one important property that is often included in *definitions* of Poisson processes,

Proposition 2.1. *If* Π *obeys a Poisson process on the real line, there exists a positive measure* μ *s.t.* $N(A_i) \sim \mathcal{PO}\left(\mu(A_i)\right)$.

We refer the reader to [18, Section 1.4] for an elegant proof of Proposition 2.1. Let $N(a, b]$ be the number of points that fall between a, b in a single draw from the Poisson

[2] We will use the two notations, $X(t)$ and X_t, interchangeably for stochastic processes in this work.

process.[3] It then follows that, $N(a,b]$ is a random variable, for which $\mathbb{E}\left[N(a,b]\right] = \mu((a,b])$.

Aside from μ, or the *mean measure* [18], let us define a measurable function, the *intensity* λ, such that

$$\mu(A) = \int_A \lambda(t)dt.$$

This function will be key to our discussions in the future and we will use it to unlock likelihood-based inference for general TPPs. For our case, the existence of the intensity is guaranteed by a simple application of the Radon-Nikodym theorem. The most important interpretation of the intensity is given through the fundamental theorem of calculus,

$$\lambda(t) = \lim_{\epsilon \downarrow 0} \epsilon^{-1} \mathbb{E}\left[N(t-\epsilon, t]\right]. \tag{2.1}$$

In other words, the intensity function, in a loose analogy with the density function in continuous probability distributions, specifies the probability that a point will land in the infinitesimal interval after time t. To understand this, note that our point processes are *simple*, i.e. almost surely, no two points coincide. Letting

$$dN(t) \triangleq \lim_{\epsilon \downarrow 0} \frac{N(t) - N(t-\epsilon)}{\epsilon},$$

we have that $dN(t) \in \{0,1\}$ by definition, and that $\lambda(t) = \mathbb{E}\left[dN(t)\right] = \mathbb{P}\{dN(t) = 1\}$.

Definition 2.2. *A* homogeneous[4] *Poisson process is a Poisson process where $\lambda(t) = \lambda$, given $\lambda \in \mathbb{R}, \lambda \geq 0$.*

In other words, homogeneous Poisson processes have constant intensity, imply-

[3] Similar to [17], we overload the $N(.)$ notation. when A is an arbitrary set, $N(A)$ refers to the number of points falling in this set. We write $N((a,b]) = N(a,b]$ removing the redundant parentheses. Finally, when $t \in \mathbb{R}$, $N(t) = N_t = N(0,t]$.

[4] Some authors, such as [17], prefer the term *stationary* or *uniform*

ing $\mu((a,b]) = \lambda|b-a|$. A property we will call *stationarity* holds trivially, and $N(a,b], N(a+\tau, b+\tau]$, are distributed identically $\forall \tau$.

Note that, by Proposition 2.1, for a homogeneous Poisson process we have

$$\mathbb{P}\{N(0,b]=0\} = \exp(-\lambda b), \tag{2.2}$$

which could also be interpreted as the probability that time of the first occurrence exceeds τ, or the *survivor function* $\mathbb{P}\{X_1 > \tau\}$, letting X_1 denotes the first occurrence time. Then, $\mathbb{P}\{X_1 \leq \tau\} = 1 - \exp(-\lambda\tau)$, *i.e.*, that the first occurrence time follows an exponential distribution. Through stationarity, the argument above applies to an arbitrary point on the real line, in that $\mathbb{P}\{N(0,b]=0\} = \mathbb{P}\{N(\tau, b+\tau]=0\}$. Finally, invoking the independence property, both lines of

$$\begin{aligned}\mathbb{P}\{N(0,b]=0\} &= \mathbb{P}\{N(0,b]=0|N(-\eta,0]=1\}\\ &= \lim_{\eta\downarrow 0}\mathbb{P}\{N(0,b]=0|N(-\eta,0]=1\}\end{aligned}$$

trivially hold. That is, the time to next occurrence is distributed exponentially, independently of when the previous occurrence was observed. This result is invariant as the previous occurrence gets arbitrarily close to the origin. It follows that, the so-called *interarrival times* of Poisson processes are distributed exponentially. Our argument also hints at the well-known *memoryless* property of exponential waiting times, namely that the time since last occurrence at an arbitrary point in the process yields no information about the time to the next occurrence. This is a direct consequence of the “complete randomness” of Poisson processes.

Finally, we must introduce the *likelihood* function for finite realizations of a Poisson process. For a finite realization $\{t_i\}_{i=1}^{N}$ of a homogeneous Poisson process on a bounded interval $(0, T]$, we can write

$$\begin{aligned}
L(\lambda|\{t_i\}) &= p(\{t_i\}|\lambda) \\
&= \lim_{\epsilon \downarrow 0} \left[\prod_i \mathbb{P}\{N(t_i, t_i + \epsilon] = 1\} \mathbb{P}\{N(t_{i-1} + \epsilon, t_i] = 0\} \right] \times \mathbb{P}\{N(t_N, T] = 0\} \\
&= \prod_i \lambda \left[\prod_i \exp(-\lambda(t_i - t_{i-1})) \right] \exp(-\lambda(T - t_N)) \\
&= \lambda^N \exp(-\lambda T),
\end{aligned}$$

where the first limit is by definition of the intensity function. The second limit is the probability that no other points are observed, and follows from the independence and stationarity of interarrival times. We will soon see that this intuition extends beyond homogeneous Poisson processes where the intensity varies across time, or even when the intensity is stochastic.

Let us also mention two key results for Poisson processes, the *superposition* and *mapping* properties. For both, we refer the reader to Kingman [18] for complete proofs.

Proposition 2.2. (Superposition property) *Given Poisson processes* Π_i*, random subsets of* $\mathbb{R}$ *with mean measures* μ_i*,* $\bigcup_i \Pi_i$ *is a Poisson process with mean measure* $\mu = \sum_i \mu_i$ *assuming* $\mu < \infty$*.*

Proof. First note that the sum of two Poisson random variables is also Poisson distributed. By induction on the number of Poisson random variables, we can write $S_N = \sum_{j=1}^N Y_j$ is distributed $\mathcal{PO}(\sum \mu_j)$. Now, noting that $(\{S_i \leq r\})_{i=1}^N$ is a decreasing sequence of sets (events) for fixed r, so that

$$\begin{aligned}\mathbb{P}\{S \leq r\} &= \lim_{j\uparrow\infty} \mathbb{P}\{S_j \leq r\} = \sum_{k=0}^{r} \lim_{j\uparrow\infty} \mathbb{P}\{S_j = k\} \\ &= \sum_{k=0}^{r} \lim_{j\uparrow\infty} \frac{1}{k!} \exp\left(-\sum_{\nu=1}^{j} \mu_\nu\right) \left(\sum_{\nu=1}^{j} \mu_\nu\right)^k \\ &= \sum_{k=0}^{r} \frac{1}{k!} \exp\left(-\sum_{\nu=1}^{\infty} \mu_\nu\right) \left(\sum_{\nu=1}^{\infty} \mu_\nu\right)^k\end{aligned}$$

proving that S is indeed distributed $\mathcal{PO}(\sum_{\nu=1}^{\infty} \mu_\nu)$.

For extending this argument to Poisson *processes*, assume for processes $\Pi_1, \Pi_2, \ldots$, the point counts on the interval A are denoted $N_1(A), N_2(A), \ldots$. Note that for the total count $N(A)$ of the point counts of $\bigcup \Pi_i$, we have that $N(A) = \sum_i N_i(A)$. This is not a simple result, although it is intuitive that, almost surely, no two points on different Poisson process will land on the same point in the union Poisson process (see the Disjointness lemma in [18]). Then, from our results above, $N(A) \sim \mathcal{PO}(\sum_i \mu_i(A))$. Noting that independence of disjoint sets holds trivially completes the proof. □

Our first result guarantees that the union of two Poisson processes is still a Poisson process. We now introduce the second important result, the mapping theorem. We give a restricted form of the more general theorem, and skip the proof.

Proposition 2.3. *(Mapping theorem) Let $\Pi \subset \mathbb{R}$ be a Poisson process, with finite mean measure μ, and $f : \mathbb{R} \to \mathbb{R}$ be a measurable function such that the induced mean measure $\mu^*(A) = \mu(f^{-1}(A))$ is non-atomic where $f^{-1}(A) \triangleq \{x \in \mathbb{R} | f(x) \in A\}$. Then $f(\Pi)$ is a Poisson process on $\mathbb{R}$ with μ^* as its mean measure.*

Intuitively, As long as the points in $f(\Pi)$ are distinct, it follows that $N(f^{-1}(A_1))$, $N(f^{-1}(A_2))$ are independent and Poisson distributed under the induced mean measure. However, the distinctness condition is not obvious. See [18, p. 18] for the complete proof. We must state one important result that derives from this theorem, namely that any inhomogeneous Poisson process can be generated from a homogeneous Poisson process with unit intensity.

Proposition 2.4. *Let* Π *be a Poisson process with unit intensity, that is* $\mu((a,b]) = |b-a|$. *Then letting* $f : \mathbb{R} \to \mathbb{R}$ *be a bijection,* $f^{-1}(t) = \int_0^t \lambda(x)dx$ *(*$\lambda(x) > 0, \forall x$*).* $f(\Pi)$ *is a non-homogeneous Poisson process with intensity* $\lambda(x)$*.*

Proof. We check the induced mean measure. Note that $\mu^*((a,b]) = \mu(f^{-1}(a,b])$ where $f^{-1}(a,b] = \{x|f(x) \in (a,b]\} = (f^{-1}(a), f^{-1}(b)]$ through monotonicity. Then, $\mu^*((a,b]) = f^{-1}(b) - f^{-1}(a) = \int_a^b \lambda(x)dx$. Noting that $\mu^*(\{x\}) = 0, \forall x$ by construction, we can invoke the mapping theorem to complete the proof. □

2.1.2. Renewal Processes

The Poisson process, in many regards, is an ideal process. Its main property, complete independence, is why it is the simplest process to work with, mathematically and computationally. In many cases, however, this assumption is too limited. For general TPPs, the complete independence property does not hold. In this section, we will look at a slightly more general TPP construction.

One way to characterize a family of TPPs is to require that the time between occurrences, interarrival times, are independent and identically distributed. Such processes are called *renewal processes.* One inspiration for their name is their use in modeling breakdowns in machine parts, those that have to be *renewed.* Here, the inherent assumption is that the "lifetime" of the part replaced is identically distributed.

Note that the Poisson process is a renewal process. Beyond having i.i.d. interarrival times, those times obey a simple exponential distribution. General renewal processes have varying interarrival time distributions, which remove the "complete independence" property of Poisson processes.

Let $F(x)$ denote the cumulative distribution function (cdf) of inter-arrival times in a renewal process which we denote $X \sim F$ i.i.d. By definition, $F(x) = \mathbb{P}\{X \leq x\}$. Then, $1 - F(x) = \mathbb{P}\{X > x\}$, the *survivor function* [17], is interpretable as the probability that a point arrives after an interval longer than t. Finally, we denote the probability density, $p_X(x) = p(x) = dF(x)/dx$. Then,

$$\mathbb{P}\{x < X \leq x + \epsilon | x < X\} = \frac{\mathbb{P}\{x < X \leq x + \epsilon, x < X\}}{\mathbb{P}\{x < X\}} = \frac{\mathbb{P}\{X \leq x + \epsilon\} - \mathbb{P}\{X \leq x\}}{1 - F(x)}.$$

Letting $\epsilon \downarrow 0$,

$$\lim_{\epsilon \downarrow 0} \epsilon^{-1} \mathbb{P}\{x < X \leq x + \epsilon | x < X\} = \frac{dF/dx}{1 - F(x)} = \frac{p(x)}{1 - F(x)}, \tag{2.3}$$

where $p(x)$ is the probability density function corresponding to X. In the context of renewal processes this function $\lambda(x) = p(x)/(1 - F(x))$ is called the *hazard function* [19,20]. Intuitively, $\lambda(x)dx$ is the probability that an occurrence will arrive in $(x, x+dx]$, on condition that it hasn't until time x.

Finally, note that solving the simple separable ordinary differential equation (ODE) (2.3), we have

$$F(x) = 1 - \exp\left(- \int_0^x \lambda(s) ds\right).$$

As $\{x < X \leq x + \epsilon | x < X\} = \{N(x, x + \epsilon] = 1 | x < X\}$, we observe that the clash of notation is not a coincidence, and hazard functions correspond to the intensity function of a Poisson process, introduced above. However, through the general renewal process construction, we lose the defining independence property of Poisson process. That is, the occurrence counts in disjoint intervals are no longer independent. This is easily seen by observing, after an arbitrary point τ, heuristically, the expected point count $\lambda(\tau)d\tau$ explicitly depends on when the last occurrence was.

Instead, renewal processes are defined through a certain type of *conditional* independence structure such that the probability density of the i-th event at t_i is dependent only on the time of the previous occurrence. Concretely, note that the joint density of all occurrence times factorizes as

$$p(\{t_i\}_{i=1}^{N}) = \prod_i p(t_i|t_{i-1}, t_{i-2}, \dots). \tag{2.4}$$

A renewal process admits a conditional independence structure,

$$p(t_i|t_{i-1}, t_{i-2}, \dots) = p(t_i|t_{i-1}) = p(t_i - t_{i-1}).$$

One could extend this construction. What if the time to the next event was conditioned not on the last, but the last two occurrences, $p(t_i|t_{i-1}, t_{i-2}, \dots) = p(t_i|t_{i-1}, t_{i-2})$? This process is known as a Wold process [17], and could assume many parametric forms. For example, one could specify that interarrival times follow an autoregressive process. However, as we shall see, conditional densities of arrival times is not the most convenient construction to work with. Instead, we will rely on another, namely the *conditional* intensity, the first topic of the next section.

Renewal processes will be the focus of Chapter 5, where we will combine them with deep learning to solve a practical problem.

2.2. Self-modulating Temporal Point Processes

2.2.1. Conditional Intensities and Likelihoods

The property that endows Poisson processes with ease of learning and inference is the same property that prohibits any realistic modeling of real-world phenomena: independence. In many discrete-event systems, one is concerned not only with inferring an underlying intensity of individual independent occurrences, but also with modelling how events affect each other.

For example, it is hard to explain earthquake occurrences as a Poisson process. This is since any meaningful models would account for the fact that earthquakes, while occurring stochastically, also excite other earthquakes (*i.e.*, aftershocks). Moreover, it is almost certain that earthquakes do not follow a renewal process. Likewise, stock market orders do not happen independently, but can occur as reactions to other orders. In other words, a fixed intensity does not capture scenarios in which random occurrences of events change the intensity for others. One way to describe such "doubly-stochastic" TPPs,[5] is the *conditional intensity*, which neatly describes the effect of *history* on *future* events.

In the last section, we specified the density of each point's time conditioned on the last occurrence. The probability distribution governing the placement of each consecutive point was equivalently specified by the density p or the hazard function λ. This construction already hinted at a formal and mathematically convenient way to specify more general TPPs. In this section, we will heuristically define the conditional intensity, note that it uniquely determines a TPP, and arrive at a general conclusion about the form of the likelihood. The introduction of this section loosely follows [17, Ch. 7], which the reader may refer to for a thorough treatment.

Let $p(t|\mathcal{H}_t)dt$ denote the probability that there is an occurrence in the small interval $(t, t + dt]$, conditioned on $\mathcal{H}_t$—the history of all occurrences until time t. Formally, in stochastic process theory $\mathcal{H}_t$ is termed the *filtration*, or an increasing set of σ-algebras. In the rest of this work, however, it will suffice to think of $\mathcal{H}_t$ as a (random) point set including all points in $(0, t]$. In line with the notation of [17], we will denote this *conditional* density as $p^*(t_i) = p(t_i|H_{t_i})$.

Then, we can define the conditional intensity function, by analogy to renewal processes.

[5] This general class of TPPs are also referred to as Cox processes.

Definition 2.3. *(Conditional intensity) function is a finite positive function such that*

$$\lambda^*(t) = \lambda(t|\mathcal{H}_t) = \lim_{\epsilon \downarrow 0} \frac{\mathbb{E}\left[N(t-\epsilon, t]|\mathcal{H}_t\right]}{\epsilon}.$$

We also note that the relationship $p(t) = \lambda(t)\exp(-\int_0^t \lambda(s)ds)$ still holds in this case. That is,

$$\lambda^*(t) = \frac{p^*(t)}{1 - F^*(t)} = \frac{d}{dt}\log(1 - F^*(t)).$$

Concretely, the conditional intensity function gives a convenient way to specify how arrival rate of events changes given the history of points. That is, processes described by a conditional intensity function often yield a causality interpretation (in the sense of Granger-causality [21]). It is also convenient since, for reasons analogous to the chain rule of probability, the likelihood function takes a convenient form,

$$\begin{aligned} \ell(\lambda^*) = p(\{t_i\}_{i=1}^N) &= \prod_i p^*(t_i) && (2.5) \\ &= \exp\left(-\int_0^T \lambda^*(s)ds\right)\prod_i \lambda^*(t_i). && (2.6) \end{aligned}$$

Note also that, under mild conditions, a conditional intensity function uniquely determines a TPP (see, *e.g.*, [22]).

We have arrived at a convenient and intuitive way to fully specify a point process on the real line. TPPs defined in terms of their conditional intensities have been referred to as *evolutionary*, or simply as *conditional intensity* processes, among others [17, 20, 23, 24]. In the sequel, we will refer to such models as *self-modulating* TPPs as in [24]. The next section introduces the most typical self-modulating TPP, which will feature frequently in the rest of this thesis.

2.2.2. Hawkes Processes

Hawkes Processes (HP) [25,26], also referred to as *self-exciting processes*, constitute arguably the most popular class of self-modulating TPPs. Introduced in the 1970s to study seismic event data, they have been applied in a wide variety of application domains to date, including neuroscience [9], quantitative finance [27–29], social media analysis [30–32], network analysis [33,34], among others.

Definition 2.4. *An HP is defined as a self-modulating TPP with the conditional intensity*

$$\lambda^*(t) = \lambda^*(t|\mathcal{H}_t) = \mu + \alpha \int dN(t')g(t-t'). \tag{2.7}$$

That is, the conditional intensity of the process depends on the history of events through a linear relationship. Alternatively, put in the more familiar discrete sum notation instead of the stochastic integral notation,

$$\lambda^*(t) = \mu + \alpha \sum_{i|t_i<t} g(t-t_i), \tag{2.8}$$

given data $\{t_i\}_{i=1}^n$. Here, the conditional intensity depends on a positive *baseline intensity* $\mu > 0$ that is often constant across time. It also depends on previous point occurrences $\{t_i|t_i < t\}$, through the *triggering kernel*, $\alpha g(x)$. Often $g(x)$ is a *causal* (*i.e.*, $g(x) = 0, \forall x < 0$) and positive function. It governs temporal relationships among events, *i.e.*, how events excite each other across time. Note that since $g(x) > 0$ by definition, and events only increase the intensity and hence *excite* further events. For mathematical convenience, we take $\int_0^\infty g(x) = 1$ without loss of generality. As such, the function g is called the *delay density.* The factor $\alpha > 0$ governs the degree of self-excitation. Note that, for stationarity, we require that $\alpha < 1$. α can be interpreted as the expected number of events that are “caused” by each event, which is why it is sometimes referred to as the *branching ratio*, or the *infectivity* ratio.

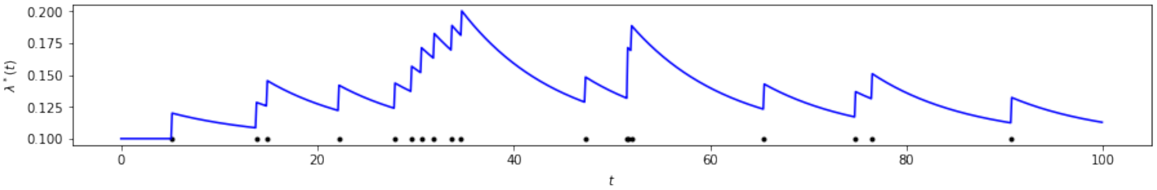


Figure 2.3. A draw from a Hawkes process.

The most common form of the HP is with an exponential delay density, where $g(x) = \beta \exp(-\beta x)$. This choice agrees with our intuition that additive excitation effects decay in time—as the delay density g is monotonically decreasing. It also yields significant computational benefits in computing the likelihood since the process can now be reduced to a Markov process for reasons analogous to memorylessness of exponential densities (see Appendix A). See Figure 2.3 for a random draw from a HP with exponential decay, with the random intensity draw also shown in blue.

Exponential densities do not capture many of the interesting relationships one can model. For example, one may have excitation patterns that are not necessarily decreasing in time. In many application domains, it is known that triggering patterns decay slowly, as in a power-law decay. In seismology, for example, this is known as Omori's law [35]. To tackle learning in more general settings, a variety of techniques for estimating flexible decays have been explored [36, 37]. This function could also be taken as bounded [4, 33], further simplifying learning and inference.

While an HP accounts for the self-exciting behavior of a single type of event, its *multidimensional* extension, the multidimensional Hawkes process (MHP) also treats *mutually exciting* behavior. That is, an MHP affords full flexibility to cover both temporal aspects of excitation as well as cross-mark interactions. Concretely,

Definition 2.5. *An MHP, as a set of counting processes* $[N_1(t), \cdots, N_d(t)]$*, is defined* *in terms of the conditional intensity functions*

$$\lambda_k(t|\mathcal{H}_t) = \mu_k + \sum_l \int dN_l(t')\phi_{kl} g_{kl}(t-t').$$

Here, $g_{lk}(t-t')$ are the set of delay densities that govern the timeline for arrival of *triggered* events. Although we can let this function depend on both l the *source* mark and k the triggered mark, we will often share the triggering kernel across marks. The matrix $[\mathbf{\Phi}]_{kl} = \phi_{kl}$, sometimes referred to as the *infectivity* matrix or *kernel*, describes the self and mutual excitation behavior of the process. For example, $\phi_{kl} = 0.2$ conveniently encodes that, in expectation, each event of type l will result in 0.2 events of type k. We give examples of this behavior in Figure 2.4, where the "mutual-excitation" effect can be observed. The stationarity condition for MHP is given in terms of the *spectral radius* $\rho(\Phi) < 1$.

Another convenient property of MHPs is that the matrix $\mathbf{\Phi}$ is interpretable as the weight matrix of a directed graph of causal influence. See, *e.g.*, [32, 34, 38]. This is why fast and stable recovery of $\mathbf{\Phi}$, and its spectral analysis, are interesting as graph discovery and analysis problems. For example, [4] gives an application of HPs to high-frequency currency trading, where interactions between different major pairs were recovered (Figure 2.5).

Another important property of HP is the Poisson-cluster process interpretation [28, Sec 2.3.7], first explored in [39]. Namely, an HP can be fully characterized in terms of an immigrant-birth process and a set of Poisson processes. Concretely,

Proposition 2.5. *(Poisson-cluster property of HPs) Let* Π_0 *be a homogeneous Poisson TPP given according to the constant baseline intensity* μ*. For each* $t_i^{(0)} \in \Pi_0$*, let* $\Pi_1^{(i)}$ *be a terminating non-homogeneous Poisson process according to the intensity function* $\lambda(t) = \alpha g(t - t_i^{(0)})$*; and* $\Pi_n^{(i)}$ *a set of non-homogeneous Poisson processes according to*

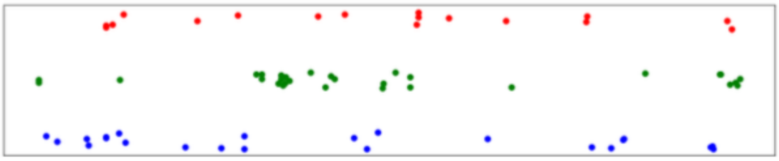
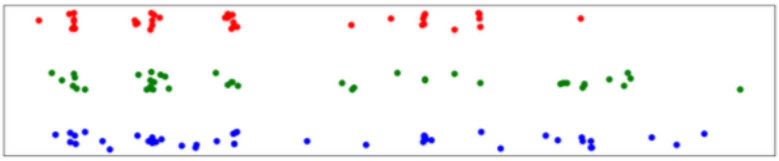
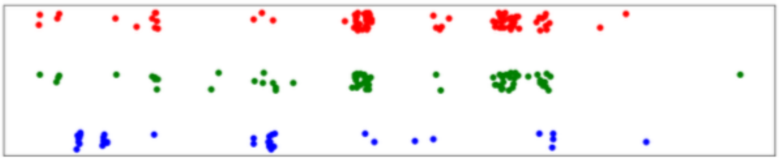
Φ	Sample
$\begin{bmatrix} 0.7 & 0 & 0 \\ 0 & 0.7 & 0 \\ 0 & 0 & 0.7 \end{bmatrix}$	
$\begin{bmatrix} 0.7 & 0.7 & 0.7 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}$	
$\begin{bmatrix} 0 & 0.8 & 0 \\ 0.8 & 0 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$	

Figure 2.4. Draws from MHPs.

$\lambda(t) = \alpha g(t - t_i^{(n-1)})$ in general. Then,

$$\Pi_0 \cup \bigcup_{n=0}^{\infty} \bigcup_{i=1}^{|\Pi_n|} \Pi_{n+1}^{(i)}$$

is a HP as defined in (2.7).

We will not prove this proposition although it should be intuitive through the superposition property of Poisson processes (Proposition 2.2). The Poisson-cluster property leads to an array of useful tools. Moments of the process can be formulated invoking the branching process interpretation [40]. The expectation-maximization algorithm for learning process parameters invokes this property [31,41], as does a specific exact sampling algorithm [42].

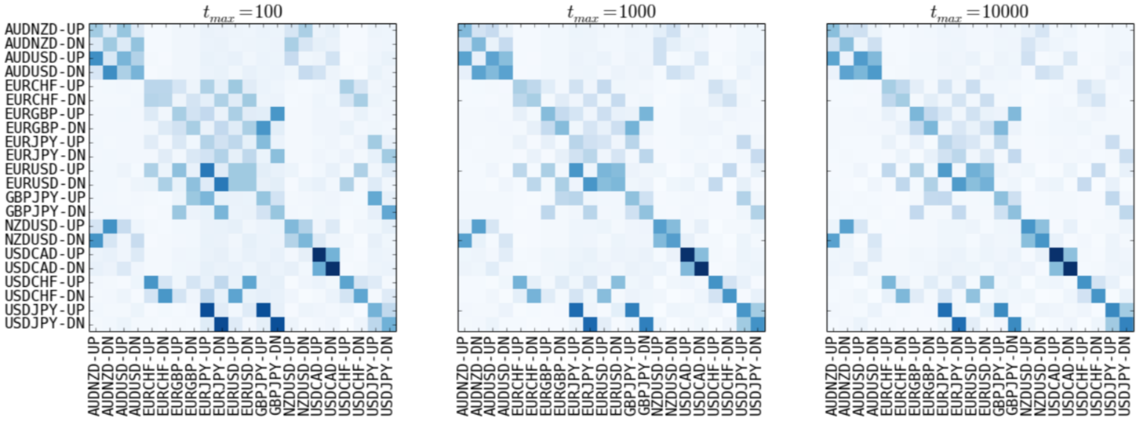


Figure 2.5. Learned MHP infectivity matrices from high-frequency currency data.

2.2.3. Moments of Weakly Stationary Processes

The fundamental difficulty in fitting HPs, or in general any point process defined through a conditional intensity function, is that the computation of the likelihood (or its gradient or expected sufficient statistics), takes time quadratic in the number of events. To see this, note that each computation of $\lambda^*(t_i)$ is already a sum over all $\{t_j\}_{j<i}$, and that the log likelihood computes the intensity for all points. As noted above, this problem can be alleviated via restricted parametric forms of the delay kernel g (*e.g.*, exponential or hyperexponential). Another method in which HP kernels have been estimated is through generalized methods of moments applied, either for parametric or nonparametric kernel forms [36–38, 43, 44].

Yet, the definition of moments in the context of MTPPs can be counter-intuitive. In this section, motivated by the applications above, we give heuristic definitions of first and second moments of a TPP. We will keep MHP as a running example, and favor brevity over a full treatment. For more details, we refer the reader to [17, Ch. 7], or the references in this section.

Formally, moment densities of an MTPP are defined as densities admitted by moment measures [17]. Here, however, we will gloss over this detail and assume such densities are always well-defined.

Definition 2.6. *(First moment density of an MTPP) Given counting processes* $\{N_k(t)\}_{k=1}^d$, *first moments are defined*

$$m_1^{(k)}(t) = \frac{\mathbb{E}\left[dN_k(t)\right]}{dt} = \lim_{\epsilon \downarrow 0} \frac{\mathbb{E}\left[N_k(t+\epsilon) - N_k(t)\right]}{\epsilon}.$$

Unpacking the notation above, we have

$$m_1^{(k)}(t) = \frac{\mathbb{E}\left[dN_k(t)\right]}{dt} = \frac{\mathbb{E}\left[\mathbb{E}\left[dN_k(t)|\mathcal{H}_t\right]\right]}{dt} = \mathbb{E}\left[\lambda_k^*(t)\right],$$

where the first equality is by definition, the second by the “tower property” for stochastic processes and the third by the definition of the intensity function. Intuitively, the first moment of a TPP is the expected intensity at any given point. Let us also define the second (centered) moment density,

Definition 2.7. *(Second centered moment density of an MTPP) Given a set of counting processes* $\{N_k(t)\}_{k=1}^d$, $l \in \{1, \cdots, d\}$ *second centered moments are defined*

$$m_2^{(k,l)}(t) = \frac{\mathbb{E}\left[dN_k(t_k)dN_l(t_l)\right] - \mathbb{E}\left[dN_k(t_k)\right]\mathbb{E}\left[dN_l(t_l)\right]}{dt_k dt_l}.$$

Again, intuitively, the second centered moment is a “covariance” of jump expectations given times t_k, t_l.

We already referred to weakly stationary TPP above. We define such processes (a.k.a. covariance-stationary processes) as those where the first and second moment densities are translation-invariant. Namely, such processes have

$$m_1^{(k)}(t) = \bar{m}_1^{(k)},$$
$$m_2^{(k,l)}(t_k, t_l) = \bar{m}_2^{(k,l)}(\tau),$$

for constant $\bar{m}_1$ and $\bar{m}_2$ that only depends on $\tau = t_k - t_l$. Assuming $\rho(\mathbf{\Phi}) < 1$, we had already noted that this property holds for MHPs.

We heuristically give moment densities for an MHP as,

$$\mathbf{m}_1 = (\mathbf{I} - \mathbf{\Phi})^{-1}\boldsymbol{\mu},$$

where $\boldsymbol{\mu} = [\mu_1, \cdots, \mu_d]$ is a vector of constant background intensities and $\mathbf{m}_1$ corresponding vector of first moment densities.

We can also characterize the relationship between integrated moment matrix $[\mathbf{M}_2]_{kl}dt = \int_0^\infty d\tau m_2^{(k,l)}(t, t+\tau)$ and MHP parameters as

$$\mathbf{M}_2 = (\mathbf{I} - \mathbf{\Phi})^{-1}\mathbf{\Lambda}(\mathbf{I} - \mathbf{\Phi}^T)^{-1}.$$

where we define $\mathbf{\Lambda}$ as the matrix with $\boldsymbol{\mu}$ along its diagonal.

Exact forms of these quantities were given in [40] where the authors rely on Poisson-cluster interpretation of HPs to compute analytical forms for general-order cumulants, and [36] where the authors rely on a Doob-Meyer decomposition argument and work with renewal-type integral equations that result. Recently, Achab *et al.* developed a generalized method of moments routine based on the integrated moments (in their paper, *cumulants*) of MHP [38]. Bacry *et al.* demonstrated that an MHP is fully determined by its first and second moments [45].

2.3. Sampling from a Temporal Point Process

One of the key challenges surrounding TPPs is predictive inference, *i.e.*, characterizing distributions of point occurrences in a future interval given a fitted process. The main difficulty arises from the fact that the expectation is an integral on an unwieldy “point configuration space,” even when the “test function” to be evaluated is

Algorithm 1: Ogata's thinning sampler, for sampling in $(0, T]$

```
for samples do in parallel
    $t \leftarrow 0$
    while $t < T$ do
        $\bar{\lambda} \leftarrow \lambda^*(t + \varepsilon)$
        Draw $z \sim \mathcal{E}(\bar{\lambda})$
        $t \leftarrow t + z$
        Draw $u \sim \mathcal{U}(0, 1)$
        if $u < \lambda^*(t)/\bar{\lambda}$ then
            Add $t$ to sampled points
        end
    end
endfor
```

relatively simple. In this light, prediction tasks in TPPs almost exclusively resort to sampling methods, often also referred to as *simulation* in the context of TPPs.

Sampling from Poisson or renewal processes is straightforward, one only needs to sample a set of interarrival times. Yet, an efficient method for drawing sample point sets from a self-modulating TPP is not obvious. The traditional algorithm for sampling in TPPs is Ogata's method [46]—an adaptation of Lewis' thinning approach [47] to conditional-intensity processes. Pseudo-code for the sampler is given in Algorithm 1, where we let $\mathcal{U}, \mathcal{E}$ denote standard uniform and exponential distributions respectively. An important assumption of the method is that λ^* is non-increasing.

Intuitively, the algorithm proceeds by pointwise rejection sampling. For every drawn point, the next point is proposed from a "faster" Poisson process, and accepted or rejected based on the ratio of the true intensity versus the proposal intensity. For a limited number of cases where the integrated intensity function—the *compensator*—can be analytically inverted, an inversion sampler can replace the rejection sampling substep in the algorithm.

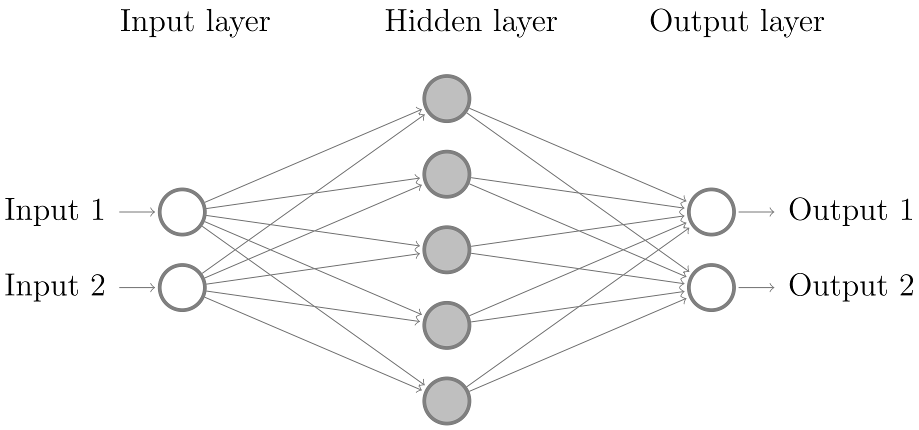


Figure 2.6. A neural network.

When the process is multivariate, the “proposal” draw is made for all d marks, and the first accepted point is taken. This results in $O(nd)$ time complexity in sampling, where n is the number of points drawn.

In Chapter 4, we will explore an alternative sampling approach for TPPs that results in orders of magnitude faster computation via an approximate sampling routine.

2.4. Deep Learning

Neural networks (NN) are function approximators that are composed of smaller computational units, called *nodes* or *neurons*. Neurons, inspired by those in biological neural networks such as the visual cortex [48], are individual simple maps from their inputs to outputs. When functions are composed as layers of such units that each compute on the output of the previous layer, the networks are referred to as feedforward NNs. One such network is illustrated in Figure 2.6. The leftmost layer is the *input* layer that represents the input of the function to be approximated. The five nodes in the middle, called *hidden* units, define a map from the two inputs to an intermediate state. The two nodes on the right map this state to the outputs of the function.

Concretely, let $f : \mathbb{R}^2 \to \mathbb{R}^2$ be the function defined by the NN in Figure 2.6. In one specific instantiation of the feedforward NN, the *multilayer perceptron* (MLP), we choose the action of each hidden unit and output unit to be an affine transformation of its inputs, mapped through a nonlinearity. Letting $\eta(.)$ denote this transformation, we define $\eta(\mathbf{x}|\mathbf{w}, b) = \sigma(\mathbf{w}^T\mathbf{x}+b)$ where $\mathbf{w} \in \mathbb{R}^2, b \in \mathbb{R}$ are parameters that are fitted with data, and $\sigma(x) = 1/(1+e^{-x})$ denotes the logistic function. Then, the NN is defined via the following construction,

$$\begin{aligned} f(\mathbf{x}) &= [\eta(\mathbf{h}|\mathbf{w}_{o_1}, b_{o_1}), \eta(\mathbf{h}|\mathbf{w}_{o_2}, b_{o_2})] \\ [\mathbf{h}]_i &= \eta(\mathbf{x}|\mathbf{w}_{h_i}, b_{h_i}). \end{aligned}$$

Here, the set of all ordered pairs $(\mathbf{w}, b)$ constitute the parameters of the NN often learned via a variant of the *backpropagation* learning algorithm.

Another popular brand of NNs is “recurrent” NNs (RNN) that allow for “self”-connections in the network architecture, retaining some memory of their inputs. Therefore, these networks are popular in a wide array of sequence learning tasks, such as in natural language processing or time-series modeling [49]. One of the most popular RNN architectures is the long short-term memory (LSTM) network [50], which uses a gating mechanism to better represent long term interactions in a sequence.

The most important result concerning MLPs (that often extends to modern NNs) is due to Hornik *et al.* [51], who showed that under the assumption of sufficiently many hidden units, MLPs are universal function approximators. NNs were proposed as early as the 1950s [52]. However, their adoption rose to unprecedented levels with the *deep learning* revolution, centered around the widespread success of *deep* NNs—often used to imply multiple hidden layers in a network architecture—in a wide array of data mining, computer vision, speech recognition and natural language processing tasks, among others [53].

Today, well-implemented easy-to-use software frameworks that feature optimization routines, data processing infrastructure, and widely used network architectures, are generally available [13, 54, 55]. Combined with their consistent performance in setting the state-of-the-art in many artificial intelligence tasks, deep learning approaches and practices feature ubiquitously in machine learning research and applications [56, 57].

2.5. Neural Temporal Point Processes

Self-modulated TPPs offer interpretable and parsimonious statistical models of discrete event occurrences. Statistical properties of popular examples such as HPs have been studied in detail. However, these simple constructions pose challenges in scalable parameter estimation and inference. Moreover, conditional intensity functions are often too limited to capture complex inter-mark and temporal relationship patterns in MTPP modeling.

A recent trend in the machine learning literature aims to tackle these problems by combining TPPs with deep learning. In this section, we survey recent developments in the machine learning literature regarding TPPs, with a special focus on deep TPPs. Our survey will guide our contributions in Chapters 4 and 5.

In their seminal paper, Du *et al.* [58] proposed Recurrent Marked TPP (RMTPP), modeling a multivariate point process via "embedding event history to a vector." The embedding is produced by an RNN, in their experiments an LSTM, which takes the interarrival times and event marks as inputs. Concretely, they take the conditional intensity

$$\lambda^*(t) = \exp(\mathbf{v}^\top \mathbf{h}_i + w(t - t_i) + b), \tag{2.9}$$

where w, b are scalar parameters, $\mathbf{v} \in \mathbb{R}^h$ where h denotes the LSTM output dimensionality. $\mathbf{h}_i$ is the output of the LSTM for point t_i. That is, $h_i = \mathrm{LSTM}(\mathbf{h}_{i-1}, t_i, k_i)$. $i = \sup\{j \in \mathbb{N} : t_j < t\}$.

Furthermore they take,

$$y_{i+1} \sim \mathcal{M}(\bar{\sigma}(\mathbf{V}\mathbf{h}_i + \mathbf{c})),$$

where we let $\bar{\sigma}$ denote the softmax function, $\mathcal{M}$ the categorical distribution, and $\mathbf{V} \in \mathbb{R}^{k \times h}$, $\mathbf{c} \in \mathbb{R}^{k}$ the weight and bias parameters of a dense NN layer that maps LSTM outputs to the categorical likelihood.

This formulation allows bypassing expensive optimization routines in general conditional intensity TPPs while leaving ample capacity for learning complex dependencies across time. This is since the latent embedding $\mathbf{h}_i$ and the point process $N(t)$ jointly constitute a Markov process. Notice how, by design, the intensity function (2.9) does not depend on the history of points, but rather only on the last point in time. This gives an RMTPP the same stochastic structure of a renewal process, although the pdf of the next time point is not explicitly identified.[5] Moreover, the compensator can be computed analytically. The most important consequence of this is that exact likelihood based computation can be carried out in linear time, in contrast to general HPs.

Let us note that this contrasts with how RNNs are generally used in time series modeling. Often, RNNs replace classical time series models that are already based on Markovian conditional independence assumptions, such as AR(p) processes. RNNs serve to express a more complex conditional distribution, in exchange for additional computational cost. In contrast, RMTPP and other examples in this section serve to "approximate" the conditional intensity function which may depend on the entire history. Therefore, they alleviate computational costs in likelihood-based inference, while still providing accurate approximate intensities. This property is also highlighted in the title of the RMTPP paper [58], as the RNN only serves to generate a vector embedding of a past event stream, enough to accurately compute the intensity.

Mei and Eisner extend this formulation in [24]. Written simultaneously with [58], the *neural Hawkes process* generates the intensity function from a *continuous time* LSTM – in which it's not the resulting intensity but the memory cell of the LSTM that decays in time. This model, while perhaps slightly more expressive in that it

[5] Strictly speaking, such a probability density does not exist as the indefinite integral $\int_0^\infty \lambda^*(t)$ converges. That is, there is always a nonzero probability that the next point will not be drawn. Such processes are called *terminating* processes, of which RMTPP is an example.

captures several decaying influences; results in an intractable integral that must be approximated via a Monte Carlo trick. However, the resulting model does not bear any of the defining properties of a HP such as additivity and linearity.

Xiao *et al.* [59] combine effects from a "background" RNN that combines discrete-time RNNs with a separate "event-based" RNN that captures relationships between asynchronous events. In a later version [60], the authors extended this joint model with overlaying "attention," or conditioning the model intensity on several past event-based RNN embeddings in the sequence.[6] Both models yield significant improvements in accuracy over RMTPP.

Jing and Smola's neural survival recommender [61] combines LSTMs with survival analysis, implicitly modeling user behaviour via self-modulating renewal processes. Again, the intensity (in their paper, the *rate*) is given by an LSTM which computes at fine-grained quanta of time. The intensity, in turn, is piecewise constant in these small intervals, making the likelihood tractable. They demonstrate favorable results in time-sensitive recommendation tasks.

Cao *et al.* propose DeepHawkes [62], where they preserve the intensity structure of an HP, and replace the individual components with RNNs. Their model uses user embeddings to capture cross-user interactions, and LSTMs to model the conditional intensity added by individual cascades. They report improved performance in social media prediction tasks. In a similar manner, Trivedi *et al.* [63] rely on *entity embeddings* and an RMTPP type deep TPP to model knowledge graphs evolving in time.

Xiao *et al.*'s Wasserstein-GAN TPP [64] combines optimal transport theory with TPP theory, and proposes generative adversarial networks (GAN) that generate arbitrary TPP draws from a standard Poisson process draw. Although WGANTPP offers an elegant way to sample from arbitrary point processes approximately, it is unclear how both learning and inference tasks scale, as they require an encoder and a decoder

[6] This structure is known as a Wold process in TPP literature.

to run in linear time in generating sequences.

Sharma *et al.* [65] rely on a gamma Markov renewal process (MRP) to model timestamps, and a variational autoencoder-like deep generative model to model complex multivariate marks to model behavioral patterns of zebrafish larvae. Their combined model requires a customized training algorithm, yet yields superior predictive performance compared to other vanilla TPPs and MRPs.

Upadhyay *et al.* [66] explore the problem of deep reinforcement learning under continuous-time feedback and actions, both of which can be cast as marked TPP. Here, the optimal policy is characterized as the optimal intensity for the agent's actions, which is learned via a custom policy gradient method that draws from deep TPP learning.

In their recent paper that gave start to the area of neural differential equations, Chen *et al.* [67] cite asynchronous discrete event data as one of the main potential applications of neural ODEs. This idea has already been explored, in a very recent treatments of neural jump stochastic differential equations (SDE) by Jia and Benson [68], and latent ODE-RNN approach by Rubanova *et al.* [69].

Omi *et al.* [70] rely on a constrained NN architecture to directly approximate the integrated hazard function—equivalently the cdf of interarrival times in a renewal process framework. They demonstrate that monotonicity of this function can be achieved with a network in which the weights are nonnegative. They further note that the intensity itself, required for likelihood computation, comes for free as it is just the gradient of the cdf and is readily computed during backpropagation. Their approach results in dramatic improvements in predictive log-likelihood on real data.

Mei *et al.* [71] explore sequential Monte Carlo methods in the backdrop of TPP, under a bidirectional continuous-time LSTM model in order to impute missing event streams. Their method is unique in focusing on the imputation task, and one of the first experimenting with SMC for TPP along with [72].

A tutorial introduction on TPPs in machine learning can be found in a 2018 tutorial by Gomez-Rodriguez and Valera [73].

3. LOW RANK HAWKES PROCESSES

Among MTPP models explored in the previous chapter, we highlighted two advantages of the MHP—its interpretability and well-explored statistical properties. We also observed that the MHP recovers a graph structure among the marks it models. For example, modelling the timestamps of social media activity of users, one could assume that related users act in response to each other's activity. That is, the rate at which they are active on social media is *excited* by related users. The MHP accurately represents this assumption, and fitting MHP parameters amounts to recovering the underlying social graph.

However, the learning problem is tricky—it requires estimating $O(d^2)$ parameters. When d is large, recovering infectivity parameters is computationally challenging. Moreover, a *curse of dimensionality* question is raised—on whether so many parameters can be fit precisely from limited observations.

One way to mitigate this problem is to assume that the underlying infectivity graph admits a "community" structure. In static analysis of social graphs, this question is often framed as a graph partitioning problem, asking whether nodes of a graph are gathered in partitions with sparse connections between different partitions [74]. Relationships between graph partitioning and low-rank matrix approximation problems are well known [75–77]. In this chapter, we focus on parameter estimation in MHPs whose infectivity matrix admit this community structure, enforced via a rank-constrained parameterization of the process.

Recall the conditional intensity function of an MHP,

$$\lambda_k(t) = \mu_k + \sum_l \int_0^t \phi_{kl} g(t-s) dN(s),$$

and that the matrix $\mathbf{\Phi} \in \mathbb{R}^{d \times d}$ is interpreted as the *infectivity* matrix. That is, each element ϕ_{kl} encodes the magnitude of the (directed) mutual excitation relationship, of events of mark l on those of mark k. $\mathbf{\Phi}$ has been interpreted as the weights in a directed graph where the nodes correspond to the individual marks [10, 33, 34]. Here, we will constrain $\mathbf{\Phi}$ to be a low-rank matrix, and interpret our findings as graph clustering or community detection.

This idea has been explored in the context of MHPs, although it has been treated with different meanings. For example, Zhou *et al.* fit a “low-rank” model by regularizing the learning objective by the nuclear norm [32]. Bacry *et al.* [78] extend estimation procedures for this model. Du *et al.* [79, 80] work with “user-item” pairs—each represented as a mark—and recover a factorized set of parameters for the overall model as is usual in recommendation tasks.

Closest to our work are those by Lemonnier *et al.* [81] and Tran *et al.* [10], both of which directly parameterize the Hawkes model with a factorized, low-rank matrix. Lemonnier *et al.* [81] propose to directly learn a factorized matrix by (i) learning the temporal interaction kernel as a combination of a set of exponential decays and (ii) employing alternating optimization to recover low-rank parameters. They discuss potentially faster algorithms when the matrix is sparse, however they do not address issues around the criticality of the process or nonnegativity of the resulting matrix. They learn $O(dr^2)$ mutual excitation parameters—where r is the “low” rank—and account for the weight between each dimension in the low-rank space explicitly.

Tran *et al.* [10] take a more direct route and work with a factorized form with $O(dr)$ parameters. Notably, they cite a connection to nonnegative matrix factorization (NMF), as we do in the next section, however do not pursue the connection further. Moreover, they give a mean-field variational Bayes algorithm for (partial) Bayesian treatment of the infectivity parameters. Although the authors work with a fixed exponential kernel, their learning algorithm has a reported $O(n^2d)$ complexity.

In the next section, we further explore the structure first introduced in [10]. We make connections to NMF explicit, and pursue them further to recover a low-rank Hawkes kernel via a fast stochastic optimization routine. To arrive at this formulation we rely on the immigrant-birth representation of MHP which leads to the EM algorithm. In Section 3.2, we take a different approach for the same problem. Namely, we demonstrate a path to low-rank MHP kernel approximation via moment-based estimation. We also show links to well-known spectral clustering methods, which yields a direct path to a "spectral clustering of event streams."

Our contributions have explicit connections to well-known low-rank matrix factorization problems and spectral methods. Our proposed algorithms are conceptually simpler, and easy to implement. Most importantly, they belong to an improved time complexity class, yielding substantially faster computational routines.

3.1. Nonnegative Factorization of Hawkes Kernels

In this section, we develop nonnegative low-rank HP (NLRHP). We first give the required background—on the MHP EM algorithm in Section 3.1.1, and NMF in Section 3.1.2. We demonstrate links of our model to NMF in Section 3.1.3, the tools of which we use to develop a generalized EM algorithm. In Section 3.1.4 we describe our optimization algorithm inspired by stochastic gradient methods in NMF.

3.1.1. Expectation-Maximization for Multidimensional Hawkes Processes

As in most models in probabilistic machine learning, gradient-based methods can be used for MHP parameter estimation. However, this approach brings inherent problems, especially since box constraints (nonnegativity conditions) have to be satisfied on all parameters learned. An approach often taken in literature is to invoke the "Poisson-cluster" characterization [39] of MHP in order to write it as a latent variable model (*i.e.*, with data augmentation, see Section 2.2.2), and to perform Expectation-Maximization (EM) [31, 82, 83].

In developing the EM algorithm, one invokes the parenthood structure among events. Concretely, the data is "augmented," with a set of "parenthood" latent variables, identifying the parent process each individual event belongs to. Conditioned on these variables, the "complete data" likelihood decomposes to a set of Poisson process likelihoods.

Let $\llbracket z_i = j \rrbracket$ denote the indicator that the event i is a child of event indexed j. Then the likelihood would be

$$\begin{aligned}\ell(\Theta, Z) = &-\int_0^T \lambda^*(s)ds \\ &+\sum_k \sum_{i\in\mathcal{I}_k} \llbracket z_i = 0 \rrbracket \log \mu_k \\ &+\sum_k \sum_{i\in\mathcal{I}_k} \sum_{j<i} \llbracket z_i = j \rrbracket \log \phi_{k,y_j} g_\beta(t_i - t_j).\end{aligned} \tag{3.1}$$

where we let $z_i = 0$ denote that the event i belongs to the respective background process. The EM algorithm proceeds first by calculating $\mathbb{P}\{z_i = j\} = \mathbb{E}\left[\llbracket z_i = j \rrbracket\right]$, or rather by accumulating these quantities implicitly in *expected sufficient statistics.* Then, in the maximization step one optimizes for $\mu, \mathbf{\Phi}, \beta$ the conditional maximizers of which are easily obtained in closed form.

The main drawback of likelihood based methods is time complexity. It is evident from (3.1) that each computation of the likelihood function, as well as related quantities such as the gradient or the Hessian in gradient ascent or expected sufficient statistics in EM, takes time quadratic in the number of total events. Both approaches benefit from a reduction to linear time for the specific case of an exponential delay density—due to the *memoryless* property of exponentials, detailed in Appendix A.

3.1.2. Nonnegative Matrix Factorization

NMF [84–86] is a low-rank matrix approximation problem that has been studied widely in machine learning. It focuses on the low-rank approximation

$$\mathbf{V} \approx \mathbf{W}^T\mathbf{H},$$

where $\mathbf{V} \in \mathbb{R}^{k\times m}, \mathbf{W} \in \mathbb{R}^{r\times k}, \mathbf{H} \in \mathbb{R}^{r\times m}$ are elementwise nonnegative matrices with $r \ll \min(k, m)$ and $\mathbf{V}$ known. The approximation is in terms of a *divergence*, an often non-symmetric measure of relative distance applied elementwise. That is, cast as an optimization problem, one seeks

$$\min_{\mathbf{W},\mathbf{H}} D(\mathbf{V}||\mathbf{W}^T\mathbf{H}) \text{ s.t. } \mathbf{W}, \mathbf{H} \geq 0.$$

A concrete example for a divergence is the squared Euclidean distance,

$$D_E(\mathbf{V}||\mathbf{W}^T\mathbf{H}) \triangleq \left\|\mathbf{V} - \mathbf{W}^T\mathbf{H}\right\|_F^2 = \sum_{km}(v_{km} - \mathbf{w}_k^T\mathbf{h}_m)^2,$$

or the Information (generalized Kullback-Leibler (KL)) divergence

$$D_{KL}(\mathbf{V}||\mathbf{W}^T\mathbf{H}) \triangleq \sum_{k,m} v_{km} \log \frac{v_{km}}{\mathbf{w}_k^T\mathbf{h}_m} - v_{km} + \mathbf{w}_k^T\mathbf{h}_m.$$

The problem is non-convex in general, and achieved with the original multiplicative updates derived from gradient descent [86], projected gradient [87] methods, or a Bayesian treatment of the model [88].

The applications of matrix factorizations toward recommender systems are well-known, where the objective is to recover a latent space which users (indexed in the above notation with k, say) and items (resp., m) are represented in matrices $\mathbf{W}, \mathbf{H}$. NMF has been linked to graph min-cut problems and spectral clustering [77], and is often interpreted as community or clique discovery in graphs. Moreover, it has been interpreted as a *soft* clustering for general data [89], or tied to probabilistic latent semantic indexing (PLSI) in text analysis [90].

3.1.3. Nonnegative Low Rank Hawkes Process

We propose to learn a factorized, low rank MHP infectivity matrix,

$$\mathbf{\Phi} = \mathbf{W}^T\mathbf{H},$$

where $\mathbf{W}, \mathbf{H} \in \mathbb{R}^{r\times d}$, and $r \ll d$. We further constrain $\mathbf{W}, \mathbf{H} \geq 0$ as above.

We can now formalize a link between low-rank MHP and NMF, and demonstrate how NMF arises naturally. First recall that the EM algorithm (see Murphy [91, Section 11.4]) proceeds by maximizing the expectation of the complete data likelihood (3.1) with respect to the posterior distributions of z_i, also called the *evidence lower bound* or the expected CDLL (ECDLL). We establish a link between NMF and the ECDLL in the following proposition, the main result of this section.

Proposition 3.1. *The evidence lower bound of a low-rank HP determined by the conditional intensity function*

$$\lambda_k^*(t) = \mu_k + \sum_{t_i<t} \mathbf{w}_k^T \mathbf{h}_{y_i} g_\beta(t - t_i), \tag{3.2}$$

is maximized under the optimal NMF of expected sufficient statistics under Information (generalized KL) divergence.

Proof. Replacing $\mathbf{\Phi}$ with $\mathbf{W}^T\mathbf{H}$, we write the terms of the bound function, the expectation of (3.1) under the posterior of $\mathbf{z}$, that depend on $\mathbf{W}, \mathbf{H}$

$$\begin{aligned}\mathcal{B}(\Theta, Z) &= -\sum_k \int_0^T \sum_{t_i<s} \mathbf{w}_k^T \mathbf{h}_{y_i} g_\beta(s-t_i)ds + \sum_k \sum_{i\in\mathcal{I}_k} \sum_{j<i} \mathbb{E}\left[[\![z_i = j]\!]\right] \log \mathbf{w}_k^T \mathbf{h}_{y_j} \\ &= -\sum_{k,l} \mathbf{w}_k^T \mathbf{h}_l \sum_{i\in\mathcal{I}_l} G_\beta(T-t_i) + \sum_{k,l} \log \mathbf{w}_k^T \mathbf{h}_l \sum_{i\in\mathcal{I}_k} \sum_{j<i, j\in\mathcal{I}_l} \mathbb{E}\left[[\![z_i = j]\!]\right].\end{aligned}$$

where G is the cdf corresponding to the delay density g. We also denote by $\mathcal{I}_k = \{i \in \mathbb{N} | y_i = k\}$ the set of all event indices belonging to mark k.

We now introduce matrices $\mathbf{D}, \mathbf{E} \in \mathbb{R}^{d\times d}$, where the diagonal matrix $\mathbf{D} = \text{diag}(\boldsymbol{\delta})$ is defined where $[\boldsymbol{\delta}]_k \triangleq \sum_{i\in\mathcal{I}_k} G_\beta(T-t_i)$, and $e_{kl} \triangleq \sum_{i\in\mathcal{I}_k} \sum_{j<i, j\in\mathcal{I}_l} \mathbb{E}\left[[\![z_i = j]\!]\right]$. The expression reduces to

$$\mathcal{B}(\Theta, Z) = \sum_{k,l} -\delta_l \mathbf{w}_k^T \mathbf{h}_l + e_{kl} \log \mathbf{w}_k^T \mathbf{h}_l = \sum_{k,l} -[\mathbf{W}^T\mathbf{H}\mathbf{D}]_{kl} + [\mathbf{E}]_{kl} \log [\mathbf{W}^T\mathbf{H}]_{kl}.$$

Rescaling the second term we have

$$\begin{aligned}\mathcal{B}(\Theta, Z) &= -\sum_{k,l} [\mathbf{W}^T\mathbf{H}\mathbf{D}]_{kl} + [\mathbf{E}]_{kl} \log [\mathbf{W}^T\mathbf{H}\mathbf{D}]_{kl} + \mathbf{E}_{kl} \log \mathbf{D}_{kl}^{-1} \\ &= -D_{KL}(\mathbf{E} || \mathbf{W}^T\mathbf{H}\mathbf{D}) + \mathbf{E}_{kl} \log \mathbf{D}_{kl}^{-1}.\end{aligned}$$

where the final term does not depend on $\mathbf{W}, \mathbf{H}$. This completes our proof that the bound maximization problem, is equivalent to finding an NMF

$$\mathbf{E} \approx \mathbf{W}^T(\mathbf{H}\mathbf{D})$$

under I-divergence. □

Having made this connection, we can invoke models, algorithms and theoretical arguments made in the context of NMF to parameterize and learn MHP. For example, our choice of nonnegative component matrices $\mathbf{W}, \mathbf{H}$ now has a clear justification, as

NMF is known to produce a more interpretable and parts-based representation [85]. It is also amenable to a "clustering" interpretation, which can be one of the goals of using a low rank MHP model [92]. Using this connection, other NMF parameterizations such as projective NMF [93] or symmetric NMF [94] can be used to approximate the infectivity matrix. Regularizing MHP kernels or imposing further structure can be justified on the same footing as NMF. Well-studied algorithms such as projected gradients [87] or multiplicative updates [86] can be used as a subroutine to MHP EM.

To invoke Proposition 3.1, one could run the MHP-EM algorithm with any NMF algorithm in the maximization step to get the component matrices $\mathbf{W}, \mathbf{H}$ for that iteration. This approach departs from the *plain vanilla* EM in one important aspect. Primarily due to the non-convexity of the NMF problem, the global optimum factorization cannot be attained. Moreover, in general, the optimal solution to NMF is not identified as under any general nonsingular map $\mathbf{C} \in \mathbb{R}^{r \times r}$ one can write $\mathbf{W}^T\mathbf{C}\mathbf{C}^{-1}\mathbf{H}$ to see that $\mathbf{W}^T\mathbf{C}$ and $\mathbf{C}^{-1}\mathbf{H}$ result in the same value of the objective function. However, our proposed approach remains valid as it is an instance of generalized EM.[1] That is, NMF algorithms ensure [86] that the objective decreases monotonically under the respective update rules. This guarantees that the EM objective is nondecreasing, and converges to a stationary point.

We give a sketch of this "batch" generalized-EM approach in Algorithm 2, where m_{ep} denotes the number of epochs which τ indexes. The computations of β, λ^* are conditioned on the parameters at epoch τ. $\mathbf{s}, \boldsymbol{\delta}, \mathbf{E}$ are expected sufficient statistics, as introduced in Proposition 3.1. For brevity, we suppress expectation computations and updates for the parameter β in the presentation of Algorithm 2. Our pseudocode also omits the "recursive" computation of $\exp(-\beta(t_i - t_j))/\lambda^*_{y_i}(t_i)$ decreasing the complexity of the inner for loop to $O(n)$. The derivation of this computation can be found in Appendix A, and an implementation can be found in `hawkeslib` [12].

[1] The term *generalized* EM appeared in the original article by Dempster *et al.* [82], although it is popular to refer to such algorithms as Majorization-Minimization approaches [95] in much of the machine learning literature. Also see Murphy [91, pp. 368–369].

Algorithm 2: Nonnegative Low Rank Hawkes-NMF (NLHRP-NMF)

```
Input: ${t_i, y_i}_{i=1}^n, T, m_{ep}$
for $\tau \in \{1, \cdots, m_{ep}\}$ do
    $e_{kl} \leftarrow 0, \forall k, l \in [d]$
    $s_k \leftarrow 0, \forall k$
    $\delta_k \leftarrow 0, \forall k$
    for $i \in [n]$ do
        $e_{y_i,l} \leftarrow e_{y_i,l} + \sum_{j \in \mathcal{I}_l} \exp(-\beta(t_i - t_j))/\lambda^*_{y_i}(t_i)$
        $s_{y_i} \leftarrow s_{y_i} + \frac{1}{\lambda^*_{y_i}(t_i)}$
        $\delta_{y_i} \leftarrow \delta_{y_i} + G_\beta(T - t_i)$
    end
    $\mu_k^{(\tau)} \leftarrow \frac{\mu_k^{(\tau-1)} \times s_k^\mu}{T}$
    $\mathbf{W}^{(\tau)}, \mathbf{X}^{(\tau)} \leftarrow \texttt{NMF}(\mathbf{E}, \boldsymbol{\delta})$
    $\mathbf{H}^{(\tau)} \leftarrow \mathbf{X}^{(\tau)}\mathbf{D}^{-1}$
end
```

The computation of the inner for-loop in Algorithm 2 requires $O(nd)$ computations, and the `NMF` subroutine, which refers to any NMF algorithm, has a general complexity of $O(d^2r)$. This brings the overall computational complexity to $O(nd + d^2r)$ per epoch. However, this is a relatively inefficient algorithm since the plain vanilla EM algorithm would just have a complexity of $O(nd + d^2)$. Moreover, naive implementations of the algorithm inevitably result in $O(d^2)$ space complexity. We will revisit this issue in the next section.

Before moving on to discuss more viable algorithms for low-rank MHP, let us discuss another theoretical issue that has not been previously treated. Recall that the stationarity condition of the MHP is given via the spectral radius of $\mathbf{\Phi}$, as $\rho(\mathbf{\Phi}) < 1$. Indeed, already having a low-rank $\mathbf{\Phi}$, we can "encourage" this condition in the learning algorithm, and build it into EM via regularization. To see how, recall that

Proposition 3.2. *(Perron's theorem [96, Section 8.2]) For every nonnegative matrix*

$\mathbf{A} \in \mathbb{R}^{d \times d}$ *there exists a nonnegative eigenvalue* $r = \rho(\mathbf{A})$ *s.t.* $r \geq 0$. r *is called the Perron root of the matrix* $\mathbf{A}$.

Proof. Omitted. See [96, Section 8.3]. □

Indeed, it is guaranteed that the spectral radius of $\mathbf{\Phi}$ is a nonnegative number. By definition, it is a member of the spectrum of $\mathbf{\Phi}$, which must have r nonzero elements at most. Moreover, the spectral radius is a global lower-bound to all operator norms. Then,

Lemma 3.1. *For* $\mathbf{A} \in \mathbb{R}^{d \times d}$ *and* $r = \rho(\mathbf{A})$, $r \leq \|\mathbf{A}\|_\infty = \max_k \sum_l a_{kl}$.

In our case, this reduces to $\max_k \sum_l \phi_{kl} = \max_k \sum_r w_{rk} \sum_l h_{rl}$, *i.e.*, both forward and backward computations of the bound can be computed in $O(dr)$ time. Then, one can simply add a regularization term to the learning objective,

$$\min_{\mathbf{W},\mathbf{H}} D_{KL}(\mathbf{E}||\mathbf{W}^T\mathbf{H}\mathbf{D}) + \zeta \max(||\mathbf{H}^T\mathbf{W}||_1, 1),$$

to ensure consistency and stationarity. Similar regularized loss functions have been explored in the NMF literature [92]. The bound in Lemma 3.1 may be too tight. Sharper bounds for the Perron root have been explored, *e.g.*, see [97].

3.1.4. Accelerating Hawkes-NMF Parameter Estimation

The connection drawn between NMF and MHP is not yet very useful, as the naive Algorithm 2 *adds* to the computational requirement for learning an MHP.

In both previous works on low-rank HP, [10,81], temporal components of the delay kernels were fixed, *i.e.*, the *rates* of exponential decays were taken as hyperparameters. Tran *et al.* [10] also pointed out that this could lead to important computational improvements, but did not pursue this avenue further. Under this assumption, it is key to note that the "memory" of the process (see Appendix A),

$$M_k(t) \triangleq \sum_{i \in \mathcal{I}_k | t_i < t} \exp(-\beta(t - t_i)),$$

as well as the sufficient statistic $\boldsymbol{\delta}$ can be precomputed. In other words, only one linear scan through the data is required. This readily suggests that a data augmentation scheme is no longer required. Indeed, under this assumption, a stochastic gradient descent algorithm can be derived, as we show below.

For ease of exposition, we will first denote the precomputed sufficient statistic $m_{ki} \triangleq M_k(t_i)$. Take the full MHP log likelihood, with respect to parameters $\mathbf{W}, \mathbf{H}, \boldsymbol{\mu}$

$$\ell(\mathbf{W}, \mathbf{H}, \boldsymbol{\mu}) = \sum_k \sum_{i \in \mathcal{I}_k} \log \left(\mu_k + \beta \sum_l \mathbf{w}_k^T \mathbf{h}_l m_{li} \right) - \sum_{k,l} \delta_l \sum_r w_{rk} h_{rl} \tag{3.3}$$

Taking derivatives, we have

$$\frac{\partial \ell}{\partial \mu_k} = \sum_{i \in \mathcal{I}_k} \lambda_k^*(t_i)^{-1} \tag{3.4a}$$

$$\frac{\partial \ell}{\partial w_{rk}} = \sum_{i \in \mathcal{I}_k} \frac{\beta \sum_l h_{rl} m_{li}}{\lambda_k^*(t_i)} - \sum_l h_{rl} \delta_l \tag{3.4b}$$

$$\frac{\partial \ell}{\partial h_{rl}} = \sum_k \sum_{i \in \mathcal{I}_k} \frac{\beta w_{rk} m_{li}}{\lambda_k^*(t_i)} - \delta_l \sum_k w_{rk} \tag{3.4c}$$

where

$$\lambda_k^*(t_i) = \mu_k + \beta \sum_r w_{rk} \sum_l m_{li} h_{rl}. \tag{3.5}$$

It is easily verified that "batch" gradient computations (3.4) require $O(ndr)$ time. A batch gradient descent algorithm is now obvious, saving for the outstanding issue of satisfying nonnegativity constraints on $\mathbf{W}, \mathbf{H}$. We will not treat these constraints explicitly here, however. First, as we have already made the connections to NMF we can recognize (3.4b) and (3.4c) as “scaled” KL-NMF parameter updates. Therefore, observations regarding projected gradient [87] and multiplicative updates [86] apply. Moreover, the objective (3.3) is easily implemented in modern deep learning libraries. Therefore, nonnegativity constraints can be satisfied as is now customary in the deep learning literature—via a nonlinear projection—such as the softplus function $\log(1+e^x)$.

Only one sequential scan over the data is required for the computation of $\mathbf{M}$. Individual points $i \in [n]$ (columns of $\mathbf{M}$) can now be parallelized over or implemented using well-tuned vectorized matrix dot product kernels, substantively accelerating the learning algorithm in practice. Another benefit is that the memory data structure $\mathbf{M}$ generalizes conveniently to non-exponential kernels. In this case, one would need to precompute this data structure in $O(n^2)$ time, a price that would be paid only once.

A stochastic gradient descent algorithm easily derives from our observations above. To expose this, we will benefit from a slight rearrangement of notation,

$$\ell(\mathbf{W}, \mathbf{H}, \boldsymbol{\mu}) = \sum_i \left(\log \left(\mu_{y_i} + \beta \sum_l \mathbf{w}_{y_i}^T \mathbf{h}_l m_{li} \right) - \frac{1}{n} \sum_{k,l} \delta_l \sum_r w_{rk} h_{rl} \right). \tag{3.6}$$

From here, we can derive gradients under single event observations,

$$\frac{\partial \ell(\Theta|(t_i, y_i))}{\partial \mu_k} = [\![y_i = k]\!] \lambda^*_{y_i}(t_i)^{-1} \tag{3.7a}$$

$$\frac{\partial \ell(\Theta|(t_i, y_i))}{\partial w_{rk}} = [\![y_i = k]\!] \frac{\beta \sum_l h_{rl} m_{li}}{\lambda^*_{y_i}(t_i)} - \sum_l h_{rl} \delta_l \tag{3.7b}$$

$$\frac{\partial \ell(\Theta|(t_i, y_i))}{\partial h_{rl}} = \frac{\beta w_{r y_i} m_{li}}{\lambda^*_{y_i}(t_i)} - \sum_k w_{rk} \delta_l. \tag{3.7c}$$

We give pseudo-code for our proposed approach in Algorithm 3, where

$$[\nabla \mathbf{w}]_{rk} \triangleq - \frac{\partial \ell(\Theta|(t_i, y_i))}{\partial w_{rk}}$$

Algorithm 3: Nonnegative Low-Rank Hawkes Process-Stochastic Gradient Descent (NLRHP-SGD)

```
Input: $\{t_i, y_i\}_{i=1}^n, T, m_{ep}$
begin
    Initialize $\mathbf{W}, \mathbf{H}, \boldsymbol{\mu}$ randomly
    $\delta_l \leftarrow 0, \forall l$
    $m_{ki} \leftarrow 0, \forall i \in [n], k \in [d]$
    for $i \in [n]$ do
        $m_{ki} \leftarrow \sum_{j \in \mathcal{I}_k} \exp(-\beta(t_i - t_j))$        // Persist the "memory"
        $\delta_{y_i} \leftarrow \delta_{y_i} + G_\beta(T - t_i)$
    end
    for $\tau \in \{1, \cdots, m_{ep}\}$ do
        for sample $j \sim \mathcal{U}([n])$ do
            Compute $\nabla_{\mathbf{W}}, \nabla_{\mathbf{H}}, \nabla_{\boldsymbol{\mu}}$ as per equations (3.7)
            $\mathbf{W} \leftarrow \mathbf{W} - \eta_\tau \nabla_{\mathbf{W}}$
            $\mathbf{H} \leftarrow \mathbf{H} - \eta_\tau \nabla_{\mathbf{H}}$
            $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} - \eta_\tau \nabla_{\boldsymbol{\mu}}$
        end
    end
end
```

is the gradient of the *negative* log likelihood loss, defined *mutatis mutandis* for $\mathbf{H}, \boldsymbol{\mu}$. We also define $\{\eta_\tau\}_{\tau=1}^{m_{ep}}$ as a sequence of step sizes that satisfies the well-known Robbins-Monro conditions [98], such as $\eta_\tau \triangleq c/\tau$ for some positive constant c.

The NLRHP-SGD algorithm has a complexity of $O(dr)$ per data point processed bringing the overall complexity to $O(ndr)$. It compares favorably to previously proposed low-rank MHPs, as we show in Table 3.1. It is easily implemented on modern deep learning libraries [13], and can benefit from their auto-differentiation capabilities as well as off-the-shelf stochastic optimization routines.

Table 3.1. Comparison of Low-rank MHP algorithms.

Model	Time Complexity	Space Complexity	Method
Sparse Low Rank HP [32]	$O(n^2d + d^2)$	$O(d^2)$	ADMM-MM
NetCodec [10]	$O(ndr + d^2r)$	$O(dr + nr)$	Variational Bayes
Scalable LRHP (SLHRP) [81]	$O(ndr^2 + r^4)$	$O(nd)$	MM
NLRHP-NMF (Algorithm 2)	$O(nd + d^2r)$	$O(d^2)$	EM-NMF
NLRHP-SGD (Algorithm 3)	$O(ndr)$	$O(nd)$	SGD

Before concluding this section, we dwell on some implications that emerge from our discussion. In many real-world cases, the infectivity parameter $\mathbf{\Phi}$ is sparse, *i.e.*, the degrees of nodes are small. Ways in which this sparsity can be invoked were considered in [81], and is often discussed in the MHP literature.

Here, again, we can invoke the connection to NMF. First, it is self-evident that our batch NMF approaches could be greatly accelerated, as sparse $\mathbf{\Phi}$ implies sparse memory $\mathbf{M}$. This would result in an improvement in both collecting sufficient statistics in NLRHP-NMF, and even further dramatic speedups in the NMF subroutine. The NLRHP-NMF algorithm would similarly benefit from the sparsity of $\mathbf{M}$, bringing the cost of parameter estimation to virtually linear time in the number of events.

Although we will not develop this idea here, note that our notation also suggests an "online low-rank" learning algorithm as an immediate next step to our approach. This is significant, as in many contexts where the MHP is applied, such as forecasting, finance, and community detection at web scale; learning tasks can be naturally framed naturally as online learning problems.

Finally, ensuring nonnegativity $\mathbf{\Phi} \geq 0$ and stationarity $\rho(\mathbf{\Phi}) < 1$ have been recurring issues in previous works—with no comprehensive treatment. Fortunately, our links to NMF and Perron-Frobenius theory yield simple and practical solutions to both.

3.2. Direct Estimation of Low Rank Symmetric Kernels

In the previous section, we showed that low-rank estimation of Hawkes kernels could be reduced to "just" an NMF problem, and proposed learning algorithms in $O(nd + d^2r)$ or $O(ndr)$ times. In this section, we will reduce the problem of low-rank parameter estimation to just a factorization problem, using moment-matching type estimators. A subset of the approach outlined in this section was published in [99].

Moment-based nonparametric estimation of HP has been explored in depth in a series of works by Bacry *et al.* [36, 45, 100]. These works focus on the stationary mean and covariance densities of the process, which they demonstrate fully determine an MHP. The recent work by Achab *et al.* [38] develops this approach further by fully abstracting away the delay density estimation, and focusing on the infectivity matrix $\mathbf{\Phi}$. The key insight is to work directly on the *integrated moment density* [40] (*i.e.*, moment measure or in their work, "integrated cumulants") of the process. Yet, as we presented in Section 2.2.2, such methods require the computation of the "kernel inversion," which takes $O(d^3)$ time in general.

Here, we explore extensions under two added constraints: we will assume that the infectivity structure is symmetric ($\mathbf{\Phi} = \mathbf{\Phi}^T$) as in [36], and low-rank as in the previous section. We show that these two assumptions result in a set of substantial computational simplifications, while paving the way to other interesting connections.

Recovering the notation of Section 2.2.2, we had defined the first and second moment densities of an MTPP as

$$\begin{aligned} m_1^{(k)}(t) = \bar{m}_1^{(k)} &= \frac{\mathbb{E}\left[dN_k(t)\right]}{dt} = \mathbb{E}\left[\lambda^*(t)\right] \\ m_2^{(k,l)}(t_k, t_l) = \bar{m}_2^{(k,l)}(\tau) &= \frac{\mathbb{E}\left[dN_k(t)dN_l(t+\tau)\right] - \mathbb{E}\left[dN_k(t)\right]\mathbb{E}\left[dN_l(t+\tau)\right]}{dtd\tau}, \end{aligned}$$

where the first equalities follow from the *stationarity* of MHP. In other words, we now *assume* that $\rho(\mathbf{\Phi}) < 1$ holds. We will denote,

$$\begin{aligned}
\mathbf{m}_1 &= [\bar{m}_1^{(1)}, \bar{m}_1^{(2)}, \cdots, \bar{m}_1^{(d)}] \\
\mathbf{M}_1 &= \mathrm{diag}(\mathbf{m}_1) \\
[\mathbf{M}_2]_{kl} &= \int_0^\infty \bar{m}_2^{(k,l)}(\tau) d\tau,
\end{aligned}$$

where $[\mathbf{M}_2]_{kl}$ denotes the *integrated* second cumulants. Their relationships to MHP parameters are well-known and given in the references above as,

$$\mathbf{m}_1 = (\mathbf{I} - \mathbf{\Phi})^{-1}\boldsymbol{\mu}, \tag{3.8a}$$

$$\mathbf{M}_2 = (\mathbf{I} - \mathbf{\Phi})^{-1}\mathbf{M}_1(\mathbf{I} - \mathbf{\Phi}^T)^{-1}. \tag{3.8b}$$

Note that, by our assumption of symmetry, $(\mathbf{I} - \mathbf{\Phi}^T)^{-1} = (\mathbf{I} - \mathbf{\Phi})^{-1}$.

Let us first highlight a set of simple properties of the quantities under consideration. We will often encounter $\mathbf{\Psi} \in \mathbb{R}^{d\times d}$ in this section, defined $\mathbf{\Psi} \triangleq (\mathbf{I} - \mathbf{\Phi})^{-1}$, which we refer to as the *inverse kernel*. Its main interpretation will be apparent in the following simple lemma. Note that,

Lemma 3.2. $\mathbf{\Psi} = \mathbf{\Psi}^T$ *and* $\mathbf{\Psi} \geq 0$.

Proof. The first property should be clear as $(\mathbf{A}^{-1})^T = (\mathbf{A}^T)^{-1}$. The second property follows from the stationarity assumption $\rho(\mathbf{\Phi}) < 1$, under which the Neumann series $(\mathbf{I} - \mathbf{\Phi})^{-1} = \sum_{k=0}^\infty \mathbf{\Phi}^k$ converges. Clearly, since $\mathbf{\Phi} \geq 0, \mathbf{\Phi}^k \geq 0$ for all k. □

Corollary 3.1. $\mathbf{M}_2 \geq 0$.

Also note that $\mathbf{M}_2$ is also positive semi-definite (psd), with positive definiteness ensured by full-rank $\mathbf{\Psi}$ and $\mathbf{m}_1 > 0$. Our approach relies on estimating low-rank $\mathbf{\Phi}$ based on moment conditions (3.8). To that end, we will estimate quantities $\mathbf{m}_1, \mathbf{M}_2$ from data, and work towards $\mathbf{\Phi}$.

We first adopt estimates of the quantities $\mathbf{m}_1, \mathbf{M}_2$ as in [38, 40],

$$[\mathbf{D}]_{kk} \triangleq [\widehat{\mathbf{m}}_1]_k = \frac{N_k(T)}{T}, \tag{3.9a}$$

$$[\mathbf{C}]_{kl} \triangleq [\widehat{\mathbf{M}_2}]_{k,l} = \frac{\sum_{i \in \mathcal{I}_l} N_k(t_i + \tau) - N_k(t_i - \tau) - 2\tau \hat{m}_1^k}{T}, \tag{3.9b}$$

where $\tau > 0$ is a long enough stretch of time for which it is assumed $\int_\tau^\infty \bar{m}_2^{(k,l)}(s)ds \approx 0$. Note that both estimates can be computed in a single pass over the data, with only $O(nd)$ operations.[6] What remains is to show that we can get to a low-rank $\mathbf{\Phi}$ from these estimates. Concretely, we will assume that $\mathbf{\Phi}$ admits the factorization

$$\hat{\mathbf{\Phi}} = \mathbf{W}^T \mathbf{W}$$

where $\mathbf{W} \in \mathbb{R}^{r \times d}$ as in the previous section. In this section, we will not assume that $\mathbf{W}$ is nonnegative.

Let us highlight one important consequence of our assumption. Beyond imposing symmetricity on $\mathbf{\Phi}$, this factorization implies that $\mathbf{\Phi} \succeq 0$, which does not need to hold true for general infectivity matrices. Yet, this assumption will be pivotal in our construction below. We will also assume that the properties given above for theoretical moments $\mathbf{M}_2, \mathbf{m}_1$ hold true for their empirical counterparts to avoid any trivialities. We will use estimates and theoretical quantities related to $\mathbf{\Phi}$ and $\mathbf{\Psi}$ interchangeably, as the intention should be clear from the context.

Letting $\mathbf{A}^{\frac{1}{2}}$ denote the matrix square root, well-defined for psd and diagonal matrices, we first show that,

Proposition 3.3. *Let $\mathbf{C} = \mathbf{\Psi} \mathbf{D} \mathbf{\Psi}$, where $\mathbf{D}$ is diagonal and positive and $\mathbf{C}$ is psd and nonnegative. There exists a unique psd solution given*

$$\mathbf{\Psi} = \mathbf{D}^{-\frac{1}{2}} \left(\mathbf{D}^{\frac{1}{2}} \mathbf{C} \mathbf{D}^{\frac{1}{2}} \right)^{\frac{1}{2}} \mathbf{D}^{-\frac{1}{2}}. \tag{3.10}$$

[6] Although this quantity was given as $O(nd^2)$ in [38], a dynamic programming approach is possible.

Proof. Correctness of the above statement is trivial. For uniqueness, note that the system of equations (3.8) result in a continuous-time algebraic Riccati equation (CARE),

$$\mathbf{C} - \mathbf{\Psi}\mathbf{D}\mathbf{\Psi} = 0.$$

as psd-ness of $\mathbf{C}, \mathbf{D}$ are assumed. The pair $[0, \mathbf{C}]$, in turn, is *detectable* since $\mathbf{C}$ is assumed full-rank. Uniqueness of a psd solution follows. See, *e.g.*, [101]. □

We find that we can fix a unique $\mathbf{\Psi}$ under a given $\mathbf{C}, \mathbf{D}$. Note that this also fixes a unique $\mathbf{\Phi}$. However, the representation (3.10) is not very useful, since a full matrix square rooting is required (in $O(d^3)$ time), which defeats the purpose of a low rank approximation. Instead, we can directly consider the loss

$$\mathcal{L}(\mathbf{W}) = ||\mathbf{C} - \hat{\mathbf{\Psi}}\mathbf{D}\hat{\mathbf{\Psi}}||_F^2, \tag{3.11}$$

where $\hat{\mathbf{\Psi}}$ denotes the approximation to $\mathbf{\Psi}$ when $\hat{\mathbf{\Phi}} = \mathbf{W}^T\mathbf{W}$. The key trick we use is through the Woodbury identity, by which we can write

$$\hat{\mathbf{\Psi}} = \mathbf{I} + \mathbf{W}^T(\mathbf{I} - \mathbf{W}\mathbf{W}^T)^{-1}\mathbf{W}.$$

Note that this computation requires $O(d^2r)$ time. This suggests that the loss (3.11) can be directly backpropagated through $\hat{\mathbf{\Psi}}$, laying the foundation for our algorithm. The gradient of the loss can be computed easily on well-implemented frameworks such as Apache MXNet. We give an illustration of the algorithm in Algorithm 4, where `TRUNC-SYM-EVD` denotes the truncated symmetric eigendecomposition.

Before moving on, let us discuss two points that arise as a consequence of our analysis. First, apart from the wide availability of computational tools for general differentiable computing, the complexity of the algorithm decreases to $O(nd + d^2r)$. The $O(nd)$ scan of the data has to be performed only *once*, in stark contrast to the previously proposed low-rank methods and the previous section. We give a comparison

Algorithm 4: Low-Rank Hawkes Process–Stochastic Gradient Descent (LRHP-SGD)

```
Input: $\{t_i, y_i\}_{i=1}^{n}, T, r$
begin
    Estimate $\mathbf{C}, \mathbf{D}$ as per (3.9)
    $\mathbf{\Sigma}, \mathbf{V} \leftarrow$ TRUNC-SYM-EVD$(\mathbf{D}^{\frac{1}{2}}\mathbf{C}\mathbf{D}^{\frac{1}{2}}, r)$
    $\mathbf{\Lambda}, \mathbf{U} \leftarrow$ TRUNC-SYM-EVD$(\mathbf{D}^{\frac{1}{2}}\mathbf{V}\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{V}^T\mathbf{D}^{\frac{1}{2}}, r)$
    $\mathbf{W}^{(0)} \leftarrow \mathbf{\Lambda}^{1/2}\mathbf{U}^T$
    for $\tau \in \{1, \cdots, m_{ep}\}$ do
        $\mathbf{W}^{(\tau)} \leftarrow \mathbf{W}^{(\tau-1)} - \eta_\tau \nabla_{\mathbf{W}}\mathcal{L}$
    end
end
```

Table 3.2. Comparison of LRHP-SGD to other methods.

Model	Time Complexity	Space Complexity	Method
NPHC (Full rank) [38]	$O(nd^2 + md^3)$	$O(d^2)$	AdaGrad [102]
NLRHP-NMF (Algorithm 2)	$O(mnd + md^2r)$	$O(d^2)$	EM-NMF
NLRHP-SGD (Algorithm 3)	$O(mndr)$	$O(nd)$	SGD
LRHP-SGD	$O(nd + md^2r)$	$O(d^2r)$	SGD

in Table 3.2, where m denotes the "number of iterations."

Another key constraint that can easily be enforced is $\mathbf{\Phi} \geq 0$, through soft mappings to the positive hyperquadrant widely available on auto-differentiation libraries. Enforcing the stationarity conditions $\rho(\mathbf{\Phi}) < 1$ also takes a trivial form. As we have access to the spectrum of $\mathbf{\Phi}$, one can easily re-normalize the spectrum to enforce that the obtained matrix leads to a stationary MHP. Let us also emphasize that LRHP-SGD is independent of the choice of the delay density—as it completely abstracts this choice away.

Finally, let us note that the full inversion of $\mathbf{\Psi}$ through the Woodbury identity may prove unstable. In this case, one can use the truncated Neumann series approximation to the inversion, defining the approximate inverse

$$\tilde{\mathbf{\Psi}}^{(p)} = \mathbf{I} + \mathbf{\Phi} + \mathbf{\Phi}^2 + \cdots + \mathbf{\Phi}^p \tag{3.12a}$$

$$= \mathbf{I} + \mathbf{W}^T\mathbf{W} + (\mathbf{W}^T\mathbf{W})^2 + \cdots + (\mathbf{W}^T\mathbf{W})^p. \tag{3.12b}$$

Indeed, we take this approach in approximately inverting the kernel in [99].

Note that this often yields a practical approximation for $\mathbf{\Phi}$, denoted $\tilde{\mathbf{\Phi}}$, with a small enough spectral radius as,

Proposition 3.4. $\left\|\mathbf{\Phi} - \tilde{\mathbf{\Phi}}^{(p)}\right\|_2 \leq \|\mathbf{\Phi}\|_2^{p+1} = \rho(\mathbf{\Phi})^{p+1}$.

Proof.

$$\begin{aligned}\mathbf{\Phi} - \tilde{\mathbf{\Phi}}^{(p)} &= \tilde{\mathbf{\Psi}}^{(p)} - \mathbf{\Psi} = \mathbf{\Phi}^{p+1}(\mathbf{I} - \mathbf{\Phi}^{p+1})^{-1}(\mathbf{I} - \mathbf{\Phi}) \\ &= \mathbf{\Phi}^{p+1}(\mathbf{I} + \mathbf{\Phi} + \mathbf{\Phi}^2 + \cdots + \mathbf{\Phi}^p)^{-1} = \mathbf{\Phi}^{p+1}\tilde{\mathbf{\Psi}}^{-1}.\end{aligned}$$

Therefore,

$$\begin{aligned}\left\|\mathbf{\Phi} - \tilde{\mathbf{\Phi}}^{(p)}\right\|_2 &= \left\|\mathbf{\Phi}^{p+1}\tilde{\mathbf{\Psi}}^{-1}\right\|_2 \leq \left\|\mathbf{\Phi}^{p+1}\right\|_2 \left\|\tilde{\mathbf{\Psi}}^{-1}\right\|_2 \\ &= \left\|\mathbf{\Phi}^{p+1}\right\|_2 \leq \|\mathbf{\Phi}\|_2^{p+1} = \rho(\mathbf{\Phi})^{p+1},\end{aligned}$$

where the the second equality is due to $\left\|\tilde{\mathbf{\Psi}}^{-1}\right\|_2 = \left\|\mathbf{I} - \tilde{\mathbf{\Phi}}\right\|_2 = 1$, and $\mathbf{\Phi}$ is low-rank (singular) by assumption. □

3.2.1. Spectral Clustering of Event Streams

Spectral clustering is a clustering method that performs dimensionality reduction based on the spectrum of the symmetric *similarity matrix* of a set of data points before clustering with well-known algorithms such as k-means [76, 103, 104]. In short, it proceeds by performing an eigendecomposition of the *graph Laplacian* matrix, defined below, and performing k-means clustering on the “most significant” eigenvectors. Its connections to graph normalized-cut problems are well-known.

In the previous section, we obtained a low-rank representation of the infectivity matrix. We also observed that this matrix was often interpreted as a graph similarity matrix. Our construction already hints toward a spectral clustering algorithm. Indeed, in this section, we show that a simple algorithm for performing spectral clustering on the networks of event types are possible.

Definition 3.1. *Let* $\mathbf{A} \in \mathbb{R}^{d\times d}$ *be a symmetric similarity matrix. The (unnormalized) graph Laplacian of* $\mathbf{A}$ *is* $\mathbf{L}_{\mathbf{A}} \triangleq \mathbf{\Delta}_{\mathbf{A}} - \mathbf{A}$ *where* $\mathbf{\Delta}_{\mathbf{A}}$ *is the diagonal degree matrix* $[\mathbf{\Delta}_{\mathbf{A}}]_{kk} \triangleq \sum_l [\mathbf{A}]_{kl}$.

Lemma 3.3. *Let* $\mathbf{A}, \mathbf{D} \in \mathbb{R}^{d\times d}$, $\mathbf{A}$ *symmetric and* $\mathbf{D}$ *diagonal.* $\mathbf{L}_{\mathbf{I}-\mathbf{A}} = -\mathbf{L}_{\mathbf{A}} = \mathbf{L}_{-\mathbf{A}}$.

Proof. $\mathbf{I} - \mathbf{\Delta}_{\mathbf{A}} = \mathbf{\Delta}_{\mathbf{B}}$, and $\mathbf{L}_{\mathbf{A}} = \mathbf{\Delta}_{\mathbf{A}} - \mathbf{A} = (\mathbf{I} - \mathbf{\Delta}_{\mathbf{B}}) - (\mathbf{I} - \mathbf{B}) = -\mathbf{L}_{\mathbf{B}}$. □

This lemma allows us to compute a spectral clustering of event streams directly by using a Laplacian-like computation on Ψ^{-1}. We give a description of this method in Algorithm 5. Here, it is worth noting that the second invocation of `TRUNC-SYM-EVD` is not a full eigendecomposition, and has only $O(dr^2)$ complexity in general, as does the computation of the Laplacian. Hence, our algorithm retains its $O(d^2r+nd)$ complexity class, and the additional computations hardly result in any further computation time in practice.

Our algorithm provides both spectral embeddings and a clustering of the underlying graph structure with no explicit representation of the graph. Moreover, it does so in a single scan of the data set. Significantly, to our knowledge, it is the first algorithm that proposes a principled way of clustering high-dimensional asynchronous event streams.

```
Algorithm 5: Low-Rank Hawkes Process–Spectral Clustering (LRHP-SC)
    Input: $\{t_i, y_i\}_{i=1}^{n}, T, r$
    Estimate $\mathbf{C}, \mathbf{D}$ as per (3.9)
    $\mathbf{\Sigma}, \mathbf{V} \leftarrow \texttt{TRUNC-SYM-EVD}(\mathbf{D}^{\frac{1}{2}}\mathbf{C}\mathbf{D}^{\frac{1}{2}}, r)$
    $\mathbf{\Psi}^{-1} \leftarrow \mathbf{D}^{\frac{1}{2}}\mathbf{V}\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{V}^T\mathbf{D}^{\frac{1}{2}}$
    Compute $\mathbf{L}_{\mathbf{\Psi}^{-1}}$ according to Definition 3.1
    Compute $\mathbf{\Lambda}, \mathbf{U} \leftarrow \texttt{TRUNC-SYM-EVD}(\mathbf{L}_{\mathbf{\Psi}^{-1}}, r)$
    Compute a $k$-means clustering of the rows of $\mathbf{U} \in \mathbb{R}^{d \times r}$
```

3.2.2. Numerical Experiments

We validate LRHP algorithms on a set of toy experiments.

In Figure 3.1, we show low rank approximations to the infectivity matrix recovered from tick-level price movements on an interbank currency exchange (cf. Figure 2.5) [4]. An accurate representation of the infectivity matrix is recovered with as little as 8 components. Remarkably, such a representation can be recovered from a data set of 36million individual events with 22 marks, in an average 3.2 seconds on a commodity laptop computer—several orders of magnitude faster than a likelihood-based learning algorithm.

We demonstrate spectral embeddings **U** obtained with the LRHP-SC algorithm in Figure 3.2. Here, the pairs followed by `_D` (resp., `_U`) denote price jump down (resp., up) events. Marks are naturally clustered by shared or “close” currency pairs with as little as two eigenvectors.

Finally, Table 3.3 gives the results of the LRHP-SC algorithm on the data. Currency pairs are naturally grouped into trades that can be interpreted as those often bundled together. For example, cluster 5 identifies an “arbitrage,” on AUD-NZD-USD; while cluster 6 identifies the opposite end of this trade.

Table 3.3. Clusters of currency exchange pairs obtained with LRHP-SC.

Cluster 1	EURCHF_D, EURUSD_U, USDCHF_U
Cluster 2	EURGBP_D, EURJPY_D, EURUSD_D, GBPJPY_D, USDJPY_U
Cluster 3	USDCHF_D
Cluster 4	EURCHF_U
Cluster 5	AUDNZD_D, AUDUSD_U, NZDUSD_D
Cluster 6	AUDNZD_U, AUDUSD_D, NZDUSD_U
Cluster 7	EURGBP_U, EURJPY_U, GBPJPY_U, USDCAD_U, USDCAD_D, USDJPY_D

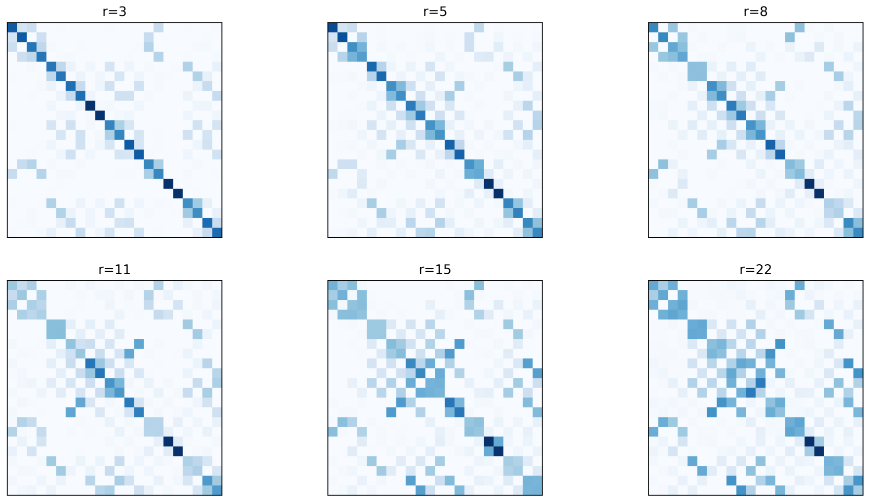


Figure 3.1. Low rank approximations to the kernel of interactions among currency pairs.

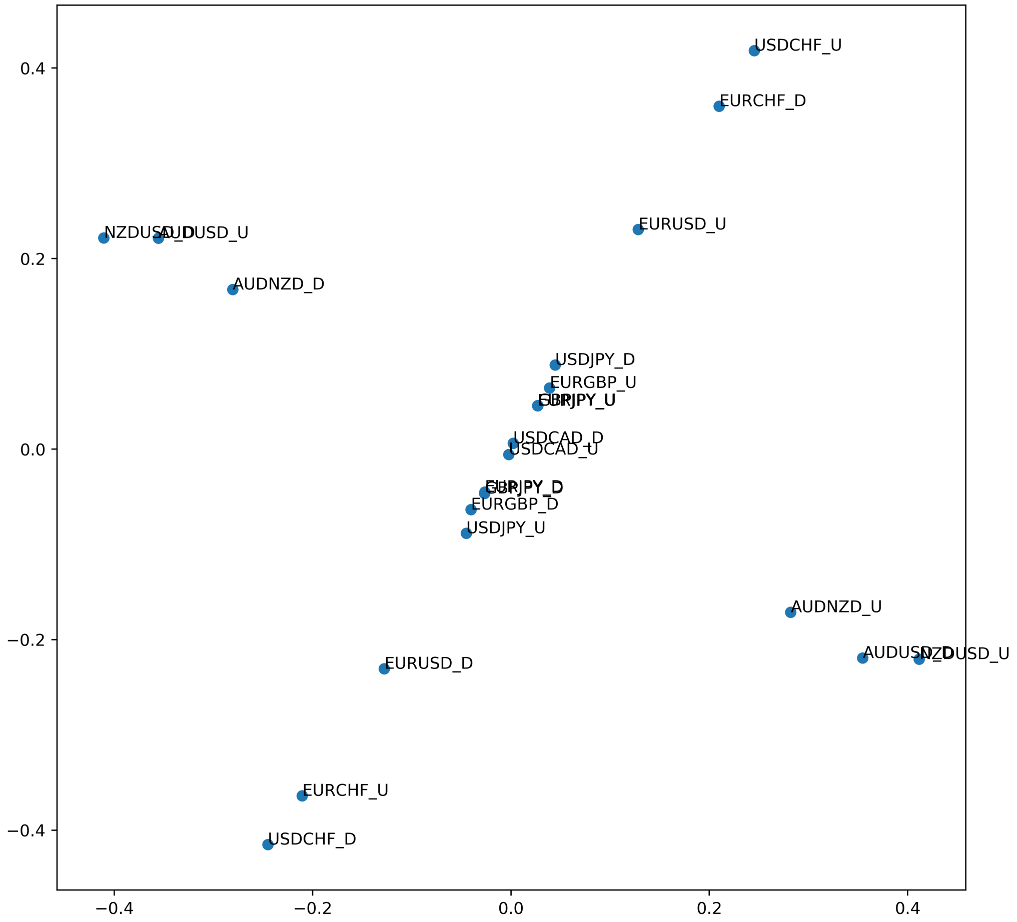


Figure 3.2. Spectral embeddings of currency pairs.

4. GLOBAL-LOCAL TEMPORAL POINT PROCESSES

In the previous section, we looked at efficient ways of working with high-dimensional asynchronous event data. Our analysis was based on the MHP, where we sought to exploit the well-studied statistical properties of the process to learn simple models in a computationally efficient manner. In this section, we change tack, and approach the same issue from a different angle. That is, we rely on the recent approaches of deep TPPs to study high-dimensionality. This chapter jointly addresses efficient learning (estimation) and sampling (simulation), in the context of a novel set of models: global-local MTPPs.

The contributions in this chapter are summarized as follows

- We explore the first "global-local" MTPP (GLMTPP), where we rely on RNNs to model global (cross-mark) excitation patterns while local (intra-mark) excitation is captured by a classical TPP.
- We introduce FastPoint, a specific instantiation of GLMTPPs that can scale to millions of correlated marks in estimation and prediction tasks. The model enables fast maximum likelihood learning two orders of magnitude faster than previous models, while simultaneously resulting in better predictive accuracy.
- We introduce a general sequential Monte Carlo routine for GLMTPP models that exploits the factorization structure as well as a Poisson process proposal to massively parallelize predictive inference in MTPP. This results in dramatic improvements in prediction time while keeping the sample variance comparable.

4.1. Global-Local Multidimensional Temporal Point Processes

In Section 2.5, we introduced neural MTPP models highlighting their main benefit in simplifying parameter estimation. These models encode sequential dependencies through an RNN, and perform well when the number of marks are reasonable.

In Chapter 3, we discussed how an increasing number of marks resulted in time and space complexity that scaled quadratically in the number of marks. In the case of MHP, the main difficulty was the *curse of dimensionality.* In Neural MTPP, the problem is slightly more elusive. Take the RMTPP [58], where each mark is governed by a categorical distribution conditioned on the same LSTM output as the next arrival time distribution. This construction results in several key drawbacks.

- During learning, specifically the computation of the log likelihood, the probability mass of *every* mark has to be accounted for, for *every* point—as the partition function of the categorical mark distribution is always needed. This results in time complexity of $O(nd)$ during NN training, which is achieved with pass-inefficient algorithms such as gradient descent. Seeing as the memory requirement also scales as $O(nd)$, the batch size is also limited during learning, further slowing computation.
- In Chapter 2, we covered sampling as the only viable predictive inference mechanism in TPPs for non-trivial probability queries. However, MTPP sampling algorithms also require $O(nd)$ computation, for reasons analogous to those outlined above—a mark has to be drawn for every point, and every probability mass has to be evaluated. Besides the computational complexity, simulation is also hindered by the fact that points must be drawn in sequence, and the algorithm only parallelizes over samples.
- Finally, note that in RMTPP-like models the entire sequence of events is fed through a single NN. That is, regardless of marks, each event is processed in sequence. When the number of marks d is large, however, the number of possibly unrelated events that arrive in a sequence before a relevant mark is observed easily grows beyond the number easily handled by RNNs—as these models are well-known to fail in representing long-term relationships.

  For example, take a financial market where individual orders of a thousand assets are modeled with a neural MTPP. Furthermore, assume that the background order arrival rate of these assets are similar. Then, in expectation, the number of orders in an arrival sequence between two orders belonging to the same asset

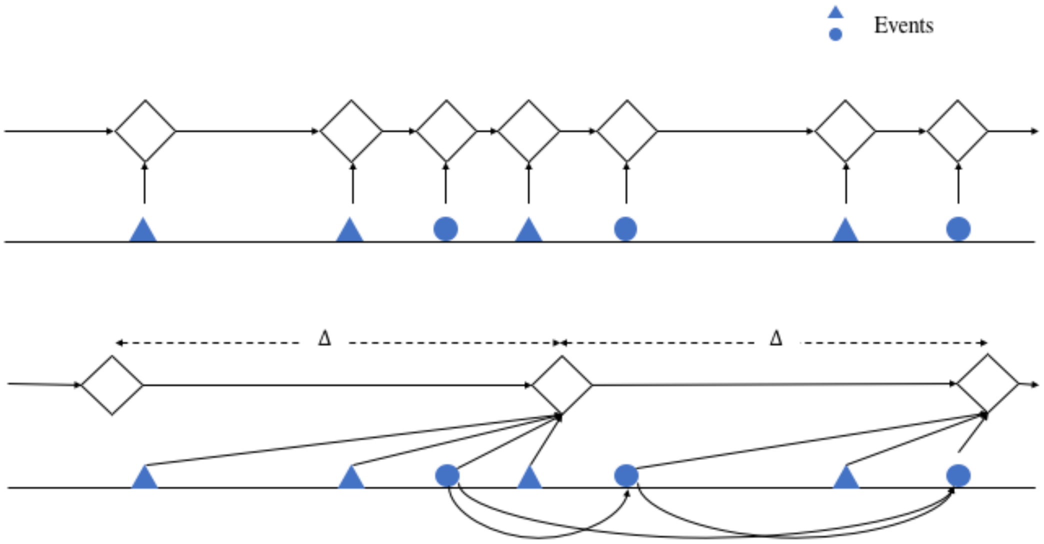


Figure 4.1. Model structures of RMTPP vs GLMTPP.

will be on the same scale as the number of assets. Therefore, it is highly unlikely that an RNN-based model will be able to capture the intra-mark (or, intra-asset, which we refer to as "local" in this chapter) behavior of the underlying process. A similar argument follows for the individual cross-mark effects. This runs contrary to our intuition that local effects probably explain much of the orderly behavior in financial markets.

To that end, we introduce GLMTPP, models that aim to address the shortcomings mentioned above while simultaneously offering a parsimonious representation of asynchronous events. GLMTPPs build on the recent success of global-local time-series models [105, 106].

Concretely, we define a GLMTPP as a MTPP where each conditional intensity function is defined

$$\lambda_k^*(t) = \lambda_k^{(g)}(t|\mathcal{H}_\tau) + \lambda_k^{(l)}(t|\mathcal{H}_t^{(k)}), \tag{4.1}$$

where $\lambda_k^{(g)}, \lambda_k^{(l)}$ refer to global and local intensity processes respectively, and $\mathcal{H}_t^{(k)}$ denotes the "history" of events of type k. Most importantly, the time $\tau = \sup\{s < t | s \in$

$\mathcal{G}\}$ is the last point before time t along $\mathcal{G} = \{0, \Delta, 2\Delta, \cdots\}$—a "fixed grid" in time. Intuitively, the "global" model is a function that "synchronizes" different processes at fixed coarse-grained points in time, and computes an additive intensity component for each individual process. Apart from this global effect, however, each process is itself a well-defined TPP governed by the local conditional intensity that depends only on points of the same mark.

A stylized depiction of FastPoint's model structure is given in Figure 4.1, where events are shown as blue shapes, the synchronization points (intensity computation) of the "global" model as diamonds, and explicit dependencies that are modeled as arrows. Above, a neural MTPP computes the intensity for every mark at each event. Below, a GLMTPP carries out this computation at only given fixed points.

The most important consequence of our model is that likelihood computation decreases from $O(nd)$ in the case of neural MTPPs or $O(n^2 + nd)$ in the case of general MTPPs to only $O(n + d|\mathcal{G}|)$ in the case of GLMTPPs. Furthermore, as the local intensity process can be an expressive functional form, long-range dependencies ignored by neural MTPPs can be captured effectively.

4.2. FastPoint: Scalable Deep Point Processes

We introduce FastPoint [107] as a specific instantiation of GLMTPP, where the global intensity is computed by an LSTM similar to neural MTPPs, and each local model is a HP.

Concretely, we take

$$\begin{aligned}\lambda_k^{(g)}(t) &= \eta(\mathbf{v}_k^\top \mathbf{h}(\mathcal{H}_\tau) + b_k),\\ \lambda_k^{(l)}(t) &= \mu_k + \sum_{\mathcal{H}_t^{(k)}} \varphi_k(t - t_i).\end{aligned}$$

Here, $\mathbf{v}, b_k, \mu_k$ are parameters of appropriate dimension. We use a slight overloading

Table 4.1. Comparison of training and sampling time complexities of MTTPs.

Process	**Conditional Intensity**	**Training Complexity**	**Sampling Complexity**
Poisson	$\lambda_k(t) = \bar{\lambda} p_k$ where $p_k = \bar{\lambda}_k / \bar{\lambda}$	$O(d)$	$O(n+d)$
Hawkes	$\lambda_k(t) = \mu_k + \sum_{\mathcal{H}_t} \varphi_k(t - t_i, y_i)$	$O(n^2 + nk)$	$O(n^2 k)$
RMTPP [58]	$\lambda_k(t) = p(k \vert h(\mathcal{H}_t)) f(t, h(\mathcal{H}_t))$	$O(nd)$	$O(nd)$
Neural Hawkes Process [24]	$\lambda_k(t) = f_k(\mathbf{w}_k^\top h(t))$ $h(t) = o_i \odot (2\sigma(2c(t)) - 1)$	$O(nd)$	$O(nd)$
FastPoint	$g(\mathbf{v}_k^\top \mathbf{h}(\mathcal{H}_\tau) + b_k) + \mu_k + \sum \varphi(t - t_i)$ $\tau = \sup\{\tau' < t \vert \tau' \in \mathcal{G}\}$ $\mathcal{G} = \{0, \Delta, 2\Delta, \dots\}$	$O(n + d\vert\mathcal{G}\vert)$	$O(n + d\vert\mathcal{G}\vert)$

of notation here to highlight that $\mathbf{h}(\mathcal{H}_\tau)$ is an embedding of the process history up to time τ implemented by an LSTM [50], similar to [24, 58]. $\eta : \mathbb{R} \to \mathbb{R}$ is a mapping to the range of the intensity function. In this chapter, η is taken as the *softplus* function, $\eta(x) = \log(1 + e^x)$. φ_k denotes the HP exponential triggering kernel $\varphi_k(x) = \alpha_k \beta_k \exp(-\beta_k x)$, introduced in Section 2.2.2.

FastPoint features individual linearly self-exciting HPs to capture intra-mark effects. Setting $\lambda_k^{(g)}(t) = 0$, for example, the model reduces to a set of independent unidimensional HPs. As demonstrated in our review in Chapter 2, this model has been explored in a wide array of application areas and yields an interpretable model of self-excitatory phenomena. Instead, the global model is meant to capture cross-mark effects, the same ones referred to as *mutual excitation* in the previous chapter. In low-rank processes, our aim was to keep the interpretable graph structure of how such effects are represented while taking advantage of computational simplifications. Here, we rely on the global-local process idea along with deep learning to simplify computation instead of low-rank approximations. This new approach brings other benefits, such as ease of implementation on GPU-ready deep learning libraries where RNNs feature as “first class citizens” while numerical algebra routines are often missing.

Shutting off the local model, in turn, our model is a neural MTPP that’s conditionally a homogeneous Poisson process for each individual mark in the intervals marked by the grid.

The computational advantage comes at the price of losing temporal granularity of how cross-mark effects are represented. Indeed, if two processes “interact” heavily, their “interaction” will have to wait until the synchronization point (the end of the interval), to be passed off to the other process. However, this is a reasonable relaxation for many real-world phenomena. For example, in finance, this encodes the assumption that intra-asset effects can be instantaneous, while “spillover” effects can take some more time. In a social network, a person’s tweets are allowed to self-excite at high temporal resolution while the effect of influencing or triggering other users is modeled at lower resolution.

FastPoint is amenable to scalable likelihood based computation due to the general structure of GLMTPPs, and individual properties of the global and local model choices. The main computational challenge in likelihood-based inference is the computation of the *compensator* $\Lambda(t) = \int_0^t \lambda^*(s)ds$ (cf. Eqn. 2.5). Here due to the additive structure we choose for GLMTPP, and the linearity of integration, this term can be computed as long as the individual compensators of global and local models can be computed exactly. That is,

$$\begin{aligned}\int_0^t \lambda^*(s)ds &= \sum_k \int_0^t \lambda_k^*(s)ds \\ &= \sum_k \int_0^t \left(\lambda_k^{(g)}(s|\mathcal{H}_\tau) + \lambda_k^{(l)}(s|\mathcal{H}_s^{(k)})\right) ds \\ &= \sum_k \Lambda_k^{(g)}(t) + \Lambda_k^{(l)}(t).\end{aligned}$$

This is the main property we exploit for an efficient implementation of FastPoint. The global and local models are chosen such that this property—exact computation of the compensator—is satisfied.

The global model (LSTM) can be changed with other deep NN architectures such as the Transformer [108] or a simple multilayer perceptron. The local model can also be switched with other univariate TPPs whose likelihoods can be computed in linear time. Some examples are HPs with "hyperexponential" kernels, renewal processes, Poisson processes, or a local univariate RMTPP. All of these choices would preserve the favorable computational properties of FastPoint. We give a comparison of asymptotic complexities of likelihood-based inference among different MTPPs in Table 4.1.

4.3. Sequential Monte Carlo Sampling for FastPoint

In this chapter, we describe an efficient sampling routine for GLMTPP, and FastPoint in specific. Namely, we describe an SMC routine that benefits from the global-local structure. We demonstrate that, using Poisson process proposals, we can improve wall-clock time in MTPP simulation by several orders of magnitude.

To start our discussion, let's focus on the main computational challenge in MTPP inference. *Predictive inference* in MTPP requires that one estimates expectations of the form

$$\mathbb{E}\left[\phi\left(\{(t_i, y_i)\}_{(t,t+T]}\right)|\mathcal{H}_t\right],$$

where $\{(t_i, y_i)\}_{(t,t+T]}$ denotes draws–random point sets–from an MTPP. Here, $(t, t+T]$ is the *forecast horizon*, and ϕ denotes some *test function* of the data, *i.e.*, a query to be evaluated. Denoting $\Pi = \{(t_i, y_i)\}_{(t,t+T]}$,

$$\mathbb{E}\left[\phi(\Pi)|\mathcal{H}_t\right] = \int_{\Pi \in \mathcal{X}} \phi(\Pi) f(\Pi|\mathcal{H}) d\Pi,$$

is an intractable integral over $\mathcal{X}$, the *point configuration space* [109], even for trivial functions ϕ such as set cardinality (*i.e.*, the number of points). Hence, MTPP predictions often take the Monte Carlo approach.

Algorithm 6: Sequential Monte Carlo sampling of FastPoint

```
Input: $T$, $\Delta$, $M$, $c$
begin
    $w_j \leftarrow 1, \forall j$
    $\tau \leftarrow t_0$
    while $\tau < T$ do
        for particles $j = 1$ to $M$ do in parallel
            Compute $\bar{\lambda}_k = \lambda^*_k(\tau)$, $\forall k \in \{1, 2, \dots, K\}$
            Draw $N_j \sim \text{Poisson}(\Delta \times \sum_k \bar{\lambda}_k)$
            Draw $\{t_i^{(j)}\}_{i=1}^{N_j} \sim \mathcal{PP}(\sum_k \bar{\lambda}_k)$
            for $i = 1$ to $N_j$ do in parallel
                Draw $y_i^{(j)}$ s.t. $p(y_i^{(j)} = k) \propto \bar{\lambda}_k$
            endfor
            $w_j \leftarrow w_j \times \frac{p(\{(t_i^{(j)}, y_i^{(j)})\} | \mathcal{H}_\tau)}{q(\{(t_i^{(j)}, y_i^{(j)})\} | \bar{\lambda}, p^*)}$
        endfor
        $\tau \leftarrow \tau + \Delta$
        $\text{ESS} \leftarrow \|\mathbf{w}\|_1^2 / \|\mathbf{w}\|_2^2$
        if $ESS < c$ then
            Resample particles, s.t.
            $\{(t_i^{(j')}, y_i^{(j')})\}_{t \in (t_0, \tau]} = \{(t_i^{(j)}, y_i^{(j)})\}_{t \in (t_0, \tau]}$ with prob. $\propto w_j$, $\forall j'$
        end
    end
end
```

Having "approximated" an arbitrary MTPP with FastPoint, we now require an efficient sampling routine to draw point sets, which, as outlined in Section 2.3 is itself a computationally challenging task.

The naive approach to draw random samples from FastPoint would be to implement Ogata's algorithm (Algorithm 1) directly. However, this would result in $O(nd)$

asymptotic complexity—in the number of calls to the random number generator—as all marks would have to be evaluated at each drawn point.

A more careful approach could invoke the conditional independence assumption that (4.1) encodes. Let $\Pi^{(k)}_{(t,s]}$ denote the random point set of mark k in the known interval $(t, s]$. It follows from our definition of GLMTPP that, given the history up to the most recent grid point τ, the draws from the individual processes are independent. That is,

$$p(\Pi_{(t,t+\tau]}|\mathcal{H}_\tau) = \prod_k p(\Pi^{(k)}_{(t,t+\tau]}|\mathcal{H}_\tau).$$

Then, one could sample individual marks in parallel using Ogata's algorithm. Nevertheless, after individual event sets are drawn, these samples have to be "merged" for most interesting probability queries—a generally expensive computation.

Our alternative approach also takes advantage of this conditional independence assumption. Moreover, however, we use importance-weighted sampling in sequence (*i.e.*, SMC) to draw each mark. Most importantly, we draw individual marks from a carefully selected homogeneous Poisson process proposal. Under certain conditions that are easily enforced, we show that Poisson processes make effective proposals. In SMC terms, Poisson processes result in surprisingly high *effective sample sizes* (ESS) for the target distribution of a HP. Finally, the algorithm results in parallelizable $O(n + d)$ calls to the random number generator, greatly accelerating simulation for MTPPs. We give pseudo-code of FastPoint-SMC in Algorithm 6. Here, w_j denotes *particle weights*, κ the resampling threshold, and $\mathcal{PP}$ the Poisson process from which timestamps are drawn. Finally, $p(\{(t_i^{(j)}, y_i^{(j)})\}|\mathcal{H}_\tau), q(\{(t_i^{(j)}, y_i^{(j)})\}|\bar{\lambda}, p^*)$ denote point configuration densities (*i.e.*, likelihoods) with respect to FastPoint and the proposal.

The most important detail is the choice of $\bar{\lambda}$, the constant intensity of the Poisson proposal. We choose $\bar{\lambda}_k = \dfrac{\mu_k + g(\mathcal{H}_\tau)}{1 - \alpha_k}$, the *long-run* intensity of the HP conditioned on the last grid point. This choice partly fulfills our first claim of high ESS, as this is the asymptotically (as $t \to \infty$) optimal Poisson proposal. However, this is not the main reason for which our sampler functions effectively. The most important result of our construction is that the sampler does not break down under high-dimensionality. This counter-intuitive consequence is due to the fact that the number of *marks* that are drawn in each grid interval are much smaller than those for which no points are drawn. That is, marks in which no points are drawn do not contribute to the sampling variance. Then, as long as the expected number of points in an interval is reasonable, the algorithm can scale to arbitrarily high dimensions.

The second claim, of linear time complexity and parallelizable computation, is due to the choice of the *homogeneous* Poisson proposal. First, note that homogeneity ensures that particles do not have to be *thinned* or drawn in sequence. Indeed, drawing from a homogeneous univariate Poisson process is as simple as drawing n exponential random variables (the interarrival times) and scaling them to the appropriate time interval in $O(n)$ time. The Poisson mapping theorem (Theorem 2.3) ensures that the *marks* of points can be drawn independently. These are drawn in parallel from a categorical distribution proportional to the intensity of each mark. As these draws can be achieved with an efficient categorical sampler, such as the *alias method*, we can bound the overall cost of the algorithm at $O(n + k)$.

4.4. Experiments

In our experiments, we compare both computational performance and accuracy of learning and sampling routines developed in this chapter.

One of the main challenges offered by working with MTPPs is the difficulty of implementing learning, as well as sampling, algorithms. Both tasks require sequential processing of variable-length data in batches, and these operations are often not available as first-class citizens of numerical linear algebra or deep learning libraries. We implement neural MTPP models, FastPoint and RMTPP, on Apache MxNet [13]. Specifically, we introduce the `hawkes_ll` operator in the MxNet backend, now available

Table 4.2. Models compared in FastPoint learning experiments.

Model	**Description**
`Hawkes`	An MHP with diagonal kernel matrix (*i.e.*, d independent univariate HP)
`RMTPP`	Recurrent Marked TPP ([58])
`B-RMTPP`	A non-terminating variant of RMTPP with constant baseline intensity
`FastPoint-`Δ	FastPoint with the grid interval length parameter set to Δ

as part of v1.5, to carry out forward and backward passes of the conditional intensity and compensator functions of an HP.[7] For learning, we use MXNet Gluon's Adam optimizer. We run experiments on AWS p3 instances equipped with NVIDIA Tesla V100 GPUs. Further notes on implementation are given in Appendix B.

4.4.1. Model Performance

We compare FastPoint to the set of benchmarks described in Table 4.2. Here, specifically, we modify `RMTPP` to include a background intensity, *i.e.*,

$$\lambda_k(t) = \mu + p(k|h(\mathcal{H}_{t_j}))\exp(\mathbf{v}_\lambda^\top h(\mathcal{H}_{t_j}) + b_\lambda + \beta(t - t_j)),$$

and call the resulting model `B-RMTPP`. This is since the original RMTPP model is a *terminating* TPP, a detail that appears to have been overlooked in the original paper. Although this would only have a minor effect in learning tasks, it renders sequential sampling algorithms invalid, as the process is not absolutely continuous with respect to the Poisson process, Poissonian proposals are invalid in general—with Ogata's thinning, or with importance sampling as we describe here.

We measure the performance of our models on two toy data sets, sampled from an MHP with a full-rank kernel matrix. We also evaluate results on three large-scale high-dimensional point data sets, from social media and an online streaming service.

[7] `github.com/apache/incubator-mxnet/blob/master/src/operator/contrib/hawkes_ll-inl.h`

Table 4.3. Data sets used in FastPoint learning experiments.

Data Set	Description
HP-5K	Forward draw from a 5000-dimensional MHP with a full kernel matrix with $n = 10^6$
HP-10K	Forward draw from a 10000-dimensional MHP with a full kernel matrix with $n = 10^6$
NCEDC	An event log of earthquakes collected from the Northern California Earthquake Data Center earthquake catalog search service [110]. We assign each earthquake to one of 1000 marks based on a clustering of their epicenters.
MemeTracker	The MemeTracker dataset includes 71566 topic clusters and 7.6 million individual events [11].
LastFM-1K	The LastFM dataset contains events belonging to 105K artists.[8]

We also include a data set of seismic events. Descriptions of our data sets can be found in Table 4.3.

The number of hidden units in neural models is fixed at 50 in toy experiments and 100 in on NCEDC, MemeTracker, and LastFM-1K. For RMTPP and FastPoint, we conduct several runs with early stopping and choose the best model based on predictive likelihood on a held-out validation set. Furthermore, we regularize NN parameters (*i.e.*, not the local Hawkes parameters of FastPoint) with weight decay. To prevent numerical issues, we preprocess all data sets to an average rate of 50 events per unit of time.

Our results are presented in Table 4.4, where we give the number of marks and events in each data set, and report predictive log-likelihood on an out-of-sample test set. We find that FastPoint categorically outperforms other models, with a significantly higher margin in real data sets.

Table 4.4. FastPoint predictive performance comparison.

	HP-5K	HP-10K	NCEDC	MemeTracker	LastFM-1K
Events (millions) (n)	1	1	0.8	7.6	18
Marks (d)	5000	10000	1000	71566	105222
Hawkes	27009	30441	10346	42406	25087
RMTPP	27008	30491	14424	42507	30489
B-RMTPP	27008	30483	14393	42304	30474
FastPoint-5	26998	**30412**	10314	**41007**	25271
FastPoint-10	**26997**	30412	10287	41253	25024
FastPoint-20	26998	30412	**10261**	41398	**24500**

In Figure 4.2 we present the contrast between FastPoint and RMTPP in terms of the wall-clock time required for computation, and how this scales in terms of dimensionality. To the left of the figure, we compare the change in training times as the number of marks increases. Clearly, while computation scales linearly in both cases, the difference for computation time in, say, 500K marks is dramatic between FastPoint and competing models. On the right, we compare how learning can be further accelerated, in the presence of even higher dimensionality, by increasing the grid interval length parameter Δ. Combined, FastPoint results in learning times of up to two orders of magnitude faster compared to baselines, while simultaneously offering improved predictive accuracy.

4.4.2. Sampling

In Figure 4.3 we start by comparing the computational performance of FastPoint-SMC, the simulation routine outlined in Section 4.3, to the Ogata algorithm for RMTPP. Specifically, we compare the time taken for sampling 100 points from a fitted model. For now, we do not account for the fact that FastPoint-SMC draws importance-weighted (hence, approximate samples). For 90K points, we can confirm that the SMC routine produces samples 300 times faster than the baseline algorithm.

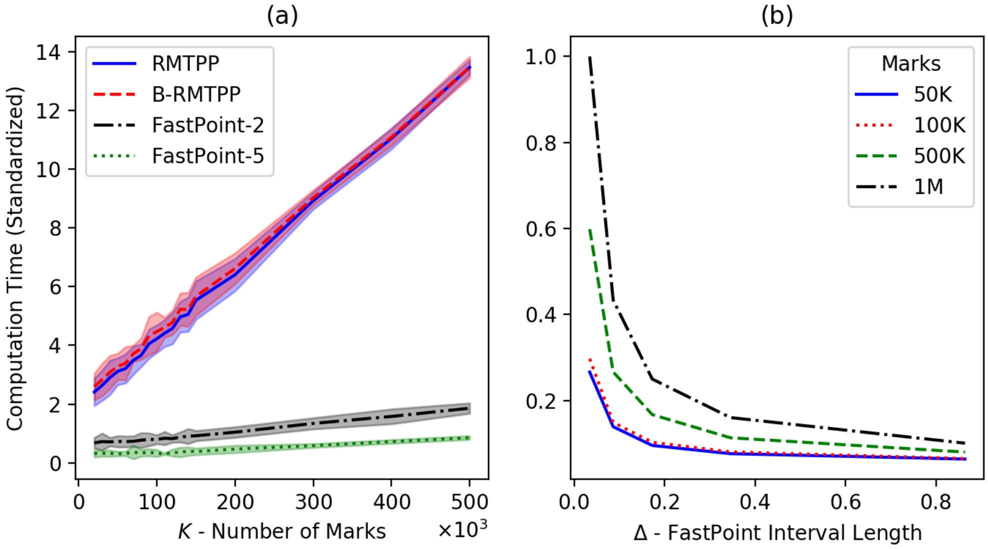


Figure 4.2. Computational results on parameter estimation of FastPoint.

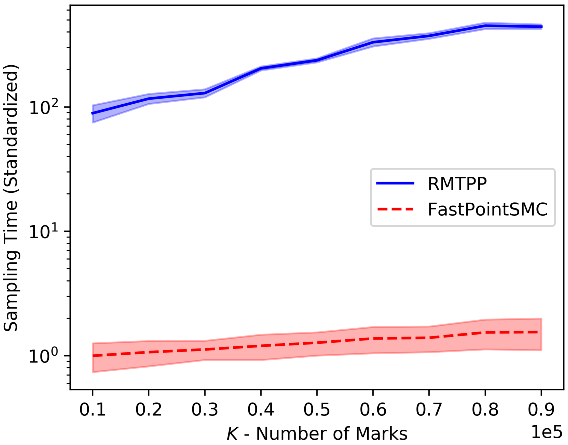
For a fair comparison, however, we must account for the variance introduced by *approximate* sampling. In order to demonstrate that the claims in Section 4.3 hold true, we penalize FastPoint-SMC by *corrected* sample sizes. Namely, we account for the added sampling variance using the ESS, due to Kish [111], defined

$$m_e = \text{ESS} = \frac{\left(\sum_i w_i\right)^2}{\sum_i w_i^2}.$$

To give a comparable statistic of sampling acceleration, we write the effective sampling time multiple,

$$\text{ESTM} = \frac{T_O^{(s)}}{T_{FP}^{(s)}} \times \frac{m_e}{m},$$

where m denotes the number of samples, $T_O^{(s)}, T_{FP}^{(s)}$ times taken to sample from the Ogata and FastPoint-SMC samplers respectively. Intuitively, ESTM computes the approximate wall-clock time that would be taken by FastPoint-SMC for estimators of "equivalent" variance with an exact sampler. In Figure 4.4 (c), we demonstrate that even when penalized for ESS, FastPoint scales favorably with respect to the number of



K $(\times 10^3)$	**Improvement**
10	89.1
20	109.1
30	115.1
40	169.9
50	186.3
60	240.7
70	267.2
80	292.0
90	284.7

Figure 4.3. Results on computational performance of FastPoint-SMC.

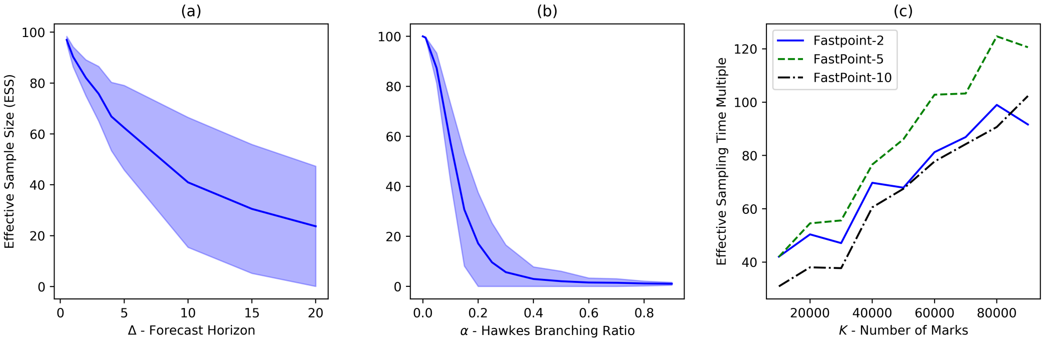


Figure 4.4. Analysis of FastPoint sampling performance.

marks such that it outperforms the baseline by two orders of magnitude.

We owe this result to the efficacy of Poisson proposals, which we scrutinize in the same figure, in plots (a) and (b). In (a) we demonstrate that, as predicted, ESS (scaled to 100) is high for short enough forecast horizons. It starts breaking down after a reasonably high Δ, 5 say–or drawing 250 points in sequence. This also points to a trade-off, which plot (c) also highlights, that there is an optimal Δ around which the sampling time improvement best offsets the loss in ESS. In (b), however, we find that the SMC algorithm fails quickly when the Hawkes branching ratio increases. In practice, this is rarely a problem however, as the model also learns to account for some of the *self*-modulation (within mark) effects also via the global model.

Overall, FastPoint—and GLMTPP in general—appear as a viable alternative to fast and accurate learning in high-dimensional point process data. Apart from accurate learning, FastPoint also provides a practicable solution to predictive inference, a widely unexplored problem in the backdrop of TPPs. Our framework also provides a demonstrably effective way to fuse deep MTPPs with well-studied classical models; retaining the favorable properties of both.

5. INTERMITTENT DEMAND FORECASTING WITH DEEP POINT PROCESSES

In Chapter 3, we focused on bringing high-dimensional MHP learning to scale. Our focus was to estimate a low rank parameterization for MHP, which paved the way to fast learning algorithms. Similarly, in Chapter 4 we approached a similar scalability issue, this time making use of RNNs to approximate the mutual excitation kernel.

In this chapter, we change our course. In contrast to the previous sections, we focus on a practical application of some of the fundamental ideas in Chapter 2. Namely, we focus on the *intermittent demand forecasting* problem, and apply TPPs to learn highly accurate forecasting models.

In summary, IDF is concerned with nonnegative time series characterized by long series of consecutive zero observations [112–116]. For example, a series of daily demand amounts of a product for which demand occurs sporadically. This is illustrated in Figure 5 where an intermittent demand pattern is visible in the time series below. Such sparse time series are well-known to render standard forecasting techniques invalid. They also raise new questions on how forecast accuracy is gauged [117], model selection [118], and model ensembling [119]. Moreover, intermittency often arises in the context of spare parts inventory management in heavy industry, airlines, maritime and defense operations [120], which requires an emphasis on accurate estimates of forecast uncertainty.

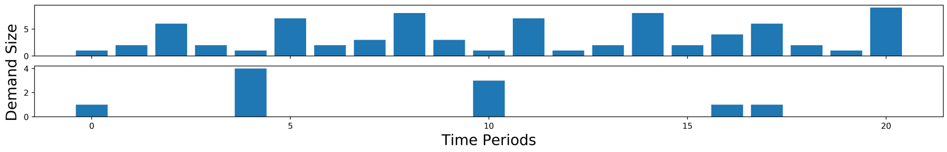


Figure 5.1. Intermittent vs. non-intermittent demand series.

Here, we will build this chapter on the key observation that existing model-based IDF approaches can be expressed as instances of renewal processes. We will then build on this observation, proposing a new set of models including those linked to neural MTPPs, towards IDF tasks. Our contributions in this chapter can be summarized as follows,

- We formalize a connection between previous IDF models and renewal processes and their self-modulated variants.
- We introduce a set of flexible and interpretable stationary renewal process models in discrete time for IDF. We also extend our models to a class of neurally-modulated processes inspired by prior work in machine learning and IDF.
- We give continuous time extensions of our models for when exact timestamps of demand events are available.
- We show substantial improvement of predictive performance on widely available IDF benchmark data.

In Section 5.1 we start by introducing the literature on IDF and set up the fundamental IDF problem. In Section 5.2, we give an argument on why IDF tasks are naturally expressed in the language of renewal theory. In Section 5.2.3 we give extensions of Croston-type IDF models to deep MTPP-like models such as those covered in Section 2.5. We give extensions to a continuous-time analogue of the problem in Section 5.2.4, and numerical results in Section 5.3 concluding the chapter.

5.1. Intermittent Demand Forecasting

Large portions of inventory catalogues, especially in heavy industry, airlines, manufacturing, and defense sectors exhibit *intermittency*—*i.e.*, demands occur sporadically in time. Although the exact definition of intermittent demand is a subject of debate [112, 113, 121–123], its main characteristics are that (i) non-zero demand sizes are observed rarely in time, and potentially (ii) these sizes exhibit overdispersion. In other words, demand is both *intermittent*, and demand amounts are potentially *lumped* in time.

Many classical forecasting heuristics fail in this context, as many such models encode an assumption on "temporal contuniuity" that IDF data does not possess. This was the original observation made by Croston in his seminal paper that started the IDF literature [124]. Namely, Croston observed that *exponential smoothing*—a staple of forecasting literature—was especially prone to failure as it would yield the highest forecasts just after a demand observation, and the lowest just before. In this light, he heuristically proposed to run exponential smoothing on the demand *sizes*—the amount of positive demand, and *intervals*—the number of intervals between consecutive points with nonzero demand. This modeling paradigm underlies most IDF models today, with Croston's method being the de facto industry standard.

Since Croston's original paper, many extensions of the method have been proposed. Schultz [125] investigated whether applying different smoothing constants to sizes and demands would yield higher predictive performance. Croston's results were verified by Willemain *et al.* [126] who also demonstrated that IDF data sets indeed exhibit autocovariance across both size and interval series. An explicit model for inventory control under Croston's model was considered in [127].

Johnston and Boylan proposed TPP models of demand sizes and intervals [128]. Notably, they derived variability estimates of IDF data observed in discrete time, while assuming underlying orders to follow a Poisson process.

In his thesis, Syntetos gave a thorough treatment of the IDF problem [129]. He pointed out an "inversion bias" that resulted from Croston's model assumptions, and gave a revised estimator, since called the Syntetos-Boylan approximation [130]. A simpler version of this approximant was also given in [131]. A similarly modified estimate was studied by Shale *et al.* [132], where the authors assumed underlying Poissonian demand arrivals as in [128]. A review of these estimators was given by [120].

Teunter *et al.* [133] directly modeled the probability of observing nonzero demand. A review and comparison of point-forecast methods in IDF was given by Babai *et al.* [134], in a study that did not post strong evidence in favor of any point forecast method under consideration. Extensions of point forecast methods with Holt-Winters type estimators have been considered in [135, 136].

While point forecast methods are specifically geared towards intermittent data, they do not address another important requirement in IDF settings—accurate forecast uncertainty estimates. Towards developing such estimates, a number of "model-based" approaches, with fully specified generative models for intermittent demand, have been proposed. This issue was first explored by Snyder [137] who considered learning a set of state-space models (SSM) and computing uncertainty estimates from forward samples drawn from the model. This approach, in their terms the "parametric bootstrap," is the approach we will take in this chapter. Notably, Synder drew connections between the renewal-type model proposed by Croston as a justification to his original forecast equations and "single-source-of-error" (SSOE) SSMs.

Shenstone and Hyndman [138] investigated stochastic models built on Croston's assumptions. Specifically, they pointed out that the i.i.d. assumptions made by Croston in defining his models were at odds with the exponential smoothing forecasts proposed. They explored a set of "modified" models where both sizes and intervals (or, their logarithms) followed an IMA(1,1) processes. Most importantly, they showed that such models built on the IMA process assumption suffered from a "convergence problem" (see Appendix C.3), as pointed out in [139, 140].

In their comprehensive article [141], Snyder *et al.* give an extended set of IDF models based on the size-interval formalism. Specifically, they compare i.i.d. (in the paper, "static"), IMA ("undamped dynamic") and AR ("damped dynamic") processes for interdemand intervals. For demand sizes, they consider negative binomial and "zero-inflated" Poisson distributions.

Yelland considered Bayesian treatment of an SSOE SSM model, with a hierarchical Bayesian approach in a follow-up work [142, 143]. Seeger *et al.* [116] gave an approximate Bayesian inference algorithm under a full SSM. Stochastic models for IDF were also explored in [144, 145]. Recently, Svetunkov and Boylan reported improvements with a multiplicative error SSOE SSM [146]. A review of model-based IDF is given by Hyndman *et al.* [147, Ch. 16].

Several works have considered taking a NN-based approach to IDF. Gutierrez *et al.* [148] experimented with feedforward NN to estimate the next period's demand. The network took the previous interdemand interval and size, taken through a single three-node hidden layer. Their method reportedly outperformed Croston's and Syntetos-Boylan methods. These results were later challenged by Kourentzes [149] who compared several NN architectures to Croston's method and Gutierrez's network. Although NNs were not able to outperform classical methods in forecast accuracy metrics, the paper reported improvements in terms of inventory control metrics (*e.g.*, service levels). RNNs were considered for the task [150, 151]. LSTMs were considered recently by Fu *et al.* [152]

5.1.1. Problem Setup

We start our discussion by setting up the notation, the problem, and introducing some of the methods on which we will base our models. We will consider univariate time-series, initially in discrete time—in contrast to the rest of the thesis. Specifically, we will consider nonnegative integer-valued time-series denoted $\{X_t\}_{t\in\{1,2,\cdots\}}, X_t \in \mathbb{N}_{\geq 0}$. While we keep using the variable t, we extend its definition as $t \in \mathbb{N}$. Whether t denotes discrete or continuous time will be clear from the context. Typically, realizations of X_t will contain few cases where $X_t > 0$.

We will benefit from casting the problem in the "size-interval" notation, already suggested by our review of Croston's method in the previous section. Namely, we will index individual points in time where $X_t > 0$ with index $i \in \{1, 2, \cdots\}$—also referred to as “issue points” in the forecasting domain. Concretely, we define the one to one map $\xi(i) = \min\{j \mid i \leq \sum_{t=1}^{j}[X_t > 0]\}$. We can now formally define the demand “sizes” $S_i = X_{\xi(i)}$ and interdemand times $Q_i = \xi(i) - \xi(i-1)$. In other words, we denote by S_i the demand size at the issue point i, while Q_i refers to the number of periods elapsed since the previous issue point. By definition, we have $S_i \geq 1$ and $Q_i \geq 1$. Note finally that (S_i, Q_i) fully determine X_t and vice versa.

We will use $\mathcal{N}, \mathcal{G}, \mathcal{PO}, \mathcal{NB}, \mathcal{E}$ to denote normal, geometric, Poisson, negative binomial, and exponential distributions respectively. These distributions will be parameterized in terms of their means. This is often not the case for the negative binomial distribution, for which we use the mean-dispersion parameterization, introduced in Appendix C.1. Furthermore, we use shifted versions of Poisson, geometric, and negative binomial distributions to accommodate that $S_i, Q_i \geq 1$. The notation $\hat{X}_t$ refers to a (conditional) estimator of X_t while x_t refers to a specific instantiation.

The main aim of our discussion in this chapter is to characterize predictive distributions such as $\mathbb{P}\{X_{t+1:t+m}|X_{1:t} = x_{1:t}\}$. However, we will do so by relying on explicit models of the bivariate process (S_i, Q_i), instead of direct modeling of sequences X_t. Let us start by introducing Croston’s method [124] in our notation. In his original paper, Croston proposed to model demand series by the forecast estimates

$$\hat{Q}_i = g_\alpha(Q_{1:i-1}) \qquad \hat{S}_i = g_\alpha(S_{1:i-1}), \tag{5.1a}$$

where g_α denotes

$$g_\alpha(x_{1:i}) = \alpha x_i + (1-\alpha) g_\alpha(x_{1:i-1}),$$

the EWMA function. Croston also discussed potential models that would give rise to the forecast functions (5.1)

$$S_i \sim \mathcal{N}(\mu, \sigma^2) \text{ i.i.d.,} \tag{5.2a}$$

$$Q_i \sim \mathcal{G}(1/\pi) \text{ i.i.d.} \tag{5.2b}$$

The paper also considers $\{S_i\} \sim \text{ARIMA}(0, 1, 1)$ with Gaussian innovations.

The discrepancy between the model assumptions, which imply a stationary i.i.d. process and forecast functions which model nonstationarity have been the subject of discussion in IDF works that followed. However, most works relied on the forecast functions (5.1) instead of using the model directly. Specifically, the Syntetos-Boylan method corrects for a bias that results from this discrepancy [130]. Yet, it should be pointed out that the bias pertaining to the interdemand times could be handled by a model assumption on the probability of demand occurrence as in [153]. The discrepancy for demand sizes, on the other hand, is often not an issue as

Proposition 5.1. *For $\{S_i\} \sim \mathcal{N}(\mu, .)$, i.i.d., the EWMA series $\{\hat{S}_i\}$ is an asymptotically unbiased estimator of the conditional mean.*

Proof.

$$\mathbb{E}\left[\hat{S}_i | S_{1:i-1}\right] = \mathbb{E}\left[g_\alpha(S_{1:i-1})\right] \longrightarrow \mu = \mathbb{E}\left[S_i\right] = \mathbb{E}\left[S_i | S_{1:i-1}\right], \text{as } i \uparrow \infty.$$

□

However, in this chapter, we take the opposite approach. We base our estimations directly on a learned generative model. For example, working directly with the model proposed in (5.2), we could learn the parameters μ, σ^2, π simply by principle of maximum likelihood. For forecast estimates, we would simply write

$$\mathbb{E}\left[X_t\right] = \mathbb{E}\left[S_{j+1}\right] \mathbb{E}\left[Q_{j+1}\right] = \mu/\pi,$$

Table 5.1. Baseline models for intermittent demand forecasting.

Model	Q_i	S_i
Static G-Po [124]	$\mathcal{G}(\mu_q)$, i.i.d.	$\mathcal{PO}(\mu_m)$, i.i.d.
Static G-NB [141]	$\mathcal{G}(\mu_q)$, i.i.d.	$\mathcal{NB}(\mu_m, \nu_m)$, i.i.d.
EWMA G-Po [138]	$\mathcal{G}(\hat{Q}_i)$ $\hat{Q}_i = g_\alpha(Q_{1:i-1})$	$\mathcal{PO}(\hat{S}_i)$ $\hat{S}_i = g_\alpha(S_{1:i-1})$
EWMA G-NB	$\mathcal{G}(\hat{Q}_i)$ $\hat{Q}_i = g_\alpha(Q_{1:i-1})$	$\mathcal{NB}(\hat{S}_i, \nu_m)$ $\hat{S}_i = g_\alpha(S_{1:i-1})$

where $j = \max\{i|\xi(i) < t\}$. However, this would not be a very useful model in several respects. First, the Gaussian distribution on S_i assigns density to all real values, including negative ones. Second, our model will fail to capture any autocorrelation, an effect that is well-known to be present in IDF data [126]. Finally, the distribution of Q_i imply a Bernoulli process of demand arrivals, *i.e.*, they encode complete independence of demand arrivals across time as a Poisson process would do in continuous time. In our experiments, following the nomenclature of [141], we call such models with i.i.d. S_i, Q_i *static* models.

As introduced in the previous section, the models of Shenstone and Hyndman [138, 147] aim to address some of these shortcomings. Their "modified Croston" models directly parameterize the conditional distributions of size and interval distributions of S_i, Q_i by the EWMA of the observed sequences $S_{1:i-1}, Q_{1:i-1}$ respectively. We call these methods *EWMA* models. We give an overview of static and EWMA models in Table 5.1, where, *e.g.*, "Static G-NB" refers to a static model with the geometric (resp. negative binomial) distribution assumed for interdemand time (resp. size) distributions.

5.2. Discrete Time Renewal Processes

5.2.1. Marked Discrete Time Models

We can now draw a parallel between TPPs and IDF models introduced so far. Above, we noted that for static models, the "arrival process" of nonzero demand followed a stationary Bernoulli process—the discrete analogue of Poisson processes (see, *e.g.*, the argument in [18]). One way in which we can extend IDF models is to view them as marked temporal "point" processes in *discrete time*, where nonzero demand points can be viewed as *point occurrences* and the demand sizes as their marks. Let us note that, although this can appear like an unacceptable stretch of terminology, this is the sense in which modern determinantal point processes [154–156] are used today.

Next, we will observe that we can extend Bernoulli processes in the same way that Poisson processes were extended to renewal processes in 2.1.2. As such, we can define marked discrete-time renewal processes (DTRP). Letting Q_i, S_i, X_t be defined as above,

Definition 5.1. *(Discrete-time renewal process) Q_i defines a DTRP if Q_i $\mathcal{S}(.)$ i.i.d. where $\mathcal{S}$ denotes any probability distribution with support in positive integers.*

Definition 5.2. *(Marked DTRP) Q_i, S_i jointly define a marked DTRP if Q_i defines a DTRP and S_i are distributed according to some mark distribution (not necessarily i.i.d.). When this distribution is independent of Q_i and $S_j, j \neq i$, we will refer to a marked DTRP with* independent marks.

Once again, our use of the term renewal process may appear to bend well-established definitions. However, a short review of early introductions of renewal theory yield that until Cox's introduction of renewal theory in continuous time [19], it was customary to introduce the theory in a discrete-time formalism. See, *e.g.*, Feller [157, Ch.13].

Apart from serving our purposes for introducing new IDF models, the reader may find it of interest that much of the fundamental insights from general renewal theory carry over nicely to a discrete-time setting. Another important connection to IDF is that renewal processes were conceived to determine when machine parts would fail and have to be "renewed," *i.e.*, they were developed toward a nearly identical problem setting to IDF.

This connection also justifies the interpretation of interarrival time distributions in renewal-type models. To make this connection, let us introduce the *hazard rate*[1]

$$h(k) = \mathbb{P}\{Q = k | Q \geq k\} = \mathbb{P}\{Q = k\}/\mathbb{P}\{Q \geq k\} = \frac{\mathbb{P}\{Q = k\}}{1 - \mathbb{P}\{Q \leq k-1\}},$$

where we suppress the index of the interdemand time. Intuitively, the hazard rate defines the probability that a demand will occur in time step k, given that it has not occurred in the previous $k-1$ steps. For geometric interdemand times, it is easy to show that the hazard rate is constant. In other words, the geometric distribution is memoryless, and the probability of a demand occurring is independent of the time since the last occurrence. By a simple change of this distribution assumption, we can encode more interesting behavior such as *aging*—the probability of demand increases as time passes, *clustering*—demand points are lumped together in time, or *quasi-periodicity.* In this paper, we explore DTRP models with a negative binomial interdemand time distribution. We demonstrate that the negative binomial can capture all of these temporal patterns in Figure 5.2. In our experiments, we will replace the geometric distribution in Static models with the negative binomial. We will call these models *Static DTRPs.*

5.2.2. Self-modulating Discrete Time Models

In Table 5.1, our EWMA models differed from their Static counterparts by how the means of conditional size and interval distributions were parameterized. By a similar analogy to the one pointed out in the previous section, we note that these

[1] This is a discrete-time analogue of the *hazard function* of Section 2.1.2

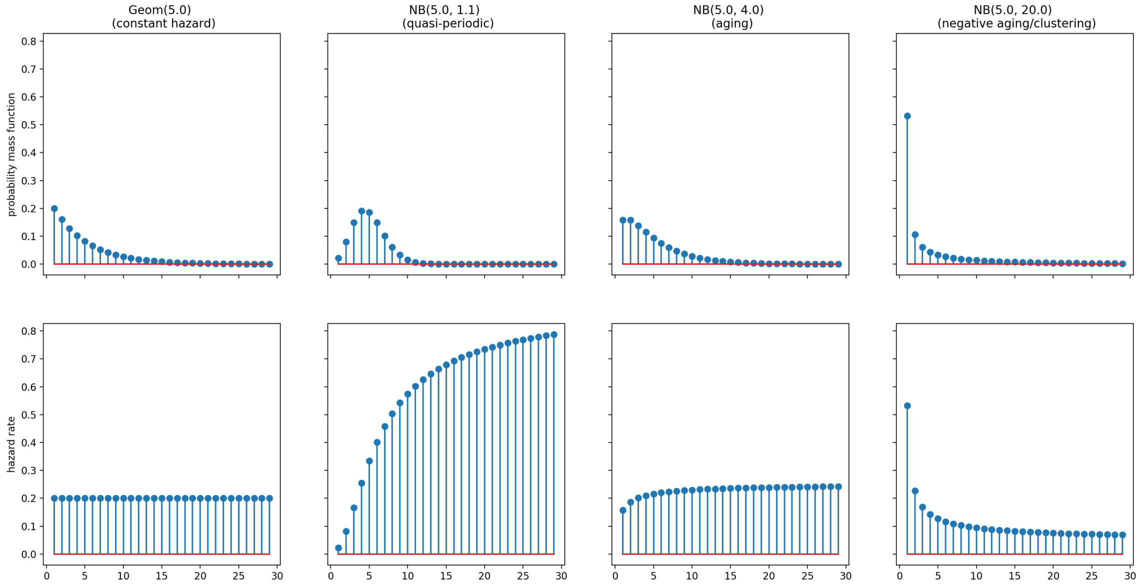


Figure 5.2. Probability functions and hazard rates of negative binomial distributions.

models can be viewed as *self-modulating* DTRPs, just like self-modulating TPPs (see Section 2.2).

For comparison, we consider a natural extension EWMA models to their general DTRP counterparts. Namely, we consider EWMA DTRP models, as presented in Table 5.2.

Before moving on, let's give a theoretical justification of our argument in favor of self-modulating DTRPs. These models are closely related to another extended class of renewal processes—Markov renewal processes (MRP) [158] (see, also, [16, Ch. 10]). Although we will not develop rigorous connections here, our self-modulating DTRP models can be cast as discrete-time Markov renewal processes where the state transitions are deterministic. Stated in terms used widely in the forecasting literature; MRPs could be viewed as multiple source of error models where our self-modulated models correspond to SSOE processes.

Table 5.2. Discrete Time Renewal Process Models.

Model	Q_i	S_i
Static DTRP NB-Po	$\mathcal{NB}(\mu_q, \nu_q)$, i.i.d.	$\mathcal{PO}(\mu_m)$, i.i.d.
Static DTRP NB-NB	$\mathcal{NB}(\mu_q, \nu_q)$, i.i.d.	$\mathcal{NB}(\mu_m, \nu_m)$, i.i.d.
EWMA DTRP NB-Po	$\mathcal{NB}(\hat{Q}_i, \nu_q)$ $\hat{Q}_i = g_\alpha(Q_{1:i-1})$	$\mathcal{PO}(\hat{S}_i)$ $\hat{S}_i = g_\alpha(S_{1:i-1})$
EWMA DTRP NB-NB	$\mathcal{NB}(\hat{Q}_i, \nu_q)$ $\hat{Q}_i = g_\alpha(Q_{1:i-1})$	$\mathcal{NB}(\hat{S}_i, \nu_m)$ $\hat{S}_i = g_\alpha(S_{1:i-1})$
Deep DTRP NB-Po	$\mathcal{NB}(\hat{Q}_i, \nu_q)$ $\hat{Q}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q_{i-1}, S_{i-1})\right)$	$\mathcal{PO}(\hat{S}_i)$ $\hat{S}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q_{i-1}, S_{i-1})\right)$
Deep DTRP NB-NB	$\mathcal{NB}(\hat{Q}_i, \nu_q)$ $\hat{Q}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q_{i-1}, S_{i-1})\right)$	$\mathcal{NB}(\hat{S}_i, \nu_m)$ $\hat{S}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q_{i-1}, S_{i-1})\right)$

5.2.3. Deep Renewal Processes

Our analogies to TPPs, both renewal processes and their self-modulating variants, suggest one final step. We can now recognize that the EWMA is just one of many possible functions that could be used to parameterize conditional distributions of the next size and interval. The EWMA is special only in that it can be computed recursively over the sequence of past observations. A natural replacement for the EWMA is the LSTM [50], which satisfies this property as well as being able to approximate a much wider class of functions.

As such, we define *Deep DTRP* models analogously to EWMA DTRPs, by replacing the EWMA by an LSTM network. While doing this, we also introduce one important change. Namely, we give both the sequence of past sizes and intervals to the LSTM, and let it jointly compute an *embedding* shared by the next size and interval's distribution. Given the computational requirement of computing through an LSTM, this introduces negligible added computational cost. However, as we will see, it yields significant benefits and addresses a key shortcoming in previous models.

Concretely, we define Deep DTRPs via the following relations,

$$\begin{aligned}\hat{Q}_i = f_q(\mathbf{h}_i), \qquad \hat{S}_i = f_s(\mathbf{h}_i),\\ \mathbf{h}_i = \mathrm{LSTM}_\Theta(\mathbf{h}_{i-1}, Q_{i-1}, S_{i-1}).\end{aligned}$$

Here, $f_q(\mathbf{x})$ (resp., f_s) is a suitably defined projection from the LSTM output $\mathbf{h}_i$ at time i to the parameter domain of the respective distributions. For example, we can define f_q as the shifted *softplus* of an affine combination of the LSTM output layer,

$$f_q(\mathbf{h}) = 1 + \log\left[1 + \exp(\mathbf{w}_q^T\mathbf{h} + b_q)\right],$$

where $\mathbf{w}_q, b_s$ are the so-called weight and bias terms for the projection. Note that we shift the result of the softplus function to the interval $[1, \infty)$ to match the parameter domain. Finally, LSTM denotes the LSTM network transformation function introduced in Section 2.4.

In our experiments, we learn parameters Θ of the LSTM network by backpropagating negative log likelihood loss through time. Finally, let us point out that our models can be seen as discrete-time analogues of neural MTPP models [24,58,107], as introduced in Section 2.5.

Our RNN differs from previous attempts in IDF in one important aspect. Instead of approximating the sequence in squared error, our NNs only serve to determine the mean of conditional size and interval distributions. This is similar to the approach taken in recent neural forecasting methods [49]. Second, when multiple intermittent series are available, we learn the parameters Θ globally—across different time series [159,160]. That is, instead of potentially overfitting a model to each individual time series, we attempt to learn a universal function that parameterizes IDF conditional distributions.

Table 5.3. Continuous Time Renewal Processes.

Model	Q'_i	S'_i
Static CTRP E-Po	$\mathcal{E}(\mu_{q'})$, i.i.d.	$\mathcal{PO}(\mu_{s'})$, i.i.d.
Static CTRP E-NB	$\mathcal{E}(\mu_{q'})$, i.i.d.	$\mathcal{NB}(\mu_{s'}, \nu_{s'})$, i.i.d.
Deep CTRP E-Po	$\mathcal{E}(\hat{Q'}_i)$ $\hat{Q'}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q'_{i-1}, S'_{i-1})\right)$	$\mathcal{PO}(\hat{S'}_i)$ $\hat{S'}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q'_{i-1}, S'_{i-1})\right)$
Deep CTRP E-NB	$\mathcal{E}(\hat{Q'}_i)$ $\hat{Q'}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q'_{i-1}, S'_{i-1})\right)$	$\mathcal{NB}(\hat{S'}_i)$ $\hat{S'}_i = f\left(\text{LSTM}_\theta(\mathbf{h}_{i-1}, Q'_{i-1}, S'_{i-1})\right)$

5.2.4. Extensions to Continuous Time

As Croston also observed in his original paper [124], IDF data often arises as a result of "aggregating" individual demand instances, *e.g.*, purchase orders, across a grid in time that is too dense. Moreover, the individual data points for these orders are often themselves available with granular timestamps. This suggests the final set of models which we propose in this chapter, connecting it to the earlier chapters, and modeling intermittent demand with TPPs.

Luckily, our setup and terminology so far make this switch very simple—as easy as changing the support of interval distributions from integers to real numbers. We give these models in Table 5.3, where introduce two main changes. First, we introduce the random sequences Q'_i, S'_i that are analogous to their discrete-time counterparts. However, we stress here that these sequences now index individual demand events, and not only non-zero demand review periods. Our second change carries our models from being DTRPs to TPPs, or continuous-time renewal processes (CTRP). We do this simply by changing the conditional distribution of Q'_i from negative binomial to exponential.

5.3. Empirical Results

We evaluate the proposed models on three data sets collected from numerous IDF scenarios. The `Parts` data set is the standard common task in IDF [120], consisting of 1046 time series of 50 time steps each. Each time series represents the demand for automobile spare parts. The `UCI` retail data set is constructed from real-time demand events [161]. Finally, the `Kaggle` data set includes product-level retail demand data. For the `Kaggle` data set, we only include demand records since 2017, and train on time series samples with 2 points in the conditioning range and 1 in the prediction range.

Our models are implemented on Apache MXNet [13]. We use the MXNet Gluon Adam optimizer to learn single hidden layer LSTMs with 10 hidden units. We map network outputs to the respective parameter ranges via the softplus activation function.

Our models are learned globally. That is, a single set of weights is learned across the different time series (*e.g.*, products). In other words, we seek to fulfill the aim set out in the previous section—of learning a generic EWMA-like function to predict intermittent demand across different individual entities.

For evaluation, we use the *quantile loss* to evaluate probabilistic forecasts. For quantile $\rho \in (0, 1)$ and time series y_t, the ρ-quantile loss is defined as

$$\mathrm{QL}_\rho(y_t, \widehat{y}_t(\rho)) = 2\left[\rho \cdot (y_t - \widehat{y}_t(\rho))[\![y_t - \widehat{y}_t(\rho) > 0]\!] + (1-\rho) \cdot (\widehat{y}_t(\rho) - y_t)[\![y_t - \widehat{y}_t(\rho) \leqslant 0]\!]\right],$$

where $y_t(\rho)$ denotes the quantile estimate from the model. In our experiments, we evaluate such estimates by taking forward samples from fitted models, *i.e.*, by the parametric bootstrap. We compare our models to static and modified Croston models. Table 5.4 summarizes our results.

Deep renewal process models categorically outperform both static and EWMA model variants. The static DTRP idea, of assuming a more flexible interarrival distribution, appears to work at least in work comparably well in most cases. Our numbers regarding Croston-type models (EWMA) are the worst in terms of predictive performance, and our findings echo those of Snyder *et al.* [141].

Finally, continuous-time models can only be tested on the `UCI` data set, where we directly model the underlying demand timestamps instead of rolling demand events up to a temporal grid. Here, we can observe tangible benefit compared to other models. However, the simpler Deep DTRP G-Po model appears to perform favorably in these cases. As such, our numerical results for CTRP models are inconclusive.

Our numerical findings are promising. We can report measurable benefit benefit in using deep RNNs for IDF, contrary to previous attempts. The main differences in the approach, of using a model-based method and using a well-implemented deep learning library, both appear to make a difference. Our results are currently inconclusive about the use of generalized renewal processes, however. Both expanding the set of experiments, and the classes of RNN-based models used in this exercise remain exciting directions for research.

Table 5.4. Results on benchmark IDF datasets.

	P50QL			P90QL		
Model	Parts	Kaggle	UCI	Parts	Kaggle	UCI
Static G-Po [124]	3.643	5.909	12.278	2.495	2.362	7.979
Static G-NB [141]	3.673	5.659	12.282	2.491	2.486	6.651
Static DTRP NB-NB	3.665	5.591	9.469	2.151	2.490	7.382
EWMA G-Po [138, 147]	4.218	6.736	13.453	3.507	2.796	8.371
EWMA G-NB	4.274	7.870	12.242	3.662	3.818	9.779
Deep DTRP G-Po	2.835	5.511	**8.404**	1.601	**2.182**	**4.295**
Deep DTRP G-NB	**2.593**	**5.467**	9.146	**1.517**	2.450	5.380
Deep CTRP G-Po	N/A	N/A	8.817	N/A	N/A	4.727
Deep CTRP G-NB	N/A	N/A	8.847	N/A	N/A	4.863

6. CONCLUSION

Temporal point processes (TPP) are ubiquitous probabilistic models, used for a diverse set of applications from modeling packet arrivals in computer networks to earthquakes. These devices, initially seen as one of the more advanced topics of classical statistics, have been increasingly featured in recent machine learning literature. TPPs not only provide parsimonious and powerful models of "timestamped" data, the object of many tasks in machine learning, they also often yield an interpretable representation.

High-dimensionality is another aspect of large timestamped data that is frequent in machine learning. Indeed, in this context, the "big data" colloquialism not only refers to the number of timestamped "events," but also to their heterogeneity—*i.e.*, the number of different types that they can assume. When examining the logs of a large social media platform, for example, the objective is not to solely understand what temporal patterns drive social media activity. Instead, one seeks to understand how individual "users" collectively give rise to the collection of social interactions.

Nevertheless, both aspects of "big data" pose scalability challenges for learning TPP-based models. General TPPs account for interactions between each pair of events and pair of marks explicitly, leading to a quadratic time complexity in the cardinalities of both sets. This is likely one of the main issues that have hindered the mainstream adoption of TPPs in machine learning. Moreover, predictions from TPPs are almost only available through sampling algorithms, which suffer from the same challenge. In this dissertation, we attempted to bring simple solutions to this challenge.

In our first set of propositions, we looked for a fast learning algorithm for approximating infectivity parameters in high-dimensional MHPs. We proposed two separate approaches, under different constraints and relaxations, that reduce the problem of learning a low-rank MHP to just a low-rank matrix approximation problem.

Our first set of algorithms, NLRHP, leveraged a surprising connection to non-negative matrix approximation problems. Building on this connection, we proposed a simple stochastic gradient descent algorithm—easily implemented on modern deep learning frameworks—to learn a low-rank Hawkes model in linear time. In the second approach, we demonstrated that low-rank MHP learning could be reduced to just a general factorization problem, with a single pass over the data. As such, we were able to accelerate low-rank learning problems—with millions of events and hundreds of marks—to a matter of seconds.

We took a different approach in proposing global-local TPPs. Namely, instead of relaxing the fidelity of the interaction kernel among marks; we relaxed the assumption of temporal granularity in cross-mark interactions. In FastPoint, we showed that one could make up for this relaxation by employing deep RNNs for cross-mark interactions, ubiquitously across a wide range of different real-world data.

Predictive inference in TPPs rely on Monte Carlo methods that require efficient samplers to draw from the fitted process. Alas, the most popular approach—thinning—is a sequential rejection sampler that hardly scales to high-dimensional data. We argued that global-local TPPs naturally suggested a sequential importance sampling approach—not only to significantly improve the time complexity of simulation but also to yield massive parallelizability. We proposed FastPoint-SMC as one of many possible sequential importance sampling algorithms to accelerate the simulation of a general approximate TPPs, and empirically demonstrated that samples of equivalent variance could be obtained at orders of magnitude faster rates.

Collectively, our approaches comprise a set of methods to accelerate learning and inference tasks in general, flexible TPP-based models. They are founded in the well-established theory of TPPs, are categorically faster by multiple orders of magnitude than competing methods, and are easily extensible to more expressive models. They represent *relaxations* of classical MTPP approaches that preserve predictive power while solving the important challenge of scaling to high dimensions.

The efficacy of our methods were illustrated on a number of numerical examples, with data collected from applications in high-frequency finance, seismology, online social media, and music streaming. While these represent only a fraction of domains where TPP data is pervasive, we dedicated the last chapter to a novel application of ideas that naturally carry over from TPPs. That is, we developed a set of models inspired by renewal processes and neural MTPPs to tackle the intermittent demand forecasting problem. The wide set of models proposed all outperform their classical counterparts that have been the go-to models in forecasting for the last half-century.

We hope that our contributions will help further the adoption of TPPs in machine learning, especially in a wide class of applications where "time" is considered an essential variable.

A number of research directions are natural next steps to this work.

The models proposed in Chapter 3 warrant further experimentation. Our initial computational results have been illustrative of their superior computational performance and interpretable results. However, our analysis of computational complexity must be backed by further numerical results. Likewise, the predictive accuracy obtained with low-rank parameters must be tested in real-world prediction tasks.

Global-local TPPs of Chapter 4 yield a flexible framework, and FastPoint's current structure barely scratches the surface of possible combinations of global and local models. Most novel RNN architectures and renewal-type TPPs could replace the global and local components of the framework, yielding powerful domain-specific adaptations. The sampler, FastPoint-SMC, relies on both the GLMTPP approach, and a surprising finding about importance sampling in temporal point processes. A stronger theoretical result could yield more general sampling routines for TPPs, by constructing efficient proposals to arbitrary processes.

The application of both low rank HP and FastPoint to specific domains, especially with web-scale data, remains an exciting pursuit. Examples include time-sensitive collaborative filtering [79], social graph analysis, and community detection [10]. We also believe there will be benefit in applying our models to domains in which TPPs have been considered traditionally, such as high-frequency finance [1, 3] or analysis of neural spike trains [6, 7].

Finally, in Chapter 5, we worked with discrete-time analogues of self-modulating renewal processes for demand forecasting. These models could be applied in other domains where temporally sparse discrete-time series are available. Our models were tested only on benchmark data sets for the intermittent demand task. Testing them on real industrial data sets, where longer contexts are available, remains as a next step.

REFERENCES

1. Cont, R., S. Stoikov and R. Talreja, “A stochastic model for order book dynamics”, *Operations research*, Vol. 58, No. 3, pp. 549–563, 2010.

2. Cont, R., “Statistical modeling of high-frequency financial data”, *IEEE Signal Processing Magazine*, Vol. 28, No. 5, pp. 16–25, 2011.

3. Bacry, E., S. Delattre, M. Hoffmann and J.-F. Muzy, “Modelling microstructure noise with mutually exciting point processes”, *Quantitative Finance*, Vol. 13, No. 1, pp. 65–77, 2013.

4. Türkmen, A. C. and A. T. Cemgil, “Modeling high-frequency price data with bounded-delay Hawkes processes”, *Mathematical and Statistical Methods for Actuarial Sciences and Finance*, pp. 507–511, Springer, 2018.

5. Perkel, D. H., G. L. Gerstein and G. P. Moore, “Neuronal spike trains and stochastic point processes: I. The single spike train”, *Biophysical journal*, Vol. 7, No. 4, pp. 391–418, 1967.

6. Kass, R. E. and V. Ventura, “A spike-train probability model”, *Neural computation*, Vol. 13, No. 8, pp. 1713–1720, 2001.

7. Brown, E. N., R. Barbieri, U. T. Eden and L. M. Frank, “Likelihood methods for neural spike train data analysis”, *Computational neuroscience: A comprehensive approach*, pp. 253–286, 2003.

8. Brown, E. N., R. E. Kass and P. P. Mitra, “Multiple neural spike train data analysis: state-of-the-art and future challenges”, *Nature neuroscience*, Vol. 7, No. 5, pp. 456–461, 2004.

9. Reynaud-Bouret, P., V. Rivoirard and C. Tuleau-Malot, “Inference of functional

connectivity in neurosciences via Hawkes processes", *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, pp. 317–320, IEEE, 2013.

10. Tran, L., M. Farajtabar, L. Song and H. Zha, "NetCodec: Community Detection from Individual Activities", S. Venkatasubramanian and J. Ye (Editors), *Proceedings of the 2015 SIAM International Conference on Data Mining*, pp. 91–99, Society for Industrial and Applied Mathematics, Philadelphia, PA, Jun. 2015.

11. Leskovec, J., L. Backstrom and J. Kleinberg, "Meme-tracking and the dynamics of the news cycle", *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 497–506, ACM, 2009.

12. Türkmen, A. C., "hawkeslib: Fast parameter estimation for simpler Hawkes processes", `https://github.com/canerturkmen/hawkeslib`, 2019.

13. Chen, T., M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang and Z. Zhang, "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems", *arXiv preprint arXiv:1512.01274*, 2015.

14. Alexandrov, A., K. Benidis, M. Bohlke-Schneider, V. Flunkert, J. Gasthaus, T. Januschowski, D. C. Maddix, S. Rangapuram, D. Salinas, J. Schulz *et al.*, "Gluonts: Probabilistic time series models in python", *arXiv preprint arXiv:1906.05264*, 2019.

15. Parzen, E., *Stochastic processes*, Vol. 24, SIAM, 1999.

16. Cinlar, E., *Introduction to stochastic processes*, Courier Corporation, 2013.

17. Daley, D. J. and D. Vere-Jones, *An introduction to the theory of point processes: Volume I: elementary theory and methods*, Springer Science & Business Media, 2007.

18. Kingman, J. F. C., *Poisson processes*, Wiley, 1993.

19. Cox, D. R., *Renewal theory*, Methuen, 1962.

20. Cox, D. R. and V. Isham, *Point processes*, Vol. 12, CRC Press, 1980.

21. Granger, C. W., "Investigating causal relations by econometric models and cross-spectral methods", *Econometrica: journal of the Econometric Society*, pp. 424–438, 1969.

22. Laub, P. J., T. Taimre and P. K. Pollett, "Hawkes Processes", *arXiv:1507.02822 [math, q-fin, stat]*, Jul. 2015, arXiv: 1507.02822.

23. Rasmussen, J. G., "Temporal point processes the conditional intensity function", *Lecture Notes, Jan*, 2011.

24. Mei, H. and J. M. Eisner, "The neural hawkes process: A neurally self-modulating multivariate point process", *Advances in Neural Information Processing Systems*, pp. 6757–6767, 2017.

25. Hawkes, A. G., "Point spectra of some mutually exciting point processes", *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 438–443, 1971.

26. Hawkes, A. G., "Spectra of some self-exciting and mutually exciting point processes", *Biometrika*, Vol. 58, No. 1, pp. 83–90, 1971.

27. Hawkes, A. G., "Hawkes processes and their applications to finance: a review", *Quantitative Finance*, Vol. 18, No. 2, pp. 193–198, 2018.

28. Bacry, E., I. Mastromatteo and J.-F. Muzy, "Hawkes processes in finance", *Market Microstructure and Liquidity*, Vol. 1, No. 01, p. 1550005, 2015.

29. Toke, I. M., "An introduction to Hawkes processes with applications to finance", *Lectures Notes from Ecole Centrale Paris, BNP Paribas Chair of Quantitative Finance*, Vol. 193, 2011.

30. Du, N., L. Song, M. Yuan and A. J. Smola, “Learning networks of heterogeneous influence”, *Advances in Neural Information Processing Systems*, pp. 2780–2788, 2012.

31. Simma, A. and M. I. Jordan, “Modeling events with cascades of Poisson processes”, *arXiv preprint arXiv:1203.3516*, 2012.

32. Zhou, K., H. Zha and L. Song, “Learning Social Infectivity in Sparse Low-rank Networks Using Multi-dimensional Hawkes Processes”, *Artificial Intelligence and Statistics*, 2013.

33. Linderman, S. W. and R. P. Adams, “Scalable bayesian inference for excitatory point process networks”, *arXiv preprint arXiv:1507.03228*, 2015.

34. Linderman, S. and R. Adams, “Discovering latent network structure in point process data”, *International Conference on Machine Learning*, pp. 1413–1421, 2014.

35. Utsu, T., Y. Ogata *et al.*, “The centenary of the Omori formula for a decay law of aftershock activity”, *Journal of Physics of the Earth*, Vol. 43, No. 1, pp. 1–33, 1995.

36. Bacry, E., K. Dayri and J.-F. Muzy, “Non-parametric kernel estimation for symmetric Hawkes processes. Application to high frequency financial data”, *The European Physical Journal B-Condensed Matter and Complex Systems*, Vol. 85, No. 5, pp. 1–12, 2012.

37. Bacry, E., T. Jaisson and J.-F. Muzy, “Estimation of slowly decreasing Hawkes kernels: Application to high frequency order book modelling”, *arXiv:1412.7096 [q-fin, stat]*, Dec. 2014, arXiv: 1412.7096.

38. Achab, M., E. Bacry, S. Gaïffas, I. Mastromatteo and J.-F. Muzy, “Uncovering causality from multivariate Hawkes integrated cumulants”, *The Journal of Ma-*

chine Learning Research, Vol. 18, No. 1, pp. 6998–7025, 2017.

39. Hawkes, A. G. and D. Oakes, “A cluster process representation of a self-exciting process”, *Journal of Applied Probability*, Vol. 11, No. 3, pp. 493–503, 1974.

40. Jovanović, S., J. Hertz and S. Rotter, “Cumulants of Hawkes point processes”, *Physical Review E*, Vol. 91, No. 4, p. 042802, 2015.

41. Veen, A. and F. P. Schoenberg, “Estimation of space–time branching process models in seismology using an em–type algorithm”, *Journal of the American Statistical Association*, Vol. 103, No. 482, pp. 614–624, 2008.

42. Møller, J. and J. G. Rasmussen, “Perfect simulation of Hawkes processes”, *Advances in Applied Probability*, Vol. 37, No. 3, pp. 629–646, Sep. 2005.

43. Da Fonseca, J. and R. Zaatour, “Hawkes process: Fast calibration, application to trade clustering, and diffusive limit”, *Journal of Futures Markets*, Vol. 34, No. 6, pp. 548–579, 2014.

44. Da Fonseca, J. and R. Zaatour, “Clustering and mean reversion in a Hawkes microstructure model”, *Journal of Futures Markets*, Vol. 35, No. 9, pp. 813–838, 2015.

45. Bacry, E. and J.-F. Muzy, “First-and second-order statistics characterization of Hawkes processes and non-parametric estimation”, *IEEE Transactions on Information Theory*, Vol. 62, No. 4, pp. 2184–2202, 2016.

46. Ogata, Y., “On Lewis’ simulation method for point processes”, *IEEE Transactions on Information Theory*, Vol. 27, No. 1, pp. 23–31, 1981.

47. Lewis, P. W. and G. S. Shedler, “Simulation of nonhomogeneous Poisson processes by thinning”, *Naval research logistics quarterly*, Vol. 26, No. 3, pp. 403–413, 1979.

48. Hubel, D. H. and T. N. Wiesel, “Cortical and callosal connections concerned with the vertical meridian of visual fields in the cat.”, *Journal of neurophysiology*, Vol. 30, No. 6, pp. 1561–1573, 1967.

49. Salinas, D., V. Flunkert and J. Gasthaus, “DeepAR: Probabilistic forecasting with autoregressive recurrent networks”, *arXiv preprint arXiv:1704.04110*, 2017.

50. Hochreiter, S. and J. Schmidhuber, “Long short-term memory”, *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

51. Hornik, K., M. Stinchcombe, H. White *et al.*, “Multilayer feedforward networks are universal approximators.”, *Neural networks*, Vol. 2, No. 5, pp. 359–366, 1989.

52. Rosenblatt, F., “The perceptron: a probabilistic model for information storage and organization in the brain.”, *Psychological review*, Vol. 65, No. 6, p. 386, 1958.

53. LeCun, Y., Y. Bengio and G. Hinton, “Deep learning”, *Nature*, Vol. 521, No. 7553, pp. 436–444, May 2015.

54. Abadi, M., P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, “Tensorflow: A system for large-scale machine learning”, *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

55. Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, “PyTorch: An imperative style, high-performance deep learning library”, *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

56. Goodfellow, I., Y. Bengio and A. Courville, *Deep learning*, MIT press, 2016.

57. Zhang, A., Z. C. Lipton, M. Li and A. J. Smola, *Dive into Deep Learning*, 2019.

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

59. Xiao, S., J. Yan, X. Yang, H. Zha and S. M. Chu, "Modeling the intensity function of point process via recurrent neural networks", *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

60. Xiao, S., J. Yan, M. Farajtabar, L. Song, X. Yang and H. Zha, "Joint modeling of event sequence and time series with attentional twin recurrent neural networks", *arXiv preprint arXiv:1703.08524*, 2017.

61. Jing, H. and A. J. Smola, "Neural survival recommender", *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 515–524, ACM, 2017.

62. Cao, Q., H. Shen, K. Cen, W. Ouyang and X. Cheng, "Deephawkes: Bridging the gap between prediction and understanding of information cascades", *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1149–1158, ACM, 2017.

63. Trivedi, R., H. Dai, Y. Wang and L. Song, "Know-evolve: Deep temporal reasoning for dynamic knowledge graphs", *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3462–3471, JMLR. org, 2017.

64. Xiao, S., M. Farajtabar, X. Ye, J. Yan, L. Song and H. Zha, "Wasserstein learning of deep generative point process models", *Advances in Neural Information Processing Systems*, pp. 3247–3257, 2017.

65. Sharma, A., R. Johnson, F. Engert and S. Linderman, "Point process latent variable models of larval zebrafish behavior", *Advances in Neural Information*

Processing Systems, pp. 10919–10930, 2018.

66. Upadhyay, U., A. De and M. G. Rodriguez, “Deep reinforcement learning of marked temporal point processes”, *Advances in Neural Information Processing Systems*, pp. 3168–3178, 2018.

67. Chen, T. Q., Y. Rubanova, J. Bettencourt and D. K. Duvenaud, “Neural ordinary differential equations”, *Advances in neural information processing systems*, pp. 6571–6583, 2018.

68. Jia, J. and A. R. Benson, “Neural jump stochastic differential equations”, *Advances in Neural Information Processing Systems*, 2019.

69. Rubanova, Y., R. T. Chen and D. Duvenaud, “Latent odes for irregularly-sampled time series”, *Advances in Neural Information Processing Systems*, 2019.

70. Omi, T., N. Ueda and K. Aihara, “Fully Neural Network based Model for General Temporal Point Processes”, *Advances in Neural Information Processing Systems*, 2019.

71. Mei, H., G. Qin and J. Eisner, “Imputing Missing Events in Continuous-Time Event Streams”, *International Conference on Machine Learning*, 2019.

72. Linderman, S. W., Y. Wang and D. M. Blei, “Bayesian Inference for Latent Hawkes Processes”, *Advances in Neural Information Processing Systems*, 2017.

73. Rodriguez, M. G. and I. Valera, “Learning with temporal point processes”, *Tutorial at ICML*, 2018.

74. Newman, M. E., “Spectral methods for community detection and graph partitioning”, *Physical Review E*, Vol. 88, No. 4, p. 042822, 2013.

75. Yang, J. and J. Leskovec, “Overlapping community detection at scale: a nonneg-

ative matrix factorization approach", *Proceedings of the sixth ACM international conference on Web search and data mining*, pp. 587–596, 2013.

76. Von Luxburg, U., "A tutorial on spectral clustering", *Statistics and computing*, Vol. 17, No. 4, pp. 395–416, 2007.

77. Ding, C., T. Li and W. Peng, "On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing", *Computational Statistics & Data Analysis*, Vol. 52, No. 8, pp. 3913–3927, 2008.

78. Bacry, E., S. Gaïffas and J.-F. Muzy, "A generalization error bound for sparse and low-rank multivariate Hawkes processes", *arXiv preprint arXiv:1501.00725*, 2015.

79. Du, N., Y. Wang, N. He, J. Sun and L. Song, "Time-sensitive recommendation from recurrent user activities", *Advances in Neural Information Processing Systems*, pp. 3492–3500, 2015.

80. Du, N., *Modeling, learning, and inference of high-dimensional asynchronous event data*, PhD Thesis, Georgia Institute of Technology, 2016.

81. Lemonnier, R., K. Scaman and A. Kalogeratos, "Multivariate Hawkes Processes for Large-scale Inference", *AAAI*, 2017.

82. Dempster, A. P., N. M. Laird and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm", *Journal of the Royal Statistical Society, Series B*, Vol. 39, No. 1, pp. 1–38, 1977.

83. Lewis, E. and G. Mohler, "A nonparametric EM algorithm for multiscale Hawkes processes", *Journal of Nonparametric Statistics*, Vol. 1, No. 1, pp. 1–20, 2011.

84. Paatero, P. and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values", *Environmetrics*,

Vol. 5, No. 2, pp. 111–126, 1994.

85. Lee, D. D. and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization”, *Nature*, Vol. 401, No. 6755, p. 788, 1999.

86. Lee, D. D. and H. S. Seung, “Algorithms for non-negative matrix factorization”, *Advances in neural information processing systems*, pp. 556–562, 2001.

87. Lin, C.-J., “Projected gradient methods for nonnegative matrix factorization”, *Neural computation*, Vol. 19, No. 10, pp. 2756–2779, 2007.

88. Cemgil, A. T., “Bayesian Inference for Nonnegative Matrix Factorisation Models”, *Computational Intelligence and Neuroscience*, Vol. 2009, p. e785152, May 2009.

89. Türkmen, A. C., “A review of nonnegative matrix factorization methods for clustering”, *arXiv preprint arXiv:1507.03194*, 2015.

90. Gaussier, E. and C. Goutte, “Relation Between PLSA and NMF and Implications”, *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 601–602, ACM, New York, NY, USA, 2005.

91. Murphy, K., *Machine Learning A Probabilistic Perspective*, The MIT Press, 2012.

92. Kim, J. and H. Park, *Sparse nonnegative matrix factorization for clustering*, Tech. rep., Georgia Institute of Technology, 2008.

93. Yuan, Z. and E. Oja, “Projective nonnegative matrix factorization for image compression and feature extraction”, *Scandinavian Conference on Image Analysis*, pp. 333–342, Springer, 2005.

94. Ding, C., X. He and H. D. Simon, “On the equivalence of nonnegative matrix factorization and spectral clustering”, *Proceedings of the 2005 SIAM International*

Conference on Data Mining, pp. 606–610, SIAM, 2005.

95. Hunter, D. R. and K. Lange, “A tutorial on MM algorithms”, *The American Statistician*, Vol. 58, No. 1, pp. 30–37, 2004.

96. Meyer, C. D., *Matrix analysis and applied linear algebra*, Vol. 71, Siam, 2000.

97. Duan, X. and B. Zhou, “Sharp bounds on the spectral radius of a nonnegative matrix”, *Linear Algebra and Its Applications*, Vol. 439, No. 10, pp. 2961–2970, 2013.

98. Robbins, H. and S. Monro, “A stochastic approximation method”, *The annals of mathematical statistics*, pp. 400–407, 1951.

99. Türkmen, A. C., G. Çapan and A. T. Cemgil, “Clustering Event Streams With Low Rank Hawkes Processes”, *IEEE Signal Processing Letters*, Vol. 27, pp. 1575–1579, 2020.

100. Bacry, E. and J.-F. Muzy, “Second order statistics characterization of Hawkes processes and non-parametric estimation”, *arXiv preprint arXiv:1401.0903*, 2014.

101. Lancaster, P. and L. Rodman, *Algebraic riccati equations*, Clarendon press, 1995.

102. Duchi, J., E. Hazan and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization”, *Journal of machine learning research*, Vol. 12, No. Jul, pp. 2121–2159, 2011.

103. Shi, J. and J. Malik, “Normalized cuts and image segmentation”, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 22, No. 8, pp. 888–905, 2000.

104. Ng, A. Y., M. I. Jordan, Y. Weiss and others, “On spectral clustering: Analysis and an algorithm”, *Advances in neural information processing systems*, Vol. 2,

pp. 849–856, 2002.

105. Wang, Y., A. Smola, D. C. Maddix, J. Gasthaus, D. Foster and T. Januschowski, "Deep Factors for Forecasting", *International Conference on Machine Learning*, 2019.

106. Maddix, D. C., Y. Wang and A. Smola, "Deep factors with gaussian processes for forecasting", *Third workshop on Bayesian Deep Learning (NeurIPS)*, 2018.

107. Türkmen, A. C., Y. Wang and A. J. Smola, "FastPoint: Scalable Deep Point Processes", *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019.

108. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is all you need", *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

109. Liniger, T. J., *Multivariate Hawkes processes*, Ph.D. Thesis, ETH Zurich, 2009.

110. NCEDC, "UC Berkeley Seismological Laboratory. Dataset. Northern California Earthquake Data Center", , 2014.

111. Kish, L., *Survey sampling*, Wiley, 1965.

112. Williams, T. M., "Stock control with sporadic and slow-moving demand", *Journal of the Operational Research Society*, Vol. 35, No. 10, pp. 939–948, 1984.

113. Eaves, A. H. and B. G. Kingsman, "Forecasting for the ordering and stock-holding of spare parts", *Journal of the Operational Research Society*, Vol. 55, No. 4, pp. 431–437, 2004.

114. Boylan, J. E., A. A. Syntetos and G. C. Karakostas, "Classification for forecasting and stock control: a case study", *Journal of the operational research society*,

Vol. 59, No. 4, pp. 473–481, 2008.

115. Böse, J.-H., V. Flunkert, J. Gasthaus, T. Januschowski, D. Lange, D. Salinas, S. Schelter, M. Seeger and Y. Wang, “Probabilistic demand forecasting at scale”, *Proceedings of the VLDB Endowment*, Vol. 10, No. 12, pp. 1694–1705, 2017.

116. Seeger, M. W., D. Salinas and V. Flunkert, “Bayesian intermittent demand forecasting for large inventories”, *Advances in Neural Information Processing Systems*, pp. 4646–4654, 2016.

117. Hyndman, R. J., “Another look at forecast-accuracy metrics for intermittent demand”, *Foresight: The International Journal of Applied Forecasting*, Vol. 4, No. 4, pp. 43–46, 2006.

118. Kourentzes, N., “On intermittent demand model optimisation and selection”, *International Journal of Production Economics*, Vol. 156, pp. 180–190, 2014.

119. Petropoulos, F. and N. Kourentzes, “Forecast combinations for intermittent demand”, *Journal of the Operational Research Society*, Vol. 66, No. 6, pp. 914–924, 2015.

120. Syntetos, A. A. and J. E. Boylan, “Intermittent demand: estimation and statistical properties”, *Service Parts Management*, pp. 1–30, Springer, 2011.

121. Syntetos, A. A., J. E. Boylan and J. D. Croston, “On the categorization of demand patterns”, *Journal of the Operational Research Society*, Vol. 56, No. 5, pp. 495–503, 2005.

122. Kostenko, A. V. and R. J. Hyndman, “A note on the categorization of demand patterns”, *Journal of the Operational Research Society*, Vol. 57, No. 10, pp. 1256–1257, 2006.

123. Boylan, J. E. and A. A. Syntetos, “Forecasting for inventory management of

service parts", *Complex system maintenance handbook*, pp. 479–506, Springer, 2008.

124. Croston, J. D., "Forecasting and stock control for intermittent demands", *Journal of the Operational Research Society*, Vol. 23, No. 3, pp. 289–303, 1972.

125. Schultz, C. R., "Forecasting and Inventory Control for Sporadic Demand Under Periodic Review", *Journal of the Operational Research Society*, Vol. 38, No. 5, pp. 453–458, May 1987.

126. Willemain, T. R., C. N. Smart, J. H. Shockor and P. A. DeSautels, "Forecasting intermittent demand in manufacturing: a comparative evaluation of Croston's method", *International Journal of forecasting*, Vol. 10, No. 4, pp. 529–538, 1994.

127. Dunsmuir, W. and R. Snyder, "Control of inventories with intermittent demand", *European Journal of Operational Research*, Vol. 40, No. 1, pp. 16–21, May 1989.

128. Johnston, F. R. and J. E. Boylan, "Forecasting for items with intermittent demand", *Journal of the operational research society*, Vol. 47, No. 1, pp. 113–121, 1996.

129. Syntetos, A., *Forecasting of intermittent demand*, PhD Thesis, Brunel University Uxbridge, 2001.

130. Syntetos, A. A. and J. E. Boylan, "On the bias of intermittent demand estimates", *International journal of production economics*, Vol. 71, No. 1-3, pp. 457–466, 2001.

131. Syntetos, A. A. and J. E. Boylan, "The accuracy of intermittent demand estimates", *International Journal of forecasting*, Vol. 21, No. 2, pp. 303–314, 2005.

132. Shale, E. A., J. E. Boylan and F. R. Johnston, "Forecasting for intermittent demand: the estimation of an unbiased average", *Journal of the Operational Research Society*, Vol. 57, No. 5, pp. 588–592, 2006.

133. Teunter, R. H., A. A. Syntetos and M. Z. Babai, “Intermittent demand: Linking forecasting to inventory obsolescence”, *European Journal of Operational Research*, Vol. 214, No. 3, pp. 606–615, 2011.

134. Babai, M. Z., A. Syntetos and R. Teunter, “Intermittent demand forecasting: An empirical study on accuracy and the risk of obsolescence”, *International Journal of Production Economics*, Vol. 157, pp. 212–219, 2014.

135. Altay, N., F. Rudisill and L. A. Litteral, “Adapting Wright’s modification of Holt’s method to forecasting intermittent demand”, *International Journal of Production Economics*, Vol. 111, No. 2, pp. 389–408, 2008.

136. Gamberini, R., F. Lolli, B. Rimini and F. Sgarbossa, “Forecasting of sporadic demand patterns with seasonality and trend components: an empirical comparison between Holt-Winters and (S) ARIMA methods”, *Mathematical Problems in Engineering*, Vol. 2010, 2010.

137. Snyder, R., “Forecasting sales of slow and fast moving inventories”, *European Journal of Operational Research*, Vol. 140, No. 3, pp. 684–699, Aug. 2002.

138. Shenstone, L. and R. J. Hyndman, “Stochastic models underlying Croston’s method for intermittent demand forecasting”, *Journal of Forecasting*, Vol. 24, No. 6, pp. 389–402, Sep. 2005.

139. Grunwald, G. K., K. Hamza and R. J. Hyndman, “Some properties and generalizations of non-negative Bayesian time series models”, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, Vol. 59, No. 3, pp. 615–626, 1997.

140. Akram, M., R. J. Hyndman and J. K. Ord, “Exponential smoothing and non-negative data”, *Australian & New Zealand Journal of Statistics*, Vol. 51, No. 4, pp. 415–432, 2009.

141. Snyder, R. D., J. K. Ord and A. Beaumont, “Forecasting the intermittent demand for slow-moving inventories: A modelling approach”, *International Journal of Forecasting*, Vol. 28, No. 2, pp. 485–496, 2012.

142. Yelland, P. M., “Bayesian forecasting for low-count time series using state-space models: An empirical evaluation for inventory management”, *International Journal of Production Economics*, Vol. 118, No. 1, pp. 95–103, 2009.

143. Yelland, P. M., “Bayesian forecasting of parts demand”, *International Journal of Forecasting*, Vol. 26, No. 2, pp. 374–396, 2010.

144. Dolgui, A., A. Pashkevich and M. Pashkevich, “Bayesian approach to modelling of quasi-periodic intermittent demand”, *IFAC Proceedings Volumes*, Vol. 38, No. 1, pp. 343–348, 2005.

145. Sandmann, W. and O. Bober, “Stochastic Models For Intermittent demands forecasting and stock control”, *Proceedings of the Vienna Conference on Mathematical Modelling, MATHMOD, Vienna, Austria*, Citeseer, 2009.

146. Svetunkov, I. and J. E. Boylan, *Multiplicative state-space models for intermittent time series*, Tech. rep., Lancaster University Management School, 2017.

147. Hyndman, R. J., A. B. Koehler, J. K. Ord and R. D. Snyder, *Forecasting with exponential smoothing: the state space approach*, Springer series in statistics, Springer, Berlin, 2008.

148. Gutierrez, R. S., A. O. Solis and S. Mukhopadhyay, “Lumpy demand forecasting using neural networks”, *International Journal of Production Economics*, Vol. 111, No. 2, pp. 409–420, Feb. 2008.

149. Kourentzes, N., “Intermittent demand forecasts with neural networks”, *International Journal of Production Economics*, Vol. 143, No. 1, pp. 198–206, 2013.

150. Nasiri Pour, A., B. Rostami Tabar and A. Rahimzadeh, "A hybrid neural network and traditional approach for forecasting lumpy demand", *Proc. World Acad. Sei. Eng. Technol*, Vol. 30, pp. 384–389, 2008.

151. Lolli, F., R. Gamberini, A. Regattieri, E. Balugani, T. Gatos and S. Gucci, "Single-hidden layer neural networks for forecasting intermittent demand", *International Journal of Production Economics*, Vol. 183, pp. 116–128, Jan. 2017.

152. Fu, W., C.-F. Chien and Z.-H. Lin, "A hybrid forecasting framework with neural network and time-series method for intermittent demand in semiconductor supply chain", *IFIP International Conference on Advances in Production Management Systems*, pp. 65–72, Springer, 2018.

153. Teunter, R. H. and L. Duncan, "Forecasting intermittent demand: a comparative study", *Journal of the Operational Research Society*, Vol. 60, No. 3, pp. 321–329, 2009.

154. Borodin, A. and G. Olshanski, "Distributions on Partitions, Point Processes, and the Hypergeometric Kernel", *Communications in Mathematical Physics*, Vol. 211, No. 2, pp. 335–358, 2000.

155. Borodin, A., "Determinantal point processes", *The Oxford Handbook of Random Matrix Theory*, Oxford University Press, 2011.

156. Kulesza, A., B. Taskar *et al.*, "Determinantal point processes for machine learning", *Foundations and Trends in Machine Learning*, Vol. 5, No. 2–3, pp. 123–286, 2012.

157. Feller, W., *An introduction to probability theory and its applications*, John Wiley & Sons, 1957.

158. Çinlar, E., "Markov Renewal Theory: A Survey", *Management Science*, Vol. 21, No. 7, p. 27, 1975.

159. Januschowski, T., J. Gasthaus, Y. Wang, S. S. Rangapuram and L. Callot, “Deep Learning for Forecasting: Current Trends and Challenges.”, *Foresight: The International Journal of Applied Forecasting*, Vol. 51, 2018.

160. Januschowski, T., J. Gasthaus, Y. Wang, D. Salinas, V. Flunkert, M. Bohlke-Schneider and L. Callot, “Criteria for classifying forecasting methods”, *International Journal of Forecasting*, 2019.

161. Chen, D., S. L. Sain and K. Guo, “Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining”, *Journal of Database Marketing & Customer Strategy Management*, Vol. 19, No. 3, pp. 197–208, 2012.

162. Shiryaev, A. N., *Probability*, Springer, 1996.

APPENDIX A: Computational Details on Hawkes Processes

Below, we give a tutorial introduction to computing the "memory" of the HP in linear time. Our observations apply in the context of the EM algorithm—when collecting sufficient statistics as well as when computing gradients. Recall the log likelihood of a MHP,

$$\log p(\Pi^{(k)}|\mathcal{H}_\tau) = -\int_0^T \lambda_k^*(s)ds + \sum_{i\in\mathcal{I}_k} \log \lambda_k^*(t_i).$$

Focusing on the first term, the *compensator*, we have

$$\int_0^T \left(\mu_k + \sum_{i|t_i<s} \varphi_k(s-t_i) \right) ds,$$

where the first summand is constant w.r.t. s, and the triggering kernel is defined

$$\sum_{i|t_i<s} \varphi_k(s-t_i) \triangleq \sum_l \phi_{kl} \sum_{t_i\in\Pi^{(l)}_{(0,s]}} \beta \exp(-\beta(s-t_i)).$$

For taking the integral of the second term, we invoke the well-known *memoryless* property of exponential decays. First let us introduce the "memory" process, defined

$$M_k(t) \triangleq \sum_{i\in\Pi^{(k)}_{(0,t]}} \exp\left(-\beta(t-t_i)\right),$$

which can be interpreted as the effect—added intensity—of all events of mark k on time t. Note that the following two recursions hold,

$$M_k(t) = \exp\left(-\beta(t-t_i)\right) M_k(t_i), \tag{A.1a}$$

$$M_k(t_i) = \exp\left(-\beta(t_i-t_{i-1})\right) \left(1 + M_k(t_{i-1})\right), \tag{A.1b}$$

where $t_{i-1}, t_i \in \Pi^{(k)}_{(0,t]}$ are the timestamps of two consecutive points. Moreover, note that

$$\begin{aligned}\int_{\tau}^{t_i} M_k(s)ds &= M_k(t_i)\int_{\tau}^{t_i} \exp\left(-\beta(s-t_i)\right)ds \\ &= \frac{M_k(t_i)}{\beta}\left[1-\exp\left(-\beta(t_i-\tau)\right)\right].\end{aligned}$$

Therefore, invoking (A.1), the compensator can be computed recursively in linear time as

$$\int_0^T \lambda_k^*(s)ds = \mu_k T + \sum_l \phi_{lk}\beta\int_0^T M_l(s)ds.$$

The same idea applies to the computation of the log intensities as

$$\sum_k \sum_{i\in\Pi^{(k)}_{(\tau,\tau+\Delta]}} \log\lambda_k^*(t_i) = \sum_k \sum_{i\in\Pi^{(k)}_{(\tau,\tau+\Delta]}} \log\left[\mu_k + \sum_l \phi_{lk}\beta M_l(t_i)\right].$$

APPENDIX B: Additional Details on Global-Local Temporal Point Processes

B.1. Likelihood Computation

First note that φ_k only depends on points in $\mathcal{H}_t^{(k)}$, *i.e.,* only points of the same mark. The conditional intensity $\lambda_k^*(t)$ depends on points of other marks $l \neq k$ only through the intensity term contributed by the global model $g(\mathbf{v}_k^\top \mathbf{h}(\mathcal{H}_\tau) + b_k)$. Then, letting $\Pi_{(\tau,\tau+\Delta]}^{(k)}$ denote the point process observation between two consecutive points on the grid $\tau \in \mathcal{G}$, it follows that

$$p\left(\bigcup_k \Pi_{(\tau,\tau+\Delta]}^{(k)} \middle| \mathcal{H}_\tau\right) = \prod_k p(\Pi_{(\tau,\tau+\Delta]}^{(k)} | \mathcal{H}_\tau).$$

Indeed, it is this conditional independence assumption that FastPoint exploits in training and sampling.

RMTPP [58] adds an exponential decay to the first term, which still leads to an integral that can be computed analytically. Other non-trivial parameterizations of the RNN, such as in Neural Hawkes Processes [24] lead to an intractable integral requiring a costly numerical approximation. However, since local information is encoded by the Hawkes self-excitation, we take the contribution of the global model as constant between two LSTM epochs.

B.2. Additional Details on FastPoint

The log likelihood of a multivariate point process is

$$\ell(\Theta_{\lambda^*}) = \sum_k \sum_{i \in \mathcal{H}^{(k)}} \log \lambda_k^*(t_i) - \sum_k \int_0^T \lambda_k^*(s) ds. \tag{B.1}$$

In Section 3.1, we defined FastPoint based on the conditional intensity

$$\lambda_k^*(t) = g(\mathbf{v}_k^\top \mathbf{h}(\mathcal{H}_\tau) + b_k) + \mu_k + \sum_{\mathcal{H}_t^{(k)}} \varphi_k(t - t_i), \tag{B.2}$$

where $\mathbf{h}(.)$ denoted the global model, $\mu_k > 0$ the background intensity, and $\varphi_k(x) = \alpha_k \beta_k \exp(-\beta_k(x))$ the scaled exponential delay density. We also defined $\tau = \sup\{\tau' \in \mathcal{G} | \tau' < t\}$, and $\mathcal{G} = \{0, \Delta, 2\Delta \dots\}$ as some uniform "grid" in time, on which the global model (RNN) "clocks" and syncs local processes. In our experiments, we set g as the softplus function for numerical stability, although other options, such as the exponential, are possible and have been used before.

Hawkes parameterization. We take Hawkes parameters $\mu_k, \alpha_k,$ and β_k as constant. A simple extension is conditioning these parameters on $h(\mathcal{H}_\tau)$, and having them vary with respect to the history. However, apart from introducing additional computational cost, this results in losing the interpretability associated with these parameters.

Note that for a well-defined point process we must have $\mu_k, \beta_k, \alpha_k > 0$. We ensure these by parameterizing μ_k and β_k through a nonlinear mapping of a real parameter, *e.g.*, $\mu_k = g(\eta_k)$ where, again, $g(.)$ denotes the softplus function and $\eta_k \in \mathbb{R}$.

Another concern we must address is the issue of *stationarity.* For HPs defined on an unbounded event space, the stationarity condition is given via the branching ratio as $\alpha < 1$. One cannot argue that the stochastic process defined by FastPoint is stationary in the weak sense, since the global intensity leads to nontrivially changing mean measures for every interval defined by the grid $\mathcal{G}$. However, it is not hard to show that $\alpha > 1$ results in nonstationarity through *supercritical* (explosive) branching and can easily lead to numerical problems in practice. To avoid this, we set

$$\alpha_k = (1 - \varepsilon)\sigma(\zeta_k),$$

where $\sigma = 1/(1 + e^{-x})$ denotes the sigmoid function, ε some small positive number, and $\zeta_k \in \mathbb{R}$ parameters to be fitted. This modification ensures the interpretability of α as branching ratios, and ensures stationarity when setting the effect of the global model to zero. Cast as an immigration-birth process, FastPoint can be interpreted as a stochastic process with a constant birth rate and delay density, along with a variable immigration rate that depends nonlinearly on other processes.

Although nonlinear mappings of Hawkes parameters work well in practice, learning them with gradient-based optimization is sensitive to initializations. Luckily, it is easy to compute initial estimates of these quantities from data. We initialize with $\alpha_k = 0.05$, $\mu_k = (1-\alpha_k)|N^{(k)}|/T$ (*i.e.*, the moment-based estimate of μ_k), and $\beta_k = \mu_k$, where $N^{(k)}$ denotes the number of points with mark k in an interval of length T.

Notes on Implementation. Implementing point processes presents a unique challenge with modern deep learning frameworks such as MXNet or TensorFlow: data representation.

Note that a point process observation $\{(t_i, y_i)\}$ is a variable length list of timestamps and discrete marks. In practice, much of the computation on this list is of a sequential nature (scans) (*e.g.*, when computing the likelihood of an HP). To implement these scans in a data parallel manner we operate on point process observations in mini-batches. This results in working with data structures commonly referred to as *ragged* tensors, on which one of the dimensions has a variable length of valid entries. In practice we represent a batch of point process observations as 3-way tensors of dimensions $(T, N, 2)$ where the first dimension refers to the sequence, the second dimension indexes the batch and the last two are “interarrival times” and marks. Note that we do not represent temporal information as absolute points in time, but only as interarrival times in reference to the previous point in the sequence.

Having marks represented in their natural sequence, it is difficult now to reap one of the key computational benefits of FastPoint: the ability to parallelize over marks. Take a partition of the set of marks $\{1, 2, \ldots, K\}$. If one represents sequences belonging to each mark partition in separate tensors, this introduces the difficulty of carrying out an *optimal merge* when a sequential representation of all marks is needed. If represented in the same tensor, one needs an index structure to address different marks in constant time. Both approaches require nontrivial implementations. For invoking both parallelisms (over batch and over marks) at the same time during training, we introduce a custom operator to the MXNet backend that carries out forward and backward computations of HP log likelihoods for K-many independent processes.

B.3. Choices of Global and Local Models

Global Model. In FastPoint, we work with an LSTM with H hidden units ($H \ll K$), that takes each point (interarrival time and a vector embedding of the mark) as an input. Although this requires an LSTM computation at each point, note that it doesn't contribute an $O(NK)$ term to the computational complexity since the LSTM output is mapped to intensity parameters *only* at the beginning of each interval (*i.e.*, points in $\mathcal{G}$).

On the other hand, FastPoint's general construction allows for easy modification of the global model. Although we do not pursue them in our experiments, we give two simple modifications here. First, one could always feed the representations of individual points (interarrival times and mark embeddings) directly into an encoder network, perhaps with positional encodings and attention as in Transformer networks [25]. The resulting *point sequence embedding* could be projected to parameterize each mark's added intensity in the next interval.

Simpler yet, we can observe that the vector $A(\tau) \in \mathbb{R}^K$ comes "for free", and is a natural embedding of the state of the point process. This vector could be transformed to parameterize the global model, for example with dense multilayer networks. This would result in significant computational advantages, and remains an exciting direction for further exploration.

Local Model. Our first choice as a local model in FastPoint is the HP. Specifically, we use a univariate HP with exponential decays, for its favorable properties in likelihood computation. In general, the key computational trick of linear-time likelihood computation exploits the fact that the joint stochastic process followed by (λ^*, t_i) is a Markov process. While this holds for exponential-decay HPs, it could easily be extended to the *hyperexponential* (mixture of exponentials) case setting

$$\varphi_k(x) = \alpha_k \sum_{m=1}^{M} \exp\left(-\beta_k^{(m)}(x)\right).$$

Unfortunately for general delay densities φ, the Markov property does not hold.

One other case where the Markov property would hold trivially is with a (homogeneous) Poisson process as the local model, reducing the intensity function to

$$\lambda_k^*(t) = g(\mathbf{v}_k^\top \mathbf{h}(\mathcal{H}_\tau) + b_k) + \mu_k. \tag{B.3}$$

This simplification leads to no significant computational advantages, although it would likely fail to capture many of the interesting self-excitatory behavior within each mark. However, one possible advantage is that in this trivial case, our SMC routine reduces to exact sampling.

Finally, let us note that RMTPP itself satisfies the Markov assumption in the extended state space $\{(\mathbf{h}_i, t_i)\}$. Indeed, one could have an RNN as the local model in addition to the global model. However, in practice, this setup introduces difficulties in implementation, as discussed above.

APPENDIX C: Appendices to Chapter 5

C.1. Negative Binomial Distribution

Here we briefly recover the mean-dispersion parameterization of the negative binomial distribution. The negative binomial distribution is often defined via a random variable X, with the probability mass function

$$\mathbb{P}\{X = k\} = \binom{k+r-1}{k}(1-\pi)^r \pi^k,$$

where $k \in \{0, 1, 2, \ldots\}, r > 0, \pi \in [0, 1]$. Note that this case, where $r \in \mathbb{R}_{>0}$ has been referred to as the Pòlya distribution, or the *generalized* negative binomial distribution. We first consider "shifting" the distribution to ensure consistency with the rest of our notation, and define $Y \in \{1, 2, \cdots\}$ such that $Y = X + 1$. We then have,

$$\mathbb{P}\{Y = k\} = \binom{k+r-2}{k-1}(1-\pi)^r \pi^{(k-1)}.$$

We also reparameterize the distribution via the mean, and "dispersion." Namely, we define

$$\mu = \frac{\pi r}{1-\pi} + 1 \qquad \nu = \frac{1}{1-\pi},$$

with $\mu > 1, \nu > 1$. It is easy to show that $\mathbb{E}[Y] = \mu$ and that $\mathbb{V}[Y]/(\mathbb{E}[Y] - 1) = \mathbb{V}[X]/\mathbb{E}[X] = \nu$, justifying the "dispersion" interpretation. By the negative binomial distribution $Y \sim \mathcal{NB}(\mu, \nu)$, we refer to the random variable determined by the probability mass function

$$p(k) = \mathbb{P}\{Y = k\} = \binom{k + \frac{\mu-1}{\nu-1} - 2}{k-1} \left(\frac{1}{\nu}\right)^{\frac{\mu-1}{\nu-1}} \left(1 - \frac{1}{\nu}\right)^{(k-1)}.$$

The cdf of Y is

$$F_Y(k) = \mathbb{P}\{Y \leq k\} = 1 - I_{1-1/\nu}\left(k, \frac{\mu - 1}{\nu - 1}\right),$$

where $I_x(a, b)$ is the regularized *incomplete Beta function.*

We can then write the *hazard rate* implied by negative binomial interarrival times,

$$\begin{aligned} h(k) = \frac{p(k)}{1 - F(k-1)} = \frac{F(k) - F(k-1)}{1 - F(k-1)} &= \frac{I_{1-1/\nu}(k-1, r) - I_{1-1/\nu}(k, r)}{I_{1-1/\nu}(k-1, r)} \\ &= 1 - \frac{I_{1-1/\nu}(k, r)}{I_{1-1/\nu}(k-1, r)}, \end{aligned}$$

where we keep $r = (\mu - 1)/(\nu - 1)$ and take $F(0) = 0$.

C.2. Forecast Functions for Static NB Models

Here we characterize the one-step-ahead forecast of static NB models. We are looking to obtain an analytical expression for $\mathbb{E}\left[Y_{n+1}|\mathcal{H}_n\right]$, where we assume information up to time n, denoted $\mathcal{H}_n$, is known. Let $Z_n = [\![Y_n > 0]\!]$ be a binary r.v. that is 1 when time step n is an issue point. We denote the number of demand points in $\mathcal{H}_n$ as i, *i.e.*, $i = \sum_{\nu=0}^{n} Z_\nu$. Then,

$$\mathbb{E}\left[Y_{n+1}|\mathcal{H}_n\right] = \mathbb{E}\left[Z_{n+1}S_{i+1}|\mathcal{H}_n\right] = \mathbb{E}\left[Z_{n+1}|\mathcal{H}_n\right]\mathbb{E}\left[S_{i+1}|\mathcal{H}_n\right] = \mathbb{E}\left[Z_{n+1}|\mathcal{H}_n\right]\mathbb{E}\left[S_{i+1}\right].$$

Here, the second and third equalities follow from our model assumptions: demand sizes and times are conditionally independent and demand sizes are independent of the past. Focusing on the first term,

$$\mathbb{E}\left[Z_{n+1}|\mathcal{H}_n\right] = \mathbb{P}\{Z_{n+1} = 1|\mathcal{H}_n\} = \mathbb{P}\{Z_{n+1} = 1|T_i\}.$$

Note that by definition of renewal processes, the probability of the next issue point depends only on the time of the last demand, here $T_i = \sum_{j=0}^{i} Q_j$. We can rewrite

$$\mathbb{P}\{Z_{n+1} = 1 | T_i\} = \mathbb{P}\{Q_{i+1} = n - T_i + 1 | Q_{i+1} > n - T_i\} = h_Q(n - T_i)$$

where h_Q is the *hazard rate* derived in the previous section. Finally, we have

$$\mathbb{E}\left[Y_{n+1} | \mathcal{H}_n\right] = h_Q(n - T_i)\mathbb{E}\left[S_{i+1}\right].$$

Conditioning on the moving averages (the "levels"), similar expressions can easily be derived for EWMA-type Markov renewal processes. Finally, tools from renewal theory can be used to characterize multi-step forecasts, see *e.g.*, discussions in [157].

C.3. Convergence Problem

For *nonnegative EWMA* models, sample paths converge to zero almost surely. This introduces a significant source of bias, especially in tasks where longer lead times need to be forecasted. In this section, we revisit this issue—the "convergence problem" as it is referred to in literature.

In our introduction of the "size-interval" notation $\{(S_i, Q_i)\}$, we implicitly defined both sequences on the state space of positive integers $\{1, 2, \cdots\}$. We will first explore how this problem manifests in the backdrop of a nonnegative EWMA model defined on S_i. Let

$$S_i - 1 \sim \mathcal{PO}(\hat{S}_{i-1} - 1), \tag{C.1a}$$

$$\hat{S}_i = (1 - \beta)\hat{S}_{i-1} + \beta S_i, \tag{C.1b}$$

where $0 < \beta \leq 1$. In other words, we define the conditional distribution of S_i via its EWMA parameterizing a (shifted) Poisson distribution. Below, we give a statement of the convergence problem and an argument that follows [139] albeit with a slightly more accessible proof for our special case.

Proposition C.1. *Let S_i be defined as above. $S_i \longrightarrow 1$ as $i \uparrow \infty$ almost surely.*

Proof. First note that (C.1b) implies the Markov property on S_i. We have $\mathbb{E}[\hat{S}_i|S_{1:i-1}] = \mathbb{E}[\hat{S}_i|\hat{S}_{i-1}] = \hat{S}_{i-1}$, *i.e.*, $\{\hat{S}_i\}$ is a positive martingale. By the martingale convergence theorem [162, Sec 7.4], $\hat{S}_i \to X$ a.s. for some random variable X. Naturally, as $\hat{S}_i \in [1, \infty)$, $X \in [1, \infty)$. Moreover, it should also follow through (C.1b) that $S_i \to X$ a.s. as well. Then, $S_i - \hat{S}_{i-1} = S_i - \mathbb{E}[S_i] \to 0$ a.s., *i.e.*, S_i converges to a degenerate random variable X. X is degenerate if and only if $\mathbb{V}[X] = X - 1 = 0$ where the first equality follows from the variance of a Poisson distribution, *i.e.*, $S_i \to 1$ a.s. □

For our model definition, the convergence problem takes a different form. Namely, the demand on issue points converges to 1 (and not 0), over long terms. However, note that $Q_i \sim \mathcal{G}(\hat{Q}_{i-1})$ with $\hat{Q}_i$ defined analogously to (C.1b),

Corollary C.1. *$Q_i \longrightarrow 1$ as $i \uparrow \infty$ a.s.*

For the proof, one only needs to note that the variance argument (for a geometric distribution parameterized via its mean) becomes $\mathbb{V}[X] = X(X-1) = 0 \implies X = 1$ as $X > 0$ by definition.

Over long forecast horizons, EWMA model assumptions result in sample paths that converge to dense trajectories (interarrival times of 1 a.s.) with demand sizes 1. It should be intuitive that *Static* models do not suffer from this issue. Moreover, [141] discusses a set of models with stationary mean processes that mitigate the convergence problem. Namely, he redefines the recurrence relation (C.1b) as

$$\hat{S}_i = (1 - \varphi - \beta)\mu + \beta \hat{S}_{i-1} + \varphi S_i,$$

where $\varphi + \beta < 1$, $\varphi, \beta, \mu \in \mathbb{R}_+$. In other words, the mean follows a stationary autoregressive process with mean μ, a parameter which is estimated via maximum likelihood.

Finally, note that the RNNs can approximate an autoregressive recurrence as they can approximate the EWMA. That is, although the network is a black-box, there is no reason to believe that it results inevitably in a convergence issue. Moreover, in theory, they can capture nonstationarities such as trend and seasonality that neither model accommodates.