UNIVERSITY OF CALIFORNIA,
IRVINE

Bayesian Point Process Models with Applications in High-Dimensional Neuroscience Studies

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Statistics

by

Derenik Haghverdian

Dissertation Committee:
Professor Babak Shahbaba, Chair
Professor Annie Qu
Professor Zhaoxia Yu

2023

DEDICATION

Dedicated to my beloved wife, Lernik.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to my advisor, Professor Babak Shahbaba, for his unwavering support throughout my PhD journey at the University of California, Irvine. Professor Shahbaba has consistently served as a source of inspiration, providing me with intellectual freedom to explore and encouraging me at every step of the way. His guidance and mentorship have been instrumental in shaping my research and academic growth.

Furthermore, I extend my sincere appreciation to Annie Qu and Zhaoxia Yu, esteemed members of my defense committee, as well as Norbert Fortin, Hernando Ombao, and Michele Guindani, who served on my advancement committee. Their valuable insights and expertise have been invaluable to the development of my research.

And last but certainly not least, I want to express my heartfelt gratitude to the most important person in my life my remarkable wife, Dr. Lernik Asserian, for her unconditional support throughout my PhD journey. None of the milestones I have achieved throughout these years would have been possible without her constant push and motivation. In moments when doubt clouded my mind and self-confidence wavered, she stood by me as a steadfast pillar of strength.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship (NSF-GRFP) under Grant No. DGE-1839285. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

VITA

Derenik Haghverdian

EDUCATION

Doctor of Philosophy in Statistics **2018-2023**
University of California *Irvine, CA*

Master of Science in Statistics **2018-2020**
University of California *Irvine, CA*

Bachelor of Science in Applied Mathematics **2016-2018**
University of California *Los Angeles, CA*

RESEARCH EXPERIENCE

Graduate Research Assistant **2018–2023**
University of California *Irvine, CA*

TEACHING EXPERIENCE

Statistics Bootcamp for Incoming Graduate Students **2019**
University of California *Irvine, CA*

Volunteer Workshop Leader **2019**
TUMO Center for Creative Technologies *Yerevan, Armenia*

Teaching Assistant **2019**
University of California *Irvine, CA*

PROFESSIONAL EXPERIENCE

Data Science Intern **Summer 2022**
Microsoft *Remote*

Guest Lecturer **Feb 2021, Feb 2022**
UC Irvine STATS 225-Bayesian Statistical Analysis *Irvine, CA*

Data Science Intern **Summer 2021**
Liberty Mutual Insurance *Remote*

REFEREED JOURNAL PUBLICATIONS

Hippocampal ensembles represent sequential relationships among an extended sequence of nonspatial events **2022**
Nature Communications

Is purchasing of vegetable dishes affected by organic or local labels? Empirical evidence from a university canteen **2022**
Appetite

REFEREED CONFERENCE PUBLICATIONS

Empirical Estimation of Privacy of Data Egress **November 2022**
Microsoft MLADS (OneAnalyst Track)

ABSTRACT OF THE DISSERTATION

Bayesian Point Process Models with Applications in High-Dimensional Neuroscience Studies

By

Derenik Haghverdian

Doctor of Philosophy in Statistics

University of California, Irvine, 2023

Professor Babak Shahbaba, Chair

The hippocampus plays a crucial role in organizing the memory of daily events, yet unraveling its mechanisms poses challenges. Decoding the information encoded in the hippocampus is particularly challenging due to sparse neuron activity in non-spatial tasks. However, accurate decoding is crucial for understanding how the hippocampus represents and processes information. This work focuses on decoding neural activity using multivariate point processes, specifically Poisson and Hawkes processes. The first two chapters concentrate on Hawkes processes, which are well-suited for modeling history-dependent phenomena characterized by clustered events, such as the spiking activity in the brain. The study introduces several self-exciting/inhibiting Hawkes process models that effectively capture the interactions among an ensemble of neurons in the hippocampus of rats. The study demonstrates that this approach enables more accurate decoding with significantly lower computational complexity. In the final chapter, we present two novel models for encoding and decoding neural activity using non-homogeneous Poisson processes. The encoding model captures neuronal dynamics in response to stimuli, revealing temporal variations in neuronal activity over time. On the other hand, the decoding model focuses on decoding the underlying stimulus patterns from the spike train data, leveraging the estimated firing rates from the encoding model. These models demonstrate improved decoding accuracy, emphasizing the importance of incorporating temporal dynamics in decoding stimulus patterns from neural activity.

Chapter 1

Introduction

1.1 The Hippocampus and Memory Formation

The hippocampus is a crucial brain region involved in the temporal organization of memory and behavior in humans. It enables us to remember our past experiences [1] [19] [16] and use this information to predict future outcomes [57] [52]. This capacity appears to be conserved across species and applies across spatial and non-spatial modalities. However, the neural mechanisms that support this ability are not well understood. Recent research suggests that the hippocampal network's propensity to generate and preserve sequential patterns of activity may underlie this fundamental capacity [12]. This view is supported by two main lines of electrophysiological evidence. First, hippocampal ensemble activity tends to exhibit sequential firing fields during the presentation of non-spatial stimuli or inter-stimulus intervals. Second, hippocampal neurons have been shown to code for sequences of spatial locations under different experimental conditions [46] [35]. However, there is still a missing piece of evidence directly linking this sequence coding framework with our fundamental ability to remember and predict event sequences across spatial and non-spatial modalities. Specifically, it is critical to demonstrate that these coding properties extend to sequences of non-spatial events unfolding over several seconds, as in daily life episodes, and are linked to the successful retrieval of such event sequences.

1.2 Recording Spiking Activity from the Hippocampus

The recording of hippocampal neuronal activity typically involves the use of specialized electrodes or microelectrode arrays that are implanted in the brains of rats [38]. These electrodes are carefully inserted into specific brain regions, such as the hippocampus, and positioned in close proximity to individual neurons to detect their electrical activity. As the neurons generate action potentials, the electrodes capture and record the corresponding voltage changes, allowing for the precise measurement of the timing and frequency of spike occurrences. Advanced recording techniques, such as multi-channel arrays, offer the capability to simultaneously monitor multiple neurons, enabling neuroscientists to investigate the intricate interactions and network dynamics among different populations of neurons. The resulting spike train data serves as a fundamental resource for exploring the neural mechanisms underlying various cognitive processes, including behavior, learning, and memory in rats. Moreover, this data provides a solid foundation for further analysis and the decoding of brain activity. The spike train data is commonly represented as a binary time series consisting of 1's and 0's, where a 1 indicates the occurrence of a spike and a 0 represents the absence of a spike. Each spike in the train is considered an event that takes place at a specific point in time. The study of such event occurrences has given rise to an extensive literature on *point processes*. Point processes provide a framework for analyzing the timing, intensity, and dependencies of these events, offering valuable insights into the underlying dynamics of neural activity. In this study, our focus is on utilizing point processes to decode the neural activity of an ensemble of neurons recorded from the CA1 region of the hippocampus in rats.

Chapter 2

Background

2.1 Point Processes

Point processes are mathematical objects that model the occurrence of events in a given domain, which is typically either time or space. These stochastic processes models describe the patterns of events, including their spatial or temporal locations, and are used to analyze the underlying mechanisms that govern their occurrence. Point processes can be used to model a wide range of phenomena in different fields, such as earthquakes [43], neuronal activity [53], and financial transactions [28], to name a few.

2.1.1 Fundamentals of Point Processes

We start our discussion by formally defining the concept of a point process.

DEFINITION 2.1. *A sequence of non-negative random variables t_k for $k = 1, 2, \dots$ defined on a probability space $(\Omega, \mathcal{F}, \mathbf{P})$ is called a **point process** on $\mathbb{R}^+$ if for all k, $t_k \leq t_{k+1}$.*

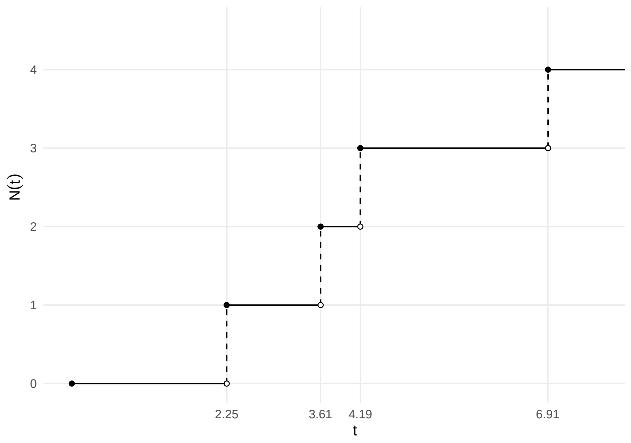


Figure 2.1: An example of a point process realisation with 4 events $\{t_1, t_2, t_3, t_4\}$ and the corresponding counting process $N(t)$.

A point process is referred to as a *temporal point process* when t_k represents the time of occurrence of the events being studied. On the other hand, when t_k denotes the 2-dimensional coordinates of the events in a plane, the point process is called a *spatial point process*.

Another way of characterizing the a point process is through its *counting process* defined below:

DEFINITION 2.2. *A* ***Counting process*** *is a stochastic process* $\{N(t) : t \geq 0\}$ *that takes values in* $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ *that satisfies:*

1. $N(0) = 0$

2. $N(t) < \infty$ *a.s.*

3. $N(t)$ *is a right-continuous step function with increments of size* $+1$

The counting process $N(t)$ counts the number of events up to and including time t.

DEFINITION 2.3. *A point process is called* ***simple*** *when when* $\mathbf{P}(N(t) = 0 \text{ or } 1) = 1$ *for all* t.

Point processes are often characterized using their *Conditional Intensity Function (CIF)*. In fact, if the CIF exists, then it can uniquely characterize the probability distributions of the point process [18]. This function, denoted by $\lambda(t)$, also known as the hazard function, is defined as the expected rate of arrival of events conditioned on the history of the process, denoted by $\mathcal{H}(t)$. The history $\mathcal{H}(t)$ includes all pertinent information about previous events that have occurred up to and including time t over the time interval $(0, t)$.

DEFINITION 2.4. *Let $N(t)$ be a point process with its associated history $\mathcal{H}(t)$. The function $\lambda(t \mid \mathcal{H}(t))$ defined by*

$$\lambda(t \mid \mathcal{H}(t)) = \lim_{h \to 0^+} \frac{\mathbf{P}(N(t+h) - N(t) = 1 \mid \mathcal{H}(t))}{h} \tag{2.1}$$

*is called the **conditional intensity function** of $N(t)$ given the history $\mathcal{H}$, if it exists.*

The CIF $\lambda(t \mid \mathcal{H}(t))$ is a non-negative function that defines the arrival rate of event at time t conditional on the history of the event before time t.

2.1.2 Poisson Processes

The Poisson process is a fundamental point process that serves as a stepping stone for understanding more complex point processes. It is commonly used to model the occurrences of events over time or space, without taking into account the past events or the history of the process. It is of significant importance among point processes due to its mathematical properties and widespread applications in various fields, such as modeling incoming telephone calls, jobs at a computer server, number of machine failures in a factory, etc.

DEFINITION 2.5. *A counting process $N(t)$ is called a **Poisson process** with intensity function (rate) $\lambda(t) \geq 0$ for all t if and only if for $0 \leq t$ and $h \to 0^+$*

1. $N(0) = 0$

2. *The process has independent increments*

3. $\mathbf{P}(N(t+h) - N(t) = 1 \mid \mathcal{H}(t)) = \lambda(t)h + o(h)$

4. $\mathbf{P}(N(t+h) - N(t) > 1 \mid \mathcal{H}(t)) = o(h)$

where the notation $o(h)$ is known as the "little-o" notation and is defined as:

$$\lim_{h \to 0^+} \frac{o(h)}{h} = 0$$

Property (1) states that the process begins with no prior events. Property (2) implies that the number of events that occur in disjoint intervals are independent of each other. Property (3) states that the rate of events is precisely $\lambda(t)$, and property (4) says that the process is orderly.

The probability distribution of the number of events in a Poisson process with intensity function $\lambda(t)$ is given by:

$$\mathbf{P}(N(t) = n) = \frac{\left(\int_0^t \lambda(s)\, ds\right)^n}{n!} \exp\left(-\int_0^t \lambda(s)\, ds\right) \tag{2.2}$$

for $n \in \mathbb{Z}^+$. Hence, $N(t)$ has a Poisson distribution with rate $\int_0^t \lambda(s)\, ds$. Note that when $\lambda(t) = \lambda$ for all t, equation 2.2 simplifies to

$$\mathbf{P}(N(t) = n) = e^{-\lambda} \frac{\lambda^n}{n!} \tag{2.3}$$

which is the probability density of a Poisson random variable with a constant rate. In this case, the process is known as the *Homogeneous Poisson process (HPP)*. The term "homogeneous" refers to the constant rate of events that does not vary over time. An *Non-Homogeneous Poisson process (NHPP)* is the general case where the rate function $\lambda(t)$ varies with time.

2.1.3 Simulating Point Processes

Simulating a point process from a homogeneous Poisson process is a matter of simulating exponential random variables (see [56] for the details) since the inter-arrival times (i.e., time between events) in a homogeneous Poisson process follow an exponential distribution with rate parameter λ. However, simulating data from an inhomogeneous Poisson process is a more complicated task than simulating from a homogeneous Poisson process. It involves generating points sequentially, and several algorithms have been developed for this purpose [15]. The following theorem describes the fundamental approach in simulating data from an inhomogeneous Poisson process:

THEOREM 2.1. *Let $\bar{N}(t)$ be a homogeneous Poisson process with intensity function $\bar{\lambda}$ and a set of events $\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_{\bar{N}(T)}$ that occurred during the time interval $(0, T]$. Suppose that $\lambda(t)$ is a non-negative function that varies over time, where $0 \leq \lambda(t) \leq \bar{\lambda}$ for $0 \leq t \leq T$. To generate an inhomogeneous Poisson process $N(t)$ with rate $\lambda(t)$, we keep an event with probability $\lambda(t)/\bar{\lambda}$ for each $k = 1, \ldots, \bar{N}(T)$, and discard it with probability $1 - \lambda(t)/\bar{\lambda}$. The resulting sequence of events forms an inhomogeneous Poisson process with rate $\lambda(t)$.*

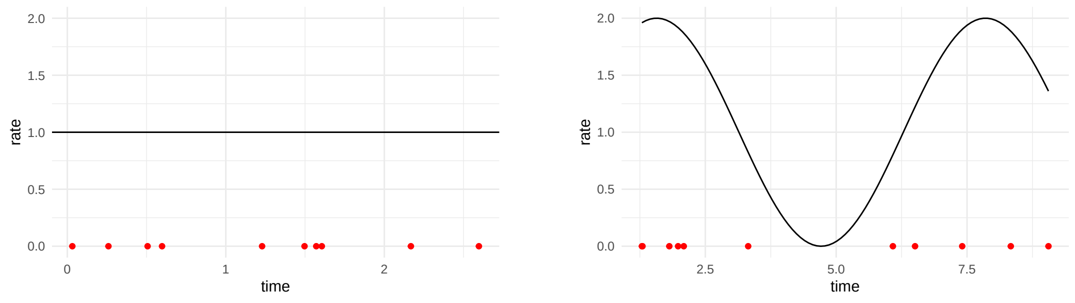
Theorem 2.1 provides the fundamental framework for an algorithm to simulate an inhomogeneous Poisson process. Algorithm 1 called the Lewis' *thinning algorithm* [34] is a step-by-step procedure that generates events from an inhomogeneous Poisson process. Figure 2.2 illustrates simulated data from homogeneous and inhomogeneous Poisson processes.

2.1.4 Parameter Estimation in Point Processes

Efficient estimation of model parameters is critical for accurate modeling and prediction of point process data. Likelihood-based methods are the most widely used techniques for parameter estimation in point processes. The likelihood function is a fundamental component of these methods, and it is dependent on the model's conditional intensity function. Given a particular realization

Algorithm 1 Generating data from a inhomogeneous Poisson process with rate $\lambda(t)$ in the time interval $(0, T]$ via the Thinning Method [34]

```
Require: $T > 0$                                   ▷ Upper bound on the time interval
Require: $\lambda(t) \geq 0$                       ▷ Non-negative rate
 1: $\bar{\lambda} \leftarrow \sup_{0 \leq t \leq T} \lambda(t)$
 2: $E \leftarrow \{\}$                            ▷ The set of events
 3: $t \leftarrow 0$
 4: while $t < T$ do
 5:     $t \leftarrow t + \text{exponential}(\bar{\lambda})$
 6:     $u \leftarrow \text{uniforma}(0, 1)$
 7:     if $u \leq \lambda(t)/\bar{\lambda}$ then  ▷ Accepting with probability $\lambda(t)/\bar{\lambda}$
 8:         $E \leftarrow t \cup E$                ▷ Append the accepted point to the set of events
 9:     end if
10: end while
11: return $E$
```



(a) Homogeneous Poisson process with rate $\lambda = 1$

(b) Inhomogeneous Poisson process with rate $\lambda(t) = 1 + sin(t)$

Figure 2.2: Realizations of Poisson processes using the thinning algorithm in 1 (The rate function is shown with the black line and the events are shown with red dots)

$\{t_i\}_{i=1,\ldots,N(T)=n}$ of a Poisson process $N(.)$ over the time interval $(0, T]$, the log-likelihood can be computed as

$$\mathcal{L} = \mathbf{P}(t_1, \ldots, t_n, N(T) = n) \tag{2.4}$$

$$= \mathbf{P}(N(T) = n)\mathbf{P}(t_1, \ldots, t_n \mid N(T) = n) \tag{2.5}$$

We know $\mathbf{P}(N(T) = n)$ from equation 2.2. We also know from chapter 2 of [18] that given that there are n events in the interval $(0, T]$, these events are i.i.d uniform random variables in $(0, T]$,

$$\mathbf{P}(t_1, \ldots, t_n \mid N(T) = n) \propto \prod_{i=1}^{n} \frac{\lambda(t_i)}{\int_0^T \lambda(s)\, ds} \tag{2.6}$$

Hence, the likelihood function is

$$\mathcal{L} = \exp\left(-\int_0^T \lambda(s)\, ds\right) \frac{(\int_0^T \lambda(s)\, ds)^n}{n!} \prod_{i=1}^{n} \frac{\lambda(t_i)}{\int_0^T \lambda(s)\, ds} \tag{2.7}$$

$$= \exp\left(-\int_0^T \lambda(s)\, ds\right) \prod_{i=1}^{n} \lambda(t_i) \tag{2.8}$$

and the log-likelihood is as follows

$$\log \mathcal{L}\left(\boldsymbol{\theta} \mid \{t_i\}_{i=1,\ldots,N(T)}\right) = \sum_{i=1}^{N(T)} \log \lambda(t_i) - \int_0^T \lambda(s)\, ds \tag{2.9}$$

where $\boldsymbol{\theta}$ contains all of the parameters of the point process model. The term $\Lambda(T) = \int_0^T \lambda(u)\, du$ is known as the *compensator*.

By using the likelihood function given in Equation 2.9, one can calculate the maximum likelihood estimates to determine the model parameters accurately. Alternatively, one can employ Bayesian inference on the model parameters by using appropriate priors, and use the resulting posterior distributions to perform neural decoding.

2.2 Guassian Processes

Gaussian processes (GP) [51] are stochastic processes that provide a powerful framework for modeling complex, non-linear relationships between variables. In this framework, a function is viewed as a random vector that follows a Gaussian distribution over its domain. In fact, Gaussian processes describe distributions over functions and are vastly used as priors over functions in Bayesian statistics. The mean and covariance of this distribution are determined by a set of hyperparameters that can be learned from data. Gaussian processes have found applications in a wide range of fields, from machine learning to finance and physics [5] [25] [24].

DEFINITION 2.6. *A **Gaussian process (GP)** is a stochastic process (a collection of random variables that are indexed by time or space) such that every finite subset of these random variables has a joint multivariate Gaussian distribution.*

A Gaussian process $f(\mathbf{x})$ is completely defined by a mean, $m(\mathbf{x})$, and a covariance function, $k(\mathbf{x}, \mathbf{x}')$.

$$m(\mathbf{x}) = \mathbb{E}(f(\mathbf{x})) \tag{2.10}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}\left[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}) - m(\mathbf{x}))'\right] \tag{2.11}$$

The mean function gives the expected value of the function at any point $\mathbf{x}$, while the covariance function describes the correlation between the function values at two points $\mathbf{x}$ and $\mathbf{x}'$. A Gaussian process is often written as:

$$f \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right) \tag{2.12}$$

which means that f is a Gaussian process with mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$.

A covariance function is called *stationary* if it only depends on the distance between the input points $\mathbf{x}$ and $\mathbf{x}'$. In other words, a stationary covariance function remains unaffected by any trans-

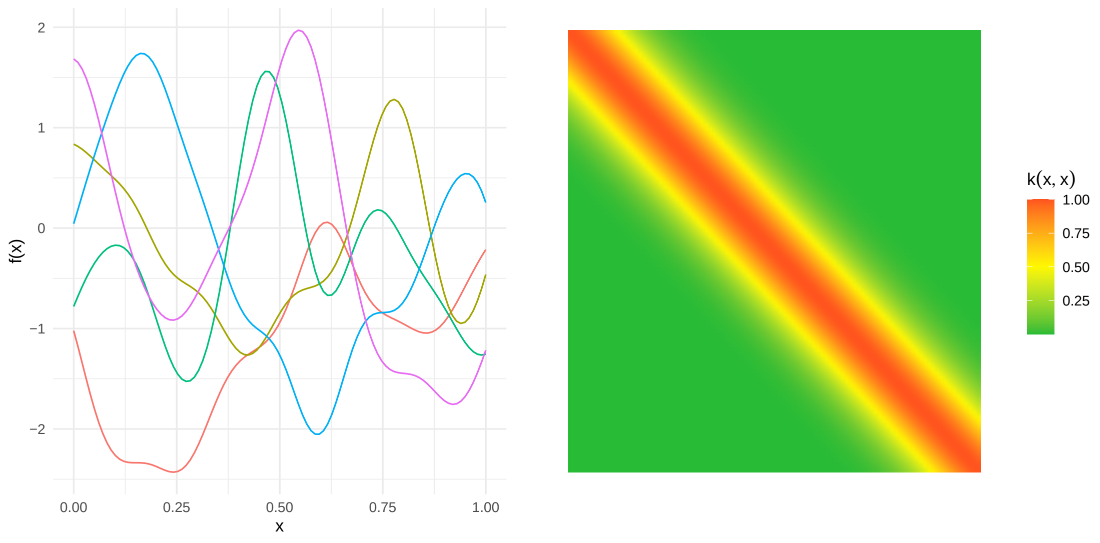


Figure 2.3: Gaussian process with squared exponential covariance function ($\alpha = 1$, $\rho = 0.1$). Left: Five draws from the GP with a zero mean function. Right: The heatmap associated with the squared exponential covariance function.

lations made in the input space. An instance of a stationary covariance function is the *squared exponential* covariance function shown in equation 2.13.

$$\text{cov}\left(f(\mathbf{x}), f(\mathbf{x}')\right) = k(\mathbf{x}, \mathbf{x}') = \alpha^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\rho^2}\right) \tag{2.13}$$

where the parameter ρ is known as the *length-scale* and is related to the frequency of the functions represented by the Gaussian process in the domain. When ρ is closer to zero, the Gaussian process can represent high-frequency functions, while larger values of ρ allow the GP to represent low-frequency functions. On the other hand, the hyperparameter α is referred to as the *marginal standard deviation* and regulates the range of the functions represented by the Gaussian process. Figure 2.3 shows the squared exponential covariance function along with multiple draws from the Gaussian process prior with covariance function parameters $\alpha = 1$ and $\rho = 0.1$. Typically, our goal is not to simply generate random functions from the prior distribution, but rather to take into account the information that the observed data provides about the function. Suppose that we have observed a training dataset denoted by $\{(x_i, f_i) : i = 1, \ldots, n\}$. The joint distribution of the outputs for the observed data set and a test dataset, $\{(x_i^*, f_i^*) : i = n+1, \ldots, n^*\}$, based on the

Gaussian process prior can be expressed as follows:

$$\begin{pmatrix} f \\ f^* \end{pmatrix} \sim \mathrm{N}\left(\mathbf{0}, \begin{pmatrix} k(\mathbf{x},\mathbf{x}) & k(\mathbf{x},\mathbf{x}^*) \\ k(\mathbf{x}^*,\mathbf{x}) & k(\mathbf{x}^*,\mathbf{x}^*) \end{pmatrix}\right) \tag{2.14}$$

where $k(\mathbf{x},\mathbf{x})$ denotes the $n \times n$ matrix of the covariances for the training dataset, $k(\mathbf{x},\mathbf{x}^*)$ denotes the $n \times n^*$ matrix of the covariances between the training and test datasets. Matrices $k(\mathbf{x}^*,\mathbf{x})$ and $k(\mathbf{x}^*,\mathbf{x}^*)$ have similar interpretations. The posterior distribution of the outputs for the test dataset can be computed analytically using the properties of multivariate Gaussian distributions [3]:

$$\mathbf{f}^* \mid \mathbf{x}^*, \mathbf{x}, \mathbf{f} \sim \mathrm{N}\left(k(\mathbf{x}^*,\mathbf{x})k(\mathbf{x},\mathbf{x})^{-1}\mathbf{f}, k(\mathbf{x}^*,\mathbf{x}^*) - k(\mathbf{x}^*,\mathbf{x})k(\mathbf{x},\mathbf{x})^{-1}k(\mathbf{x},\mathbf{x}^*)\right) \tag{2.15}$$

Chapter 3

Multivariate Hawkes Processes for Neural Decoding

3.1 Introduction

Our understanding of how the brain processes information has considerably advanced in recent years due to the rise in novel techniques collecting large multidimensional datasets, such as recordings of the activity of a large number of neurons or brain regions during behavior. However, extracting the key information from such complex datasets poses significant analytical challenges. While stochastic process models provide a very flexible framework for modeling such data, their application to the emerging high-dimensional neuroscience studies is limited due to their lack of scalability. This issue is commonly addressed through dimensionality reduction techniques [17]. In contrast, we propose a flexible stochastic process model based on multivariate Hawkes processes that can easily scale up to high-dimensional settings that help us in the understanding of network organization and function in the brain. An important feature of the multivariate Hawkes processes model is that the interactions between neurons are modeled using the non-symmetric excitation matrix, which enables us to model asymmetric relations with different magnitudes between neurons. This is in contrast to the commonly used covariance matrix approach that limits all interactions to be symmetric and equal in magnitude. Our proposed method is motivated by, and applied to, spiking activity from simultaneously recorded neuronal ensembles in the hippocampus of rats, however it can be extended and applied to a wide range of statistical machine learning problems involving multivariate time series.

The inspiration for this method is an experiment in which neural activity was recorded from the hippocampus of animals performing a complex sequence memory task. The hippocampus, a structure located deep inside the brain, is known to play a key role in the temporal organization of our memories and behaviors [1] [57]. In particular, it is important for remembering past sequences of events as well as planning future sequences of actions or decisions. However, the specific hippocampal mechanisms supporting this capacity remain unknown. Understanding these neural mechanisms requires the ability to decode the information represented in neuronal ensembles using a technique known as *neural decoding*, which is challenging task since it typically involves modeling high-dimensional and sparse spike trains. When developing decoding methods involving neuronal interactions, the excitation/inhibition across neurons is a crucial factor that must be considered [21] [62] [44]. In addition, the model should account for the influence of the history of the process on its future. This is particularly true for brain activity data, where history-dependent neuronal behavior defined by brief spike sequences with shorter inter spike intervals [30], called *bursting*, are common. [61] partially addressed these aspects of the neural data by a framework known as point-process generalized linear models (PP-GLM). The limitation of the PP-GLM approach is that it models the behavior of a selected neuron and its interactions with others (not all neurons simultaneously) using a discrete approximation of a continuous time point processes. In this paper, we address these issues by introducing a new neural decoding method based on multivariate Hawkes process (MHP) models with self- and cross-exciting/inhibiting characteristics.

Hawkes processes [27] are stochastic processes that are commonly used to model history-dependent self-exciting/inhibiting phenomena such as earthquakes [43] and trade orders [28]. MHPs are point processes in which the occurrence of an event in one marginal process influences the rate of occurrence of events in the other marginal processes by different amounts. Hawkes processes (point processes in general) are uniquely characterized by their time-varying non-negative conditional intensity function (CIF), usually denoted by $\lambda(t)$. It is through the CIF that the details of the point process such as dependence on history and mutual excitation in the marginal processes are modeled. The majority of the literature focuses on self-exciting Hawkes processes since they are easily interpretable and usually modeled with linear CIFs that have low computational cost. Self-inhibition, on the other hand, is modeled by decreasing the CIF for a period of time, and since there is a chance that the CIF could become negative after multiple self-inhibiting events, the CIF is usually passed through a non-linear link function such as the Sigmoid or the Rectified Linear Unit (ReLU). These non-linear CIFs have quadratic computational complexity and do not scale to high-dimensional problems. Our proposed method discussed in the Methods section keeps the computation of CIF linear in a novel way by using separate parameters for self-excitation and self-inhibition. In addition, this new parameterization allows for cross-excitation and cross-inhibition among neurons while keeping the computational cost linear in terms of dimension.

The remainder of this chapter is organized as follows. We will first provide a brief overview of multivariate Hawkes process (MHP), the regularity conditions for its existence, and the required log-likelihood computation. Later, we formally introduce the flocking Hawkes process models – our extended, reparameterized, and computationally efficient variants of MHP. After showing the simulation results, we present the rat electrophysiology experiment and evaluate our proposed models in terms of their neural decoding accuracy. Lastly, we discuss future directions and possible extensions to improve our method.

3.2 Background

3.2.1 Multivariate Hawkes Process

A multivariate Hawkes process (MHP) [27], also known as a mutually exciting point process, is a *simple* (in the sense that no two events occur at the same time) multivariate counting process $\mathbf{N}(t) = (N_1(t), \ldots, N_M(t))$, where $N_m(t),\ m = 1, 2, \ldots, M$ is the m-th marginal counting process that counts the number of events of type m up to time t. The MHP, $\mathbf{N}(t)$, satisfies the following conditions:

(i) $N_m(0) = 0$ for $m = 1, 2, \ldots, M$.

(ii) The conditional intensity function (CIF) defined by

$$\begin{aligned} \lambda_m\left(t \mid \mathcal{H}_t\right) &= \lim_{h \downarrow 0} \frac{\mathbb{E}\left[N_m(t, t+h] \mid \mathcal{H}_t\right]}{h} \\ &= \mu_m + \sum_{n=1}^{M} \int_0^t \nu_{m,n}(t-s) N_n(ds) \end{aligned} \tag{3.1}$$

is a non-negative left-continuous function where $\mathcal{H}_t$ is the history of the process up to time t, $\mu_m \in \mathbb{R}^+$ is the background intensity of the m-th marginal process, and $\nu_{m,n} : \mathbb{R}^+ \to \mathbb{R}^+,\ m, n = 1, 2, \ldots, M$, is called the excitation/triggering function.

(iii) For each $m = 1, 2, \ldots, M$,

$$\mathbf{P}(N_m(t, t+h] = 1 \mid \mathcal{H}_t) = \lambda_m(t \mid \mathcal{H}_t)h + o(h) \tag{3.2}$$

that is, $\lambda_m\left(t \mid \mathcal{H}_t\right)$ is the CIF of the marginal process $N_m(t)$.

(iv) The process is *orderly*; that is, for $m = 1, 2, \ldots, M$,

$$\mathbf{P}\left(N_m(t, t+h] \geq 2 \mid \mathcal{H}_t\right) = o(h). \tag{3.3}$$

The CIF in Equation (3.1) can be interpreted as the instantaneous expected rate of the arrival of events at time t conditioned on $\mathcal{H}_t$. A common and interpretable choice for the triggering function is the exponential decay function. Hawkes originally [27] used this function since it simplified the derivations of the likelihood computations by introducing a recursive pattern which decreased the computational complexity from $O(n^2)$ to $O(n)$, where n is the total number of events [14]. In this case, the CIF of the m-th marginal process has the following form:

$$\lambda_m\left(t \mid \mathcal{H}_t\right) = \mu_m + \sum_{n=1}^{M} \sum_{\{i:t_{n,i}<t\}} \nu_{m,n}(t - t_{n,i}) \tag{3.4}$$

where $\nu_{m,n}(t) = \alpha_{m,n} e^{-\beta_{m,n} t}$. For each $n = 1, 2, \ldots, M$, $t_{n,i}$ is the i-th event in the n-th marginal process prior to time t, $\alpha_{m,n}$, $m, n = 1, 2, \ldots, M$, is the amount by which the CIF of the m-th marginal process increases instantaneously when an event occurs in the n-th marginal process, and $\beta_{m,n}$ for $m, n = 1, 2, \ldots, M$, is the amount by which this instantaneous increase decays over time. So, the half-life of this decay over time is proportional to $\beta_{m,n}^{-1}$. Note that the MHP has $M + M^2 + M^2$ parameters corresponding to μ_m, $\alpha_{m,n}$, and $\beta_{m,n}$, respectively. Thus, the number of parameters in MHP is $O(M^2)$.

3.2.2 Stationarity Conditions

An MHP with exponential decay triggering function is said to be *stationary* whenever the spectral radius of the branching matrix is strictly less than 1; that is, $\rho\left(\mathbf{\Gamma}\right) = \max\{|\boldsymbol{\lambda}_1|, \ldots, |\boldsymbol{\lambda}_M|\} < 1$ where $\mathbf{\Gamma} = \{\alpha_{m,n}/\beta_{m,n}\}$, and $\rho(.)$ denotes the spectral radius of the branching matrix $\mathbf{\Gamma}$ with eigenvalues $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_M$. The stationarity condition assures that the Hawkes process is non-exploding; that is, the total number of events in any time interval is bounded. The stationarity condition is specifically important when simulating data according to an MHP. We elaborate more on this in the Simulation Study section.

3.2.3 Likelihood Function and Posterior Inference

For a particular realization $\boldsymbol{\omega}$ of an MHP with event times $\{t_{n,i}\}_{i=1,\ldots,N_n(T),\ n=1,2,\ldots,M}$ observed on the time interval $(0, T]$, the log-likelihood can be computed as

$$\log \mathcal{L}\left(\boldsymbol{\theta} \mid \boldsymbol{\omega}\right) = \sum_{m=1}^{M} \left\{ \int_0^T \log \lambda_m(t \mid \boldsymbol{\omega}) N_m(dt) - \int_0^T \lambda_m(t \mid \boldsymbol{\omega})\, dt \right\} \tag{3.5}$$

where $\boldsymbol{\theta} = (\boldsymbol{\mu}, \mathbf{A}, \mathbf{B})$ contains the parameters of the MHP model, where $\mathbf{A}$ and $\mathbf{B}$ are $M \times M$ matrices with $\alpha_{m,n}$ and $\beta_{m,n}$ as the components of the m-th row and n-th column, respectively.

Combining the likelihood function in Equation (3.5) with appropriate priors for the model parameters $\boldsymbol{\theta}$, we can conduct Bayesian inference on the model parameters and use the resulting posterior distributions for neural decoding. To this end, we could use sampling algorithms such as Hamiltonian Monte Carlo [41] or Variational Bayes (VB) [29]. While we have implemented our models using both methods, the results presented in this paper are based on the latter.

3.2.4 Variational Bayes inference

A probabilistic model defines a joint probability distribution, $p(\mathbf{x}, \boldsymbol{\theta})$, over the observed data, $\mathbf{x}$, and a set of unobserved model parameters, $\boldsymbol{\theta}$. The task of inference is to compute the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{x})$, considering the prior distribution $p(\boldsymbol{\theta})$ on the model parameters. In many cases, the posterior $p(\mathbf{x}, \boldsymbol{\theta})$ has a very complicated form, which highlights the necessity of approximate inference methods. VB is a method that uses optimization to approximate complex posterior distributions by considering a member, $q(\boldsymbol{\theta})$, in a family of densities, $\mathcal{Q}$, that is closest, in Kullback-Leibler divergence [33], to the target (posterior) density. Let

$$q^*(\boldsymbol{\theta}) = \underset{q \in \mathcal{Q}}{\arg\min}\, \mathbb{KL}\left(q(\boldsymbol{\theta}) \mid\mid p(\mathbf{x} \mid \boldsymbol{\theta})\right), \tag{3.6}$$

where

$$\mathbb{KL}\left(q(\mathbf{z}) \mid\mid p(\mathbf{x} \mid \mathbf{z})\right) = -\int q(\mathbf{z}) \log \frac{p(\mathbf{x} \mid \mathbf{z})}{q(\mathbf{z})}\, dx \tag{3.7}$$

Compared to Markov Chain Monte Carlo (MCMC) methods [22] [26] [41], the dominant method of inference in Bayesian statistics, VB methods are much faster and easier to scale to big-data settings. The main VB algorithm is known as the *mean-field* algorithm, which assumes independence among the variational parameters,

$$q(\boldsymbol{\theta}) = \prod_i q_i(\theta_i), \tag{3.8}$$

where q_i is the marginal distribution for $\theta_i \in \boldsymbol{\theta}$ (see [4]).

3.3 Methods

As mentioned in the Multivariate Hawkes Process section, the amount by which the CIFs increase after each event and the amount of interaction among the processes (neurons) in the MHP model are controlled by the off-diagonal elements in the excitation matrix $\mathbf{A}$, and the decay rates in the CIFs are controlled by the decay matrix $\mathbf{B}$. It is common to model inhibition through the $\alpha_{m,n}$ parameters by letting them to be negative [6]. However, this leads to non-linear CIF functions that are computationally expensive since the CIF must remain a non-negative function. To address this issue, we propose an alternative parameterization where excitation (self and cross) and inhibition (self and cross) are absorbed separately through $\mathbf{A}$ and $\mathbf{B}$, respectively. In our approach, the $\alpha_{m,n}$ parameters are non-negative and are only responsible for the self- and cross-excitation among processes. On the other hand, $\beta_{m,n}$ parameters control the inhibition effects. Therefore, the proposed approach has the following parameterization:

$$\alpha_{m,n} \geq 0, \quad \beta_{m,n} = \begin{cases} \eta_m^{\text{self}} & m = n \\ \\ \eta_m^{\text{self}} + \eta_{m,n} & m \neq n \end{cases} \tag{3.9}$$

for $m, n = 1, \ldots, M$, where η_m^{self} and $\eta_{m,n}$ are non-negative. That is, each off-diagonal element in matrix $\mathbf{B}$ is the sum of the corresponding diagonal element and an extra term, $\eta_{m,n}$. Since these additive $\eta_{m,n}$ are non-negative, they can only increase the decay amount by forcing the CIF to decay faster to its base intensity and model inhibition effects. In other words, $\eta_{m,n}$ is the additive amount in the decay rate for neuron m caused by a spike in neuron n. This way, we account for primary feature of inhibition in complex neuronal networks [21], while ensuring CIFs remain linear and non-negative. We refer to the resulting model as flocking Hawkes process (FHP). In what follows, we discuss three different alternatives for capturing the connections (flocking) among the processes (neurons) through specifications of $\mathbf{A}$ and $\mathbf{B}$.

3.3.1 Sparse FHP (sFHP)

One possible approach is to allow for fully connected neurons while imposing sparsity-encouraging priors such as the horseshoe [48] on the off-diagonal elements of $\mathbf{A}$ to force sparse estimates of the parameters that correspond to interaction among neurons. This is a reasonable assumption since not all neurons interact with each other with the same magnitude. Note that we only put the horseshoe prior on the excitation parameters since based on the form of the exponential triggering function, if the $\alpha_{m,n}$ parameters shrink to zero, then the entire triggering function shrinks to zero and the $\beta_{m,n}$ parameters become irrelevant.

Suppose the off-diagonal elements of $\mathbf{A}$ are collected in the vector $\tilde{\boldsymbol{\alpha}}$ that has length equal to $K = M(M-1)$, where K is the total number of off-diagonal elements in $\mathbf{A}$. For $k = 1, \ldots, K$, the prior on these parameters has the following form:

$$\begin{aligned} \tilde{\alpha}_k | \xi_k, \tau &\overset{\text{ind}}{\sim} \text{Half-Normal}(0, \tau^2 \tilde{\xi}_k^2), \qquad \tilde{\xi}_k^2 = \frac{c^2 \xi_k^2}{c^2 + \tau^2 \xi_k^2} \\ \xi_k &\overset{\text{ind}}{\sim} \text{Half-Cauchy}(0, 1) \\ \tau &\sim \text{Half-Cauchy}(0, 1) \\ c^2 &\sim \text{Inv-Gamma}(a, b) \end{aligned} \tag{3.10}$$

where τ, also known as the global shrinkage parameter, shrinks all of the $\tilde{\alpha}_k \in \tilde{\boldsymbol{\alpha}}, k = 1, \ldots, K$, parameters to zero, while $\xi_k, k = 1, \ldots, K$, known as the local shrinkage parameters, help certain $\tilde{\alpha}_k$ parameters to escape from shrinkage.

3.3.2 Clustered FHP (cFHP)

In many real applications, the multivariate processes have an underlying structure that can be expressed as clusters or, in general, graphs. For example, in a typical electrophysiological experiment, multiple neurons are recorded using a single tetrode (four tiny electrodes that are twisted together and are used to record the extracellular field potentials) forming a natural clustering of neurons. The above method can easily be extended to preserve such structures. To illustrate, suppose that M neurons are associated with G clusters. Each cluster includes certain number of neurons (one or more). Under these assumptions, the excitation and decay matrices, $\mathbf{A}$ and $\mathbf{B}$, respectively, have a block-diagonal form such that each block matrix corresponds to a cluster. That is, $\mathbf{A} = \text{block-diag}\,(\mathbf{A}_1, \ldots, \mathbf{A}_G)$ and $\mathbf{B} = \text{block-diag}\,(\mathbf{B}_1, \ldots, \mathbf{B}_G)$ where $\mathbf{A}_g$ and $\mathbf{B}_g$, $g = 1, \ldots, G$, are $M_g \times M_g$ matrices, where M_g is the number of neurons associated with cluster g. Note that $M = \sum_{g=1}^{G} M_g$.

The inhibition effects are incorporated into each block similar to the structure in Equation (3.9).

Thus, for $g = 1, \ldots, G$, and $m, n = 1, \ldots, M_g$, the components of $\mathbf{A}_g$ and $\mathbf{B}_g$ are as follows:

$$\alpha^g_{m,n} = \begin{cases} \gamma^{g,\text{self}}_m & m = n \\ \gamma^g_{m,n} & m \neq n \end{cases} \qquad \beta^g_{m,n} = \begin{cases} \eta^{g,\text{self}}_m & m = n \\ \eta^{g,\text{self}}_m + \eta^g_{m,n} & m \neq n \end{cases} \tag{3.11}$$

for non-negative $\gamma^{g,\text{self}}_m$, $\gamma^g_{m,n}$, $\eta^{g,\text{self}}_m$, and $\eta^g_{m,n}$.

Similar to the sFHP model structures, we put a sparsity inducing regularized horseshoe prior [48] on the off-diagonal elements of each $\mathbf{A}_g$ (off-diagonal elements in each block of the block-diagonal matrix $\mathbf{A}$) to force sparse estimates of the parameters that correspond to cross-excitation among neurons within a cluster.

3.3.3 Pooled FHP (pFHP)

As mentioned in the Multivariate Hawkes Process section, the number of parameters in a fully connected MHP model grows quadratically with M, the dimension of the MHP, making it infeasible for high-dimensional settings. Alternatively, we can impose a simplifying assumption for the way neurons (marginal processes) are interacting with each other. In contrast to the two models discussed above, a more computationally feasible approach, yet still biologically realistic, is to assume that the firing pattern of each neuron is tightly linked to that of a group (assembly) of other neurons. This influence can be efficiently captured by aggregating the simultaneous activity of other neurons in the ensemble, rather than accounting for each binary (one-on-one) interaction. In this approach, an ensemble of neurons influences any specific neuron through their collective and aggregated firing patterns. This assumption is motivated by our understanding of coding properties of large ensembles, specifically of cell assemblies [11], and is consistent with the principles of hyperdimensional computing models of brain function [49].

In this model structure, we connect each neuron to all the other neurons. More specifically, in computing the CIF of neuron m, we aggregate all of the spike times of the rest of the $M-1$ neurons into a single point process (similar to superposition of Poisson processes) and compute the CIF as if we have a bivariate Hawkes process, (i.e. neuron m connected to all other neurons combined). The following parameterization combined with Equation (3.4) creates this aggregate (pooling) effect. Thus, the components of $\mathbf{A}$ and $\mathbf{B}$ are as follows:

$$\alpha_{m,n} = \begin{cases} \gamma_m^{\text{self}} & m = n \\ \gamma_m^{\text{others}} & m \neq n \end{cases}, \qquad \beta_{m,n} = \begin{cases} \eta_m^{\text{self}} & m = n \\ \eta_m^{\text{self}} + \eta_m^{\text{others}} & m \neq n \end{cases} \tag{3.12}$$

for non-negative γ_m^{self}, γ_m^{others}, η_m^{self}, and η_m^{others}. In this parameterization, each neuron has its own unique self-excitation and self-decay parameters, but the cross-excitation and cross-decay parameters are shared. Note that this parameterization is $O(M)$ in number of parameters.

3.3.4 Identifiability of the Parameters

In the different parameterizations that we introduced above, the η_m^{self} parameters are updated by and linked to the m-th marginal process, while the parameters $\eta_{m,n}$ are informed by the rest of the marginal processes. This is analogous to the global-local shrinkage parameters discussed in [50], where the regression coefficients are assumed to be distributed as a continuous scale mixture of Gaussian distributions, i.e. $\beta_i|\tau, \lambda_n, \sigma^2 \sim \mathcal{N}(0, \sigma^2 \cdot \tau^2 \cdot \lambda_i^2)$ for $i = 1, \ldots, n$. Here, $\tau \in \mathbb{R}^+$ denotes the *global* shrinkage parameter, while the vector $\lambda_n = \{\lambda_i\}_{i=1}^n,\ \lambda_i \in \mathbb{R}^+$ contains all the *local* shrinkage parameters. In the Bayesian setting, the global shrinkage parameter is informed by all of the data, whereas the local shrinkage parameters are informed by each feature separately.

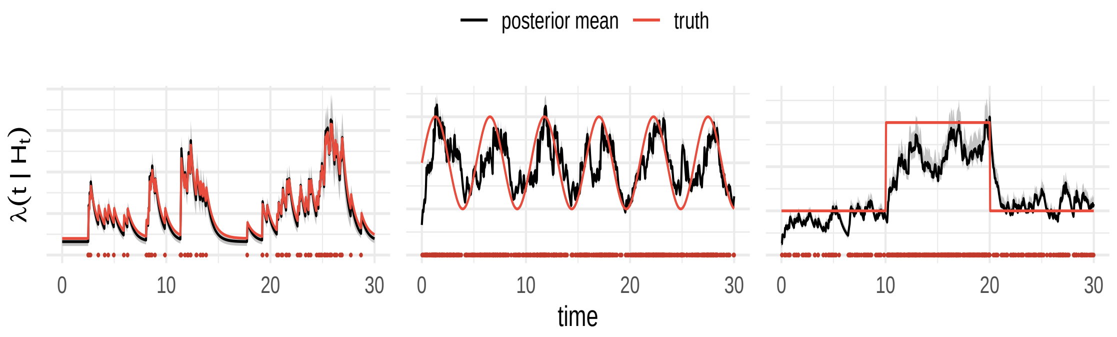


Figure 3.1: Posterior distribution of the CIF for three different univariate data generating mechanisms. Univariate Hawkes process simulated with Ogata's [42] thinning algorithm (left), non-homogeneous Poisson process with periodic intensity function (center), and non-homogeneous Poisson process with a step intensity function (right). The events/spikes are shown with red points under each subplot.

3.4 Simulation Study

In this section, we present some simulations to illustrate the proposed model structures introduced in the Methods section. To start, we simulate data under three different data generating mechanisms for single neurons and fit the data using a univariate Hawkes process model. Figure 3.1 demonstrates that the fitted posterior CIFs capture the general functional form of the true intensity functions (true data generating mechanisms) in all three cases. When the data are generated according to the Hawkes model (Figure 3.1 left), the posterior mean of the CIF is almost identical to the true CIF. In cases where the data are generated from a non-Hawkes model, such as non-homogeneous Poisson processes with different intensity functions (Figure 3.1 center and right), the model still approximates the overall behavior of the true CIFs.

Next, we generate MHP data using Ogata's Modified Thinning Algorithm [42]. In particular, we simulate a 3-tetrode 6-neuron network over the $(0, 30]$ time interval. As mentioned above, the neurons associated with each tetrode form a cluster. Here, there are 1, 2, and 3 neurons, in each tetrode, respectively; i.e., $G = 3, M = 6, M_1 = 1, M_2 = 2, M_3 = 3$. The model parameters are drawn from the following distributions:

$$\begin{aligned} \mu_m &\overset{\text{iid}}{\sim} \text{uniform}(1.5, 2) \\ \alpha^g_{m,n} &\overset{\text{iid}}{\sim} \text{exponential}(1) \\ \beta^g_{m,n} &\overset{\text{iid}}{\sim} \text{exponential}(1) \end{aligned} \tag{3.13}$$

for $m, n = 1, \ldots, M_g$ and $g = 1, \ldots, G$.

As mentioned in the Multivariate Hawkes Process section, in order for the simulated Hawkes process to be stationary, the parameters $\alpha_{m,n}$ and $\beta_{m,n}$ should be such that the branching matrix, $\mathbf{\Gamma}$, has a spectral radius strictly less than 1; i.e. $\rho(\mathbf{\Gamma}) < 1$. We consider the following two approaches both of which perform similarly:

(i) Generate several sets of candidate parameters, and then select the set that satisfied the stationarity condition.

(ii) Generate a set of candidate parameters, and then normalize the corresponding branching matrix, $\mathbf{\Gamma}$, such that the Frobenius norm is less than 1 since we have $\rho(\mathbf{\Gamma}) \leq \|\mathbf{\Gamma}\|_F < 1$, where $\|.\|_F$ denotes the Frobenius norm.

Because of the underlying structure of the data, we fit the cFHP model. Here, we use Automatic Differentiation Variational Inference (ADVI) [32] in Stan [13] with weakly informative log-normal priors on the base intensity, self-excitation, and decay parameters.

Note that the posterior distribution of the parameters in the MHP model should also satisfy the stationarity condition. For that, we sample parameters from the approximate mean-field variational distribution and base our posterior inference only on the subset of these samples that satisfies the stationarity condition. Note that, the majority of the posterior samples satisfied this condition.

Figure 3.2 illustrates the posterior distribution of the CIFs corresponding to each neuron in the

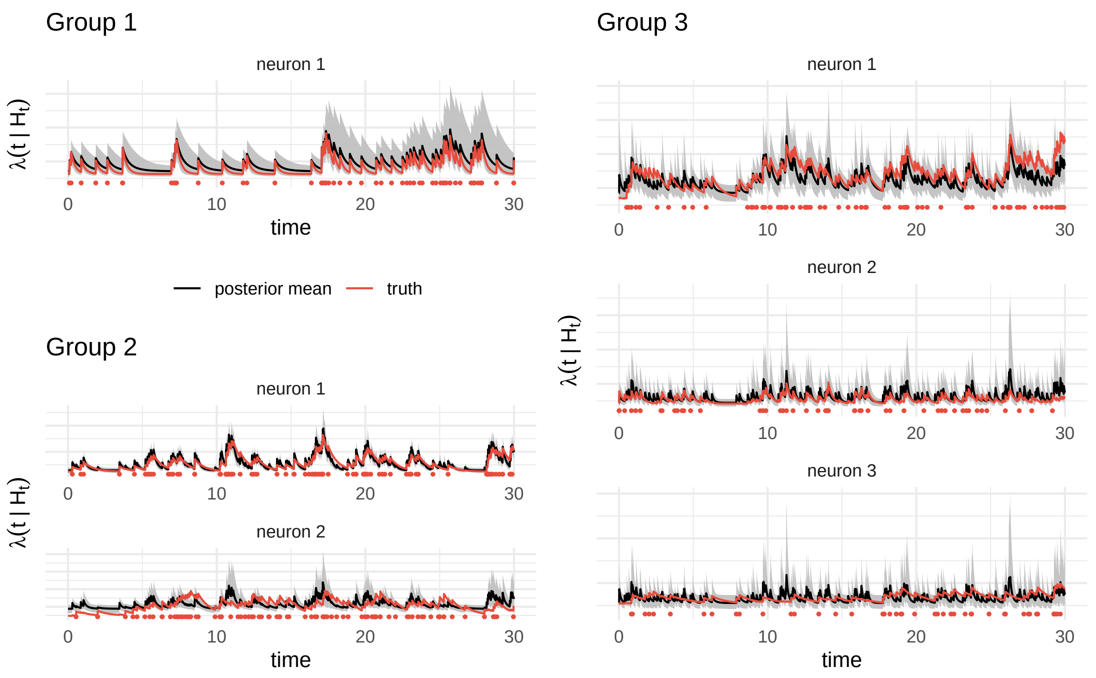


Figure 3.2: Posterior distribution of the CIF for the simulated 3-tetrode 6-neuron network. The gray region in the CIF subplots corresponds to pointwise posterior 95% credible intervals. The events/spikes are shown with red points under each subplot.

simulated data using the cFHP model structure. As we can see, the posterior distributions capture the true CIF for each neuron within its 95% credible intervals.

Next, the cFHP model is tested for non-Hawkes simulated data shown in Figure 3.3. In particular, we simulate a 3-tetrode 6-neuron network over the $(0, 30]$ time interval, with 1, 2, and 3 neurons, using non-homogeneous Poisson processes with different intensity functions. For the single neuron in tetrode 1 (Figure 3.3 Group 1), we use a mixture of two Gaussian distributions with different dispersion parameters as the intensity function. In the second tetrode (Figure 3.3 Group 2), we induce cross-excitation between the pair of neurons by selecting two intensity functions such that the number of spikes are higher in $(10, 20)$ time interval. For the third tetrode (Figure 3.3 Group 3), we induce inhibition between neurons 1 and 2 by having two step functions with opposite behaviors; i.e., a higher number of spikes in a time interval in neuron 1 corresponds to a lower number of spikes in the same time interval in neuron 2 and vice versa, and neuron 3 spikes are generated from a periodic intensity function with low frequency.

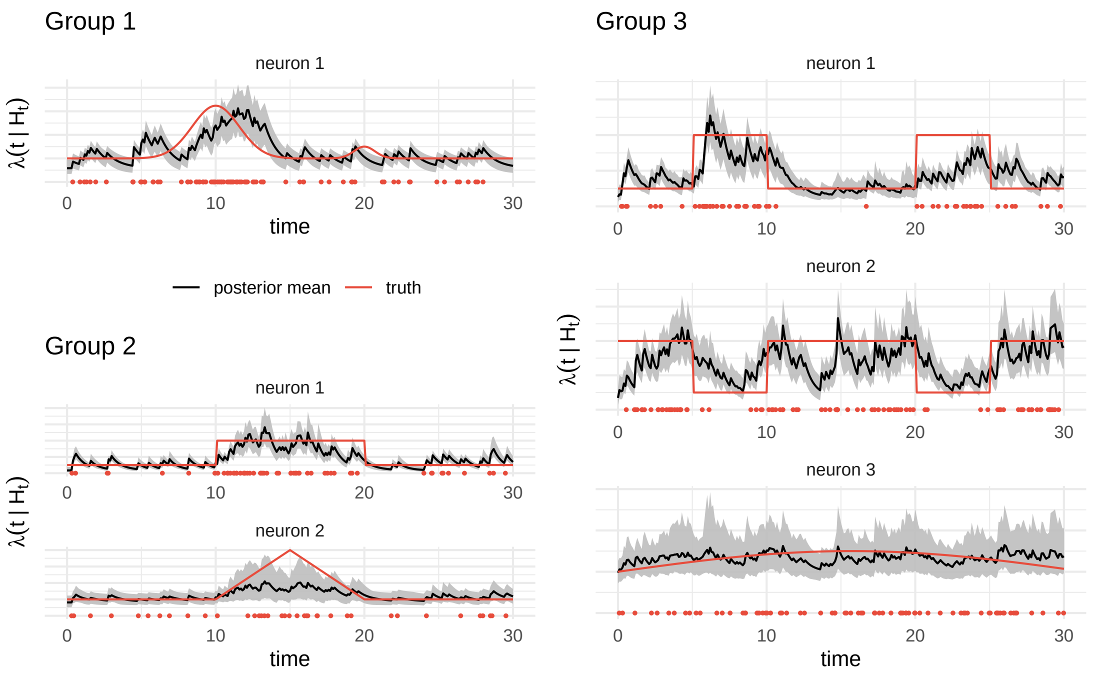


Figure 3.3: Posterior distribution of the CIF for the simulated 3-tetrode 6-neuron network under different data generating mechanisms – a mixture of two Gaussian distributions with different dispersion parameters (Group 1), induced cross-excitation between the pair of neurons with higher number of spikes in $(10, 20)$ time interval (Group 2), induced inhibition between neurons 1 and 2 with opposite behaviors (Group 3).

Finally, to illustrate the computational efficiency gained by our proposed models, we simulate spike train data for different network sizes (10 datasets for each network size), and compute the fitting time of each dataset. For the simulated datasets in the cFHP model structure, we group the neurons into 5 clusters for each network size. Figure 3.4 summarizes these computational times as well as the time-complexity of a "fully" connected multivariate Hawkes process (i.e. excitation-only full interaction among neurons).

As expected, pFHP has substantially lower computational cost compared to the two other alternative parameterizations of FHP; i.e. sFHP and cFHP. In addition, the fully connected model has the highest computational cost for almost all network sizes.

Table 3.1: Decoding results for different model structures. Accuracy (ACC) and F1 scores of predicting odors from CIF predictors using Lasso – a penalized multinomial logistic classification.

	SuperChris		Stella		Buchanan	
Model	ACC	F1	ACC	F1	ACC	F1
Smoothing	0.67	0.635	0.48	0.451	0.57	0.509
Full	0.61	0.463	0.61	0.478	0.75	0.594
Non-interacting	0.67	0.589	0.65	0.582	0.74	0.583
sFHP	0.78	0.741	0.71	0.669	0.78	0.623
cFHP	0.78	0.574	0.69	0.586	0.72	0.576
pFHP	0.79	0.576	0.66	0.576	0.78	0.620

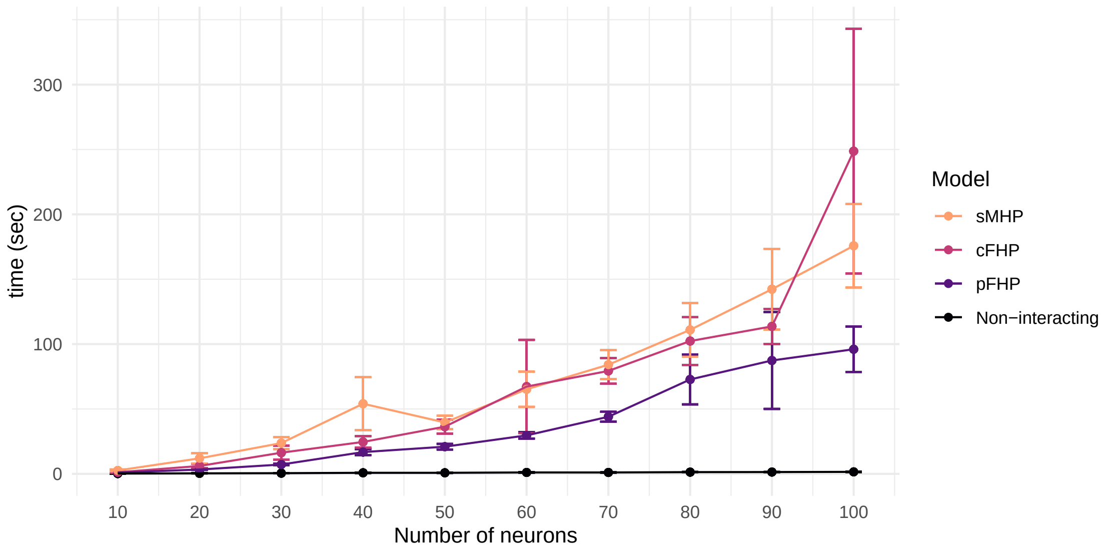


Figure 3.4: Computational (CPU) time for fitting the models on simulated data sets with different number of neurons using the ADVI algorithm in Stan. Corresponding CPU times using MCMC (not included in the plot) are significantly higher, but they follow a similar pattern. The simulations were conducted on a 16 core Intel(R) Xeon(R) Gold 5218 2.30GHz CPU Processor.

3.5 Neural Decoding

We now apply our models to real neural data and evaluate their performance in terms of decoding accuracy. In this experiment, to investigate the temporal organization of memory and behavior, neural activity from the dorsal CA1 region of the hippocampus have been recorded as rats performed a complex sequence memory task [58]. Briefly, the task involves presenting rats with a repeated sequence of non-spatial events (five stimuli: odors A, B, C, D, and E) at a single port, and requiring them to identify each stimulus as "in sequence" (InSeq; i.e. AB$\underline{\text{C}}$...) or "out of sequence" (OutSeq; e.g., AB$\underline{\text{D}}$...). Spiking and local field potential (LFP) activity have been recorded using a chronically-implanted hyperdrive with approximately 20 tetrodes. Data analysis focuses on three rats (named SuperChris, Stella, and Buchanan). On average, about 40 of the recorded neurons (per rat) are active throughout the experiments.

3.5.1 Data

We focus on odor A, B, C, and D trials in which the odor is presented "in sequence" and is correctly identified as such by the animal. For those trials, the spiking activity are extracted during the time window of $(100, 400)$ milliseconds (ms) after the rat's nose enters the port. This window has been chosen to focus on the time period in which hippocampal activity primarily reflects the processing of the incoming odor (previous work shows that earlier and later windows contain information about the previous and upcoming odor; [58]. We exclude odor E trials from the analysis to balance the classes (sampling of odor E was more limited than that of the other odors since errors terminated the sequence). By performing this down sampling of the spike train data, we obtain on average about 200 trials per rat, where each trial contains $M \approx 40$ neurons and lasts for 300 ms. Different models are fitted to each trial using ADVI and the pointwise posterior mean of the CIF per each neuron are calculated. Lastly, the fitted CIF matrices are used as predictors in a Lasso model – a penalized multinomial logistic classification [60] – to distinguish between the odor stimuli. Note that we use a simple classification model since our focus is on extracting information from the neurons through FHP. We use a $60\% - 40\%$ train-test split with 10-fold cross-validation to tune the penalty term in the classification model. The final model is then evaluated on the test dataset to measure the accuracy of the decoding model.

3.5.2 Results

The decoding accuracy (ACC) and the F1 scores corresponding to different model structures introduced in Equations (3.9), (3.11), and (3.12) are summarized in Table 3.1. The performance of our models differ across rats. This is expected due to the natural rat-to-rat variability, different number of neurons in each rat, variability in the location of the implanted electrodes, etc. We compare our proposed models' performances with three baseline methods. The first method is a smoothing procedure – commonly used in neuroscience – in which the activity of each neuron, independent of the rest, is smoothed by convolving a Gaussian kernel with the binary spike train data. In the second method, we used a fully connected multivariate Hawkes model with no inhibition. In the third method, we simply model the multivariate process as a set of "non-interacting" univariate Hawkes processes; i.e. an MHP with a diagonal excitation matrix. The "smoothing" and "non-interacting" settings are unrealistic since they assume the occurrence of spikes in one neuron does not influence the CIF of any other neuron in the network. Furthermore, the "Full" model is also not a realistic model since in a network of neurons not all neurons interact with each other completely.

Overall, we see that in all rats the smoothing approach has the lowest classification accuracy. The "non-interacting" structure performs worse than the sFHP and cFHP structures (introduced in Equations (3.9) and (3.11), respectively) suggesting the need to include interaction effects between neurons into consideration. Additionally, our proposed pFHP (introduced in Equation (3.12)) models provide similar results to the sFHP model, while having a substantially lower computational complexity. Lastly, the Full model with the highest computational cost, does not increase the decoding accuracy, which verifies the sparsity of neuronal connections.

3.6 Discussion and Conclusion

In this paper, we have proposed a flexible, yet scalable multivariate stochastic process model that can capture interactions both in terms of excitation and inhibition in high-dimensional point processes such as neuron spiking activities. Our proposed approach is a novel way of modeling the excitation and inhibition effects among the processes. In contrast to the existing approaches that model inhibition by using negative $\alpha_{m,n}$ parameters and non-linear CIFs, our proposed method keeps the CIF linear by assigning the excitation effects to the $\alpha_{m,n}$ parameters, and the inhibition effects to the $\beta_{m,n}$ parameters. Using this approach, the computational time is significantly reduced due to the reduced number of parameters. Additionally, it allows us to use the recursive framework induced by the exponential decay triggering function.

Future extensions of our model can address some of its shortcoming. For example, the results presented in this paper are based on variational inference, which is an approximate method. In the future, we will explore alternative scalable Markov chain Monte Carlo (MCMC) methods such as Hamiltonian Monte Carlo (HMC) [41] and Variational HMC [63]. In our current models, the base intensity of each neuron is kept constant during the observation interval. In future research, we could use time-varying base intensity functions such as splines or Gaussian processes. Future directions could also focus on increasing our model's flexibility (without substantially increasing its computational overload) by using multiple filters similar to convolutional neural networks (CNN) to capture the influence of an ensemble of neurons in various ways. In addition, future extensions of our model could allow for integrating multiple data modalities (e.g., spike trains and LFP).

Chapter 4

Self-Modulating Multivariate Hawkes Processes

4.1 Introduction

Originally introduced by Hawkes in 1971, *Hawkes processes (HP)* are point processes that capture the behavior of events exhibiting self-exciting dynamics, resulting in the occurrence of events in clusters or bursts in time or space [27]. These processes have found widespread applications in various fields, including social media dynamics [55], analysis of neuronal activity [53], and high frequency financial markets [40]. In the multivariate version of the Hawkes process, multiple sub-processes operate simultaneously, generating events, where the occurrence of an event in a sub-process is influenced by all past events from all sub-processes. The interactions between these different types of events from different sub-processes are modeled using kernel functions, also referred to as *triggering* or *excitation* functions. The original model of the multivariate Hawkes process primarily focuses on *mutually exciting* interactions, whereby the occurrence of an event increases the likelihood of other events happening. This is achieved through the use of non-negative triggering functions. However, these models do not adequately capture physiological phenomena like neuronal activity in the brain due to the presence of inhibitory effects, which are crucial for self-regulation in such cases [20].

The self-exciting Hawkes process has received significant attention, but there is growing interest in modeling inhibition, where certain events reduce the probability of subsequent events. This involves using negative triggering functions and adding a non-linear operator to maintain the positivity of the intensity function. The non-linear Hawkes process, introduced by Brémaud and Massoulié (1996) [7], addresses this by constructing processes using bi-dimensional marked Poisson processes. Other models, such as the neural Hawkes process [37] and the self-limiting Hawkes process [45], have been developed to incorporate inhibition. The neural Hawkes process combines a multivariate Hawkes process with a recurrent neural network, while the self-limiting Hawkes process introduces inhibition as a multiplicative term in the self-exciting intensity function. Apostolopoulou et al. [2] proposed a Bayesian point process model that incorporate a nonlinear component to capture both excitatory and inhibitory relationships in continuous time using Polya-Gamma augmentation and Poisson thinning. In particular, they use a multiplicative probability term to shrink the intensity function of the process in the presence of inhibitory effects.

In this chapter, we introduce a novel model called the *Self-Modulating multivariate Hawkes Process (SMHP)*. This model offers a unique approach to simultaneously and efficiently model both excitation and inhibition processes. Instead of emphasizing the intricate and detailed one-by-one interactions between neurons, our approach focuses on the collective influence of neurons on each other. Our approach reduces the number of parameters to linear instead of quadratic in comparison to the commonly used non-linear multivariate Hawkes process. Numerical studies conducted using both simulated data and real neuronal spiking activity datasets demonstrate that our proposed method achieves comparable performance to the commonly used non-linear Hawkes process model, while significantly reducing computational time. This finding highlights the efficiency and effectiveness of our method in capturing the complex dynamics of neuronal spiking activity.

The remainder of this chapter is organized as follows. First, we will provide a brief overview of the multivariate Hawkes process (MHP), discussing its existence under regularity conditions and the computation of the required log-likelihood. Next, we will introduce our novel model, the Self-Modulating Multivariate Hawkes Process (SMHP), which is designed for computational efficiency. We will present simulation results to demonstrate the performance of the SMHP model. Additionally, we will discuss a rat electrophysiology experiment and evaluate our proposed models based on their neural decoding accuracy. Finally, we will conclude with a discussion of future directions and potential extensions for our method.

4.2 Background

The motivation behind our proposed methodology stems from the analysis of spiking activity recordings obtained from an ensemble of neurons in the hippocampus. The hippocampus, a region in the brain associated with memory formation, is a complex system comprising multiple interconnected neurons [57]. Understanding the dynamics and interactions within this neuronal ensemble is crucial for unraveling the underlying mechanisms of cognitive processes.

Suppose we have an ensemble of M neurons that fire in the time interval $(0, T]$. The *conditional intensity function (CIF)* of neuron m (interpreted as the instantaneous firing rate) in a self-exciting multivariate Hawkes process with the exponential triggering kernel has the following form:

$$\lambda_m\left(t \mid \mathcal{H}_t\right) = \mu_m + \sum_{n=1}^{M} \sum_{t_{n,i}<t} \alpha_{m,n} e^{-\beta_{m,n}(t-t_{n,i})} \tag{4.1}$$

where $\mathcal{H}_t$ is the *history* of the process up to time t containing the event times in all sub-processes, $\mu_m > 0$ is the *baseline intensity* of neuron m, $\alpha_{m,n} \geq 0$ is the *instantaneous increase* in the CIF of the m-th marginal process when an event occurs in the n-th marginal process, and $\beta_{m,n} \geq 0$, is the amount by which this instantaneous increase *decays* over time.

There are three main shortcomings with the formulation of the CIF in Equation 4.1. Firstly, the CIF only increases when an event occurs in any of the marginal processes, resulting in a mutually *exciting* multivariate Hawkes process. This parameterization cannot handle cases where marginal processes have inhibitory effects on each other, as the CIF does not decrease when an event occurs in a different marginal process. Inhibitory effects play a vital role in self-regulation within neuronal systems, and their omission limits the ability of the models to capture and reflect the complex dynamics of neuronal activity [20]. Secondly, the parameters $\alpha_{m,n}$ and $\beta_{m,n}$ define all possible interactions among different marginal processes. This means that the mutual interaction between any pair of neurons m and n can be described using these parameters, resulting in a large number of parameters ($M + M^2 + M^2 = \mathcal{O}(M^2)$) for high-dimensional spike trains, which can make the learning process difficult due to the quadratic number of parameters. Furthermore, this parameterization is very fine-grained, as each event in each of the sub-processes directly influences the CIF of all other sub-processes. When dealing with numerous of neurons firing over a long period of time, we may not be interested in the exact interaction between a specific pair of neurons. Instead, our interest may lie in investigating the behavior of an individual neuron in relation to the collective activity of all other neurons in the ensemble.

The common approach to introducing inhibitory effects into the model is to allow negative values for the $\alpha_{m,n}$ parameters. However, the negative value of the $\alpha_{m,n}$ parameter can make the CIF negative. The CIF is a non-negative quantity, hence, the common approach is to pass the CIF through a positive non-linear function ϕ, ensuring that the CIF remains strictly positive as in equation 4.2:

$$\lambda_m\left(t \mid \mathcal{H}_t\right) = \phi\left(\mu_m + \sum_{n=1}^{M} \sum_{t_{n,i}<t} \alpha_{m,n} e^{-\beta_{m,n}(t-t_{n,i})}\right) \tag{4.2}$$

where $\phi : \mathbb{R} \to \mathbb{R}^+$. Common choices of ϕ include the Rectified Linear Unit function (ReLU), the Softplus function (a smooth approximation to the ReLU function), or the exponential functions [37]. The introduction of the non-linearity ϕ to the conditional intensity function (CIF) in Equation 4.2, combined with the quadratic number of parameters, adds complexity to the learning process.

This complexity arises from the presence of an integral term in the likelihood function of the Hawkes process model, as discussed in Section 4.3.3.

A multivariate Hawkes process with an exponential decay triggering function is considered *stationary* when the spectral radius of its branching matrix is strictly less than 1. Mathematically, this condition is expressed as $\rho(\mathbf{\Gamma}) = \max\{|\boldsymbol{\lambda}_1|, \ldots, |\boldsymbol{\lambda}_M|\} < 1$, where $\mathbf{\Gamma} = \{\alpha_{m,n}/\beta_{m,n}\}$. Here, $\rho(\cdot)$ represents the spectral radius of the branching matrix $\mathbf{\Gamma}$, which contains eigenvalues $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_M$. The stationarity condition holds significant importance as it ensures that the Hawkes process does not explode, meaning that the total number of events within any given time interval remains bounded. This condition becomes particularly relevant when generating simulated data based on an MHP.

4.3 Methods

To address the limitations discussed in the previous section, we propose a novel parameterization of the conditional intensity function (CIF) that incorporates inhibitory effects among marginal processes and reduces the number of parameters. This parameterization enables the analysis of interactions between individual neurons and ensembles of other neurons. The new CIF introduces a parameter Δt that allows us to determine the temporal range over which past events influence the behavior of the system. This parameter controls how far back in history the CIF looks to evaluate the impact of past events on the current behavior. By controlling the temporal extent of past events, we gain flexibility in analyzing the level of "auto-regressiveness" in the process.

4.3.1 A Self-Modulating Multivariate Hawkes Process (SMHP)

The conditional intensity function (CIF) in the SMHP model introduces a separation of the influence of each neuron on itself from the effect of the remaining neurons in the ensemble. This separation is achieved by dividing the CIF into two distinct components: the *ensemble effect* and the *self effect*. The ensemble effect represents the combined influence of all other neurons in the system on the activity of a specific neuron under study. It measures how the collective behavior of the ensemble affects the rate of event occurrences in the neuron being observed. On the other hand, the self effect captures the intrinsic impact of a neuron on its own activity, taking into account the self-excitation dynamics within the neuron. By isolating the self effect, we gain insights into the inherent properties and behaviors of each individual neuron. By decoupling the ensemble effect from the self effect, we can separate the collective impact of the ensemble from the individual contributions of each neuron. The CIF of the SMHP model for neuron m at time t is expressed as:

$$\lambda_m\left(t \mid \mathcal{H}_t\right) = \phi\left(f_m(t)\left[\mu_m + \sum_{t_{m,i} \in (t-\Delta t, t)} \alpha_m e^{-\beta_m(t-t_{m,i})}\right]\right) \tag{4.3}$$

$$f_m(t) = w_0 + \sum_{\substack{n=1 \\ n \neq m}}^{M} w_n g_n(t) \tag{4.4}$$

$$g_n(t) = \sum_i \frac{\mathbf{1}_{t_{n,i} \in (t-\Delta t, t)}}{\Delta t} \tag{4.5}$$

where $\mu_m, \beta_m > 0$ and $\alpha_m, w_0, w_n \in \mathbb{R}$ for $m = 1, \dots, M$. The function $\phi(x) = \log(1 + e^x)$ known as the softplus function ensures that the CIF stays positive. The term $f_m(t)$ represents the ensemble effect of the other neurons on neuron m at time t. It is computed as the sum of a bias term w_0 and a linear combination over the activity of all other neurons in the ensemble (except from neuron m), denoted by $w_n g_n(t)$. The ensemble effect captures the combined influence of the other neurons on neuron m, incorporating both excitatory ($w_n > 0$) and inhibitory ($w_n < 0$) effects. The function $g_n(t)$ represents the aggregate influence of neuron n on neuron m at time t. It takes into account the history of spike events from neuron n within the time window $(t - \Delta t, t)$. The self effect of neuron m on itself is captured by the non-negative term:

$$\mu_m + \sum_{t_{m,i} \in (t-\Delta t, t)} \alpha_m e^{-\beta_m(t-t_{m,i})} \tag{4.6}$$

This term encompasses the self-excitation or self-inhibition dynamics within neuron m, and includes the baseline firing rate μ_m, the excitation/inhibition parameter α_m, and the decay parameter β_m. It quantifies the influence of neuron m on its own activity, taking into account the history of the events in the time interval $(t - \Delta t, t)$.

4.3.2 Properties of the new CIF

Indeed, the CIF presented in Equation 4.3 exhibits interesting properties depending on the configuration of its parameters. In this section, we delve into these properties and demonstrate that under certain parameter settings, the CIF simplifies to well-known point processes. Firstly, let's consider the case when there is no ensemble effect. In this case, $w_n = 0, \forall n$ and the CIF simplifies to:

$$\lambda_m \left(t \mid \mathcal{H}_t\right) = \phi\left(w_0\left[\mu_m + \sum_{t_{m,i}\in[t-\Delta t,t)} \alpha_m e^{-\beta_m(t-t_{m,i})}\right]\right) \tag{4.7}$$

This form of the CIF corresponds to the intensity function of the m-th marginal sub-process in M *independent non-linear Hawkes process*, where the firing rate of neuron m is determined solely by its own past spike events. In this case, $w_0\mu_m$ become the base intensities, and $w_0\alpha_m$ become the self-excitation/inhibition parameters for the neurons. Note that the w_0, μ_m, and α_m parameters stay identifiable since w_0 depends on all of the neurons, while the μ_m, and α_m parameters depend only on individual neurons.

Next, let's consider the case when, in addition to the w_n terms, the terms α_m are also set to zero in Equation (1). In this case, the CIF simplifies to:

$$\lambda_m \left(t \mid \mathcal{H}_t\right) = \phi\left(w_0\mu_m\right) \tag{4.8}$$

which is the intensity function of the m-th marginal sub-process in M *independent homogeneous Poisson processes* each with rate $\phi\left(w_0\mu_m\right)$.

Lastly, let's consider the case when only the α_m paramters are set to zero. In this case, the CIF simplifies to:

$$\lambda_m\left(t \mid \mathcal{H}_t\right)=\phi\left(f_m(t) \mu_m\right) \tag{4.9}$$

which is the intensity function of a mutually exciting/inhibiting multivariate point process where the firing rate of neuron m is only determined by the past activity of other neurons.

The parameter Δt plays a crucial role in controlling the degree of "autoregressiveness" of the point process. As $\Delta t \rightarrow 0$, the process becomes less dependent on the past history and approaches a state where $\lambda_m(t)$ converges to $\phi(w_0 \mu_m)$. In this case, the process becomes largely determined by the bias term w_0 and the baseline firing rate μ_m, with minimal influence from the past events, resembling a homogeneous Poisson process. On the other hand, as $\Delta t \rightarrow \infty$, the process takes into account the entire history of events in the past. The CIF becomes more sensitive to the temporal patterns and dependencies among events, resulting in a richer and more complex model of the point process. Therefore, the choice of Δt allows us to control the level of memory and temporal dynamics captured by the process, ranging from a simple and “memoryless” behavior to a more intricate and history-dependent pattern.

4.3.3 Likelihood Function

To motivate the idea behind the computation of the likelihood, consider an experiment where we are monitoring the spiking activity of an ensemble of M neurons over the time interval $(0, T]$. In addition, suppose that we have S independent repetitions of the experiment, resulting in S sequences of spiking activity from the M neurons. We discretize the time interval $(0, T]$ into millisecond bins, ensuring that at most one spike occurs within each bin. To represent the spiking activity, we use the binary variable $\mathcal{Y}_{t,m,s} \in \{0,1\}$, where $\mathcal{Y}_{t,m,s} = 1$ indicates the presence of a spike at time bin t in neuron m during sequence s. Hence, the spiking activity dataset is a 3-dimensional array $\mathcal{Y}$ with dimensions $1000T \times M \times S$, containing the spike information for S sequences of M-variate point processes, each lasting for $J = 1000T$ milliseconds. The contribution of sequence s to the log-likelihood of the model is as follows:

$$\ell^{(s)}\left(\boldsymbol{\theta} \mid \mathcal{Y}_{:,:,s}\right) = \sum_{m=1}^{M} \left(\sum_{j=1}^{J} \mathcal{Y}_{j,m,s} \log \lambda_m(t_j \mid \boldsymbol{\theta}, \mathcal{H}_t) - \int_0^T \lambda_m(s \mid \boldsymbol{\theta}, \mathcal{H}_t)\, ds \right) \tag{4.10}$$

where $\mathcal{Y}_{:,:,s}$ is the spiking activity of all neurons in sequence s, $\boldsymbol{\theta}$ contains all the parameters of the model, and $t_j \in \{0.001, 0.002, \ldots, T - 0.002, T - 0.001, T\}$. Consequently, the log-likelihood for the entire data set is going to be equal to:

$$\ell\left(\boldsymbol{\theta} \mid \mathcal{Y}\right) = \sum_{s=1}^{S} \ell^{(s)}\left(\boldsymbol{\theta} \mid \mathcal{Y}_{:,:,s}\right) \tag{4.11}$$

The integral in Equation 4.10 is known as the *compensator* of the point process, and is denoted by $\Lambda_m(T)$. However, due to the presence of the non-linear function ϕ (in Equation 4.3), it is not feasible to obtain a closed-form solution for the compensator. As a result, we resort to numerical approximation methods to evaluate this integral. In particular, we employ the composite Simpson's $1/3$ rule [10] as a numerical approximation technique for approximating the integral in the compensator. To apply the Simpson's rule, we divide the interval $(0, T]$ into K sub-intervals of length T/K such that the end-points of the k-th interval are $\left(\frac{(k-1)T}{K}, \frac{kT}{K}\right)$ for $k = 1, \ldots, K$. Then we approximate the integral over the k-th sub-interval with second degree Lagrange interpolating polynomials with equally-spaced nodes:

$$\begin{aligned}
&\int_{\frac{(k-1)T}{K}}^{\frac{kT}{K}} \lambda_m(s \mid \boldsymbol{\theta}, \mathcal{H}_t)\, ds \\
&\approx \frac{T}{6K}\left[\lambda_m\left(\frac{(k-1)T}{K}\middle|\boldsymbol{\theta}, \mathcal{H}_t\right) + 4\lambda_m\left(\frac{\frac{(k-1)T}{K} + \frac{kT}{K}}{2}\middle|\boldsymbol{\theta}, \mathcal{H}_t\right) + \lambda_m\left(\frac{kT}{K}\middle|\boldsymbol{\theta}, \mathcal{H}_t\right)\right] \\
&= \frac{T}{6K}\left[\lambda_m\left(\frac{(k-1)T}{K}\middle|\boldsymbol{\theta}, \mathcal{H}_t\right) + 4\lambda_m\left(\frac{T}{2K}\middle|\boldsymbol{\theta}, \mathcal{H}_t\right) + \lambda_m\left(\frac{kT}{K}\middle|\boldsymbol{\theta}, \mathcal{H}_t\right)\right]
\end{aligned} \tag{4.12}$$

which results in an approximation to the compensator by:

$$\begin{aligned}&\int_0^T \lambda_m(s\Big|\boldsymbol{\theta}, \mathcal{H}_t)\, ds \\ &\approx \sum_{k=1}^{K} \frac{T}{6K}\left[\lambda_m\left(\frac{(k-1)T}{K}\Big|\boldsymbol{\theta}, \mathcal{H}_t\right) + 4\lambda_m\left(\frac{T}{2K}\Big|\boldsymbol{\theta}, \mathcal{H}_t\right) + \lambda_m\left(\frac{kT}{K}\Big|\boldsymbol{\theta}, \mathcal{H}_t\right)\right]\end{aligned} \tag{4.13}$$

4.4 Results

In this section, we demonstrate the application of our Self-Modulating Multivariate Hawkes Process (SMHP) model through simulation studies and analysis of real-world data. This allows us to comprehensively assess the performance and effectiveness of our model in capturing the intricate dynamics of neural activity. In the simulation studies, we generate neuronal spiking activity to evaluate the performance of our model. To assess the goodness-of-fit, we utilize a Kolmogorov-Smirnov (KS) test combined with the well-known Time-rescaling theorem [18], which enables us to examine how well our model captures the underlying patterns of neuronal firing. Additionally, we measure the predictive performance of our model and compare it to other existing Hawkes process models, providing insights into its superiority and potential advantages. Moving to the real-data application, we apply the SMHP model to analyze recorded spike data from the CA1 region of the hippocampus in rats. We evaluate the performance of the model using a left-out dataset with known ground truth to assess the ability to accurately represent the underlying dynamics.

4.4.1 Simulation Study

In the first simulation study, our goal is to demonstrate the fundamental characteristics of the model described in Equations 4.3 to 4.5. First, we simulate the spiking activity of a network of three neurons over a time interval of $(0, T] = (0, 30]$ according to the CIF introduced in Equations 4.3 to 4.5 using Ogata's Modified Thinning Algorithm [42]. Specifically, we impose the following structure on the excitation/inhibition effects between neurons. We consider that neurons exhibit a self-exciting behavior, while have different behavior in relation to other neurons. Neuron 1 exerts an inhibitory influence on the spiking activity of neurons 2 and 3, meaning that whenever neuron 1 generats a spike, it actively suppresses the occurrence of spikes in neurons 2 and 3. On the other hand, neuron 2 exhibits an excitatory effect on both neuron 1 and neuron 3. That is, when neuron 2 produces a spike, it enhances the likelihood of spikes occurring in neurons 1 and 3. Furthermore, neuron 3 also exhibits an excitatory effect on neurons 1 and 2, although to a lesser extent. Its spikes have a stimulating impact on the spiking activity of neurons 1 and 2, but the magnitude of this effect is comparatively smaller compared to that of neuron 2. The model parameters are drawn from the following distributions:

$$\mu_m \overset{\text{iid}}{\sim} \text{uniform}(0, 2) \tag{4.14}$$

$$\alpha_m \overset{\text{iid}}{\sim} \text{Half-Normal}(1) \tag{4.15}$$

$$\beta_m \overset{\text{iid}}{\sim} \text{Normal}(2 \max_m\{\alpha_m\}, 0.2) \tag{4.16}$$

$$w_0 \overset{\text{iid}}{\sim} \text{Normal}(0, 0.5) \tag{4.17}$$

$$w_m \overset{\text{iid}}{\sim} \text{Normal}(0, 1) \tag{4.18}$$

for $m = 1, \ldots, M$. The mean of the Normal prior distribution for the β_m parameters is carefully selected to maintain the stationarity of the model and prevent an excessive increase in the number of spikes. To achieve this, we set the mean of the prior distribution to be $2\max_m\{\alpha_m\}$, where $\max_m\{\alpha_m\}$ represents the maximum value among all the α_m parameters. This choice helps to stabilize the model and ensure that the spiking activity remains bounded (see Section 4.2 for the details on the stationarity). Figure 4.1 shows the simulated spikes from the network described above. It is worth noting that the inhibitory behavior of neuron 1 towards other neurons is clearly observed. As a result, the CIFs of neurons 2 and 3 exhibit a decreasing trend from the second half of the time interval until the end. This decrease is due to the increasing number of spikes generated by neuron 1, which inhibits the firing activity of neurons 2 and 3 during that period.

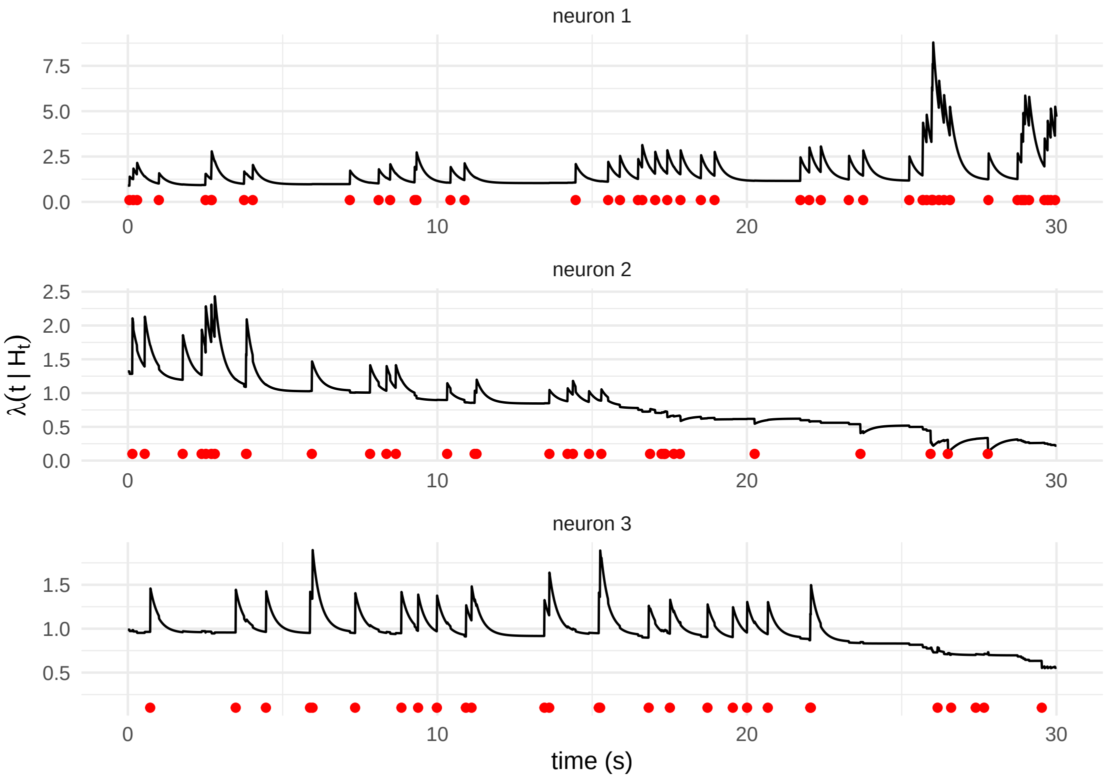


Figure 4.1: Simulated spikes (represented by red dots) along with the conditional intensity functions (represented by black lines) corresponding to a network of 3 interconnected neurons over the time interval $(0, 30]$ seconds. Each of the neurons have a self-excitatory behavior. However, they influence each other differently. Neuron 1, the top sub-plot, has an inhibitory effect on the activity of the other neurons. Whenever neuron 1 generates a spike, it suppresses the likelihood of spikes occurring in neurons 2 and 3. As a result, we see a decrease in the CIFs of neurons 2 and 3 from the second half of the time interval onwards. On the other hand, neuron 2 and neuron 3, represented by the middle and the bottom sub-plot respectively, have an excitatory effect on the activity of other neurons. When either neuron 2 or neuron 3 produces a spike, it enhances the probability of spikes occurring in the other neurons.

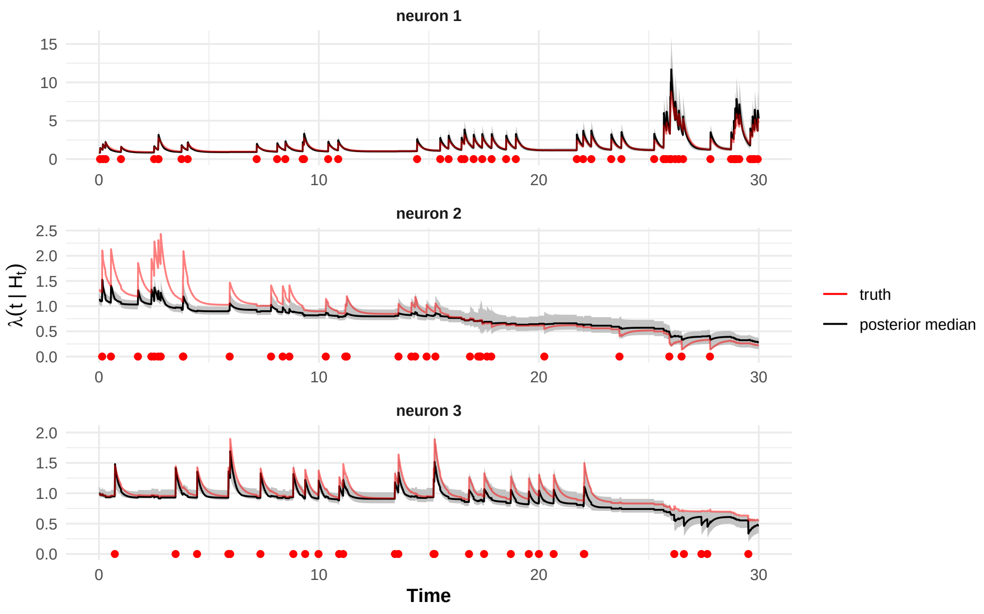


Figure 4.2: Point-wise posterior median of the CIF (depicted by black lines) for the simulated 3-neuron network shown in Figure 4.1. The gray region in the CIF subplots corresponds to point-wise posterior 95% credible intervals. Spikes are shown with red points under each subplot and the true intensity function is depicted with the red curves.

To fit the proposed model, we use Automatic Differentiation Variational Inference (ADVI) [32] in Stan [13] with weakly informative priors. Figure 4.2 illustrates the posterior distribution of the CIFs corresponding to each neuron in the simulated data. As we can see, the posterior distributions capture the true CIF for each neuron within its 95% credible intervals. Assessing the adequacy of a point process model in fitting observed spike data is a crucial practical consideration. The Time-Rescaling Theorem, a well-known concept in point process analysis [18], is an important tool for conducting this evaluation. According to the theorem, if we have a realization $0 \leq t_1 < t_2 < \cdots < t_{N(T)} \leq T$ from a point process $N(.)$ with a non-negative intensity function $\lambda(t)$, then the sequence $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_{N(T)})\}$ forms a Poisson process with a unit rate, where $\Lambda(t_k) = \int_0^{t_k} \lambda(u)\, du < \infty$ is referred to as the *compensator* evaluated at time t_k. The compensator plays a key role in assessing the goodness-of-fit of the encoding model. To evaluate the goodness-of-fit, the non-parametric Kolmogorov-Smirnov (KS) test can be performed by comparing the values $\{\Lambda_m(t_1), \Lambda_m(t_2), \ldots, \Lambda_m(t_{N(T)})\}$ of neuron m to those of a unit-rate Poisson process. Since the inter-arrival times of events in a unit-rate Poisson process follow an exponential distribution

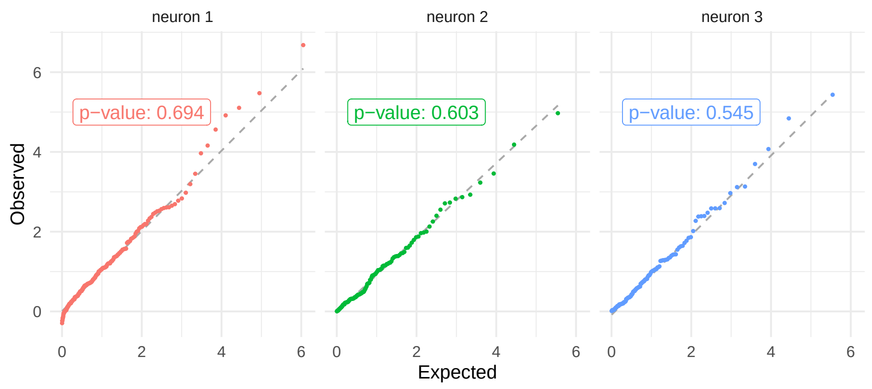
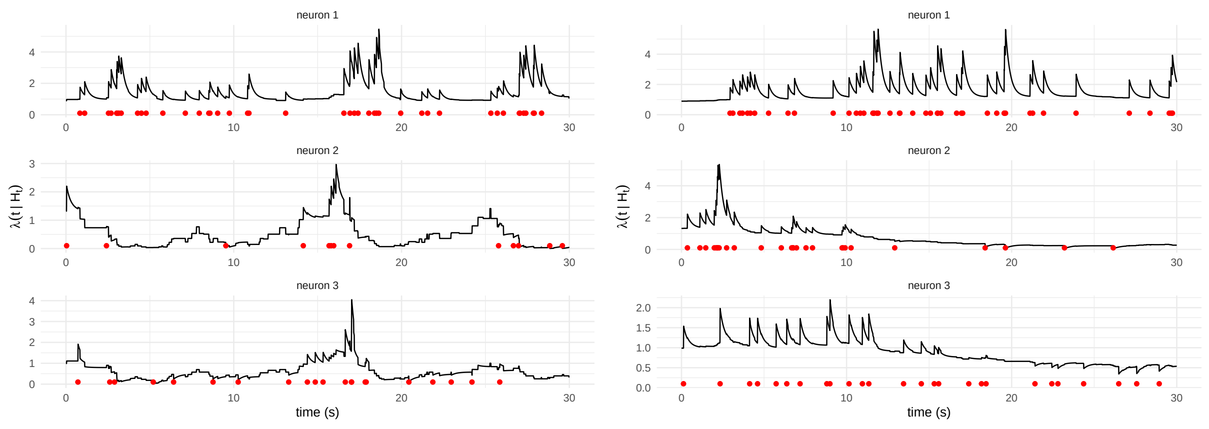


Figure 4.3: The goodness-of-fit quantile-quantile plots illustrate the comparison between the inter-arrival times of the time-rescaled spikes from the simulated spike train data and the theoretical quantiles from the exponential distribution with a rate parameter of 1. This plot provides insights into the adequacy of the encoding model. The p-values resulting from the Kolmogorov-Smirnov (KS) tests are displayed on each sub-plot, serving as an indication of the goodness of fit. Larger p-values suggest a reasonable fit (H_0 : time-rescaled spikes follow an exponential(1) distribution), implying that the encoding model accurately captures the underlying distribution of the inter-arrival times in the spike train data.

with a rate of 1, we can compare the values $\{\Lambda_m(t_k+1) - \Lambda_m(t_k)\}_{k=1,\ldots,N(T)}$ to an exponential distribution with a rate of 1. The quantile-quantile plots of the time-rescaled spikes, along with the corresponding p-values from the KS tests, are presented in Figure 4.3.

The parameter Δt plays a crucial role in Equation 4.3 as it determines the extent to which the past spiking activity influences the computation of the intensity function. In other words, Δt controls the temporal window over which the history of spiking events is taken into account. When Δt is small, the intensity function is primarily influenced by recent spiking events, giving more weight to the immediate past. This implies that the process is more reactive to recent activity and tends to exhibit shorter memory or dependence on previous events. On the other hand, when Δt is large, the intensity function considers a broader range of past spiking events, encompassing a longer temporal window. This leads to a more persistent and memory-driven behavior, where the process takes into account a more extensive history of events. Figure 4.4 illustrates the impact of the Δt parameter on simulated spike data and their corresponding intensity functions using the same



(a) $\Delta t = 3$ (b) $\Delta t = 24$

Figure 4.4: Effect of the Δt parameter on the "autoregressiveness" of the conditional intensity functions using the same model parameters from Equation 4.14 but different Δt values. When using a small Δt, the spikes exhibit a short-term dependence on recent events, with a rapid rise and fall in response to immediate past activity. On the other hand, a larger Δt leads to a more memory-driven behavior, where the spiking activity reflects a longer temporal window of past events. This results in a smoother intensity function and a slower rise and decay pattern of the spikes.

set of parameters seen in Equation 4.14. In the first set of spikes where $\Delta t = 3$, the simulated spiking activity exhibits a more reactive and short-term dependence on past events. The spikes are primarily influenced by recent activity, and the intensity function shows a rapid rise and fall in response to immediate past events. In contrast, the second set of spikes, which corresponds to a larger value of $\Delta t = 24$, displays a more persistent and memory-driven behavior. Here, the simulated spiking activity takes into account a longer temporal window of past events, resulting in a smoother and more sustained intensity function. The spikes reflect a broader influence of previous events, showing a slower rise and decay pattern. By comparing these two sets of simulated spikes, it becomes evident that the choice of Δt has a significant impact on the temporal dynamics and memory properties of the spiking process.

In the second simulation study, we investigate the utility of the fitted CIFs under different parameter configurations to understand how they can assist in neural decoding and classification tasks. We analyze the performance of the SMHP model by utilizing the fitted CIFs as predictors to classification models, aiming to evaluate their effectiveness in accurately decoding and classifying neural activity patterns. To this end, we simulate the spiking activity of a network of 10 neurons over a time interval of 0 to 5 seconds. The entire interval is divided into sub-intervals of one second each, corresponding to different stimuli. We generate 10 different sets of spiking activity datasets for the neurons within this time frame. In other words, we have generated 10 realizations of spiking activity for a 10-variate point process, where each realization corresponds to the spiking activity of the 10 neurons in the network over the specified time interval. The goal is to mimic the behavior observed in real neural networks, where interactions between neurons can be inhibitory or excitatory, and some neurons may not exhibit any specific patterns of influence exposed to different stimuli during an experiment. Within this network, we incorporate both inhibitory and excitatory connections. Some neurons act as inhibitors, exerting an inhibitory effect on the activity of other neurons. On the other hand, we have neurons that act as excitatory agents, stimulating the activity of other neurons. In addition to the inhibitory and excitatory interactions, there are neurons in the network that do not possess any specific patterns of influence. These neurons are not actively involved in inhibiting or exciting other neurons. The resulting spike train data are shown in Figure 4.5. From this figure, we can observe specific firing patterns of different neurons in response to different stimuli. Neuron 1, for example, exhibits firing activity exclusively during the presentation of stimulus A. During this stimulus, it inhibits the firing of neurons $2, 3, 4$, and 5, effectively suppressing their activity. On the other hand, neuron 2 shows activity specifically during the presentation of stimulus B. During this stimulus, it inhibits the firing of neurons $1, 3, 4, 5$, and 8, preventing them from generating spikes. Neuron 3, in turn, becomes active during stimulus C. It exerts inhibitory control over neurons $1, 2, 4, 5, 6, 7$, and 8, suppressing their firing during the presence of this particular stimulus. The influence of the remaining neurons can be interpreted similarly, with each neuron exhibiting specific firing patterns during certain stimuli while inhibiting the firing of other neurons.

After generating the simulated spike train data, we proceed to evaluate the predictive capabilities of the SMHP model by comparing it to three other existing models. Specifically, we compare it against a self-excitation only multivariate Hawkes process (MHP) model, the Pooled Flocking Hawkes Process (pFHP) model presented in Chapter 3, and a non-linear multivariate Hawkes pro-

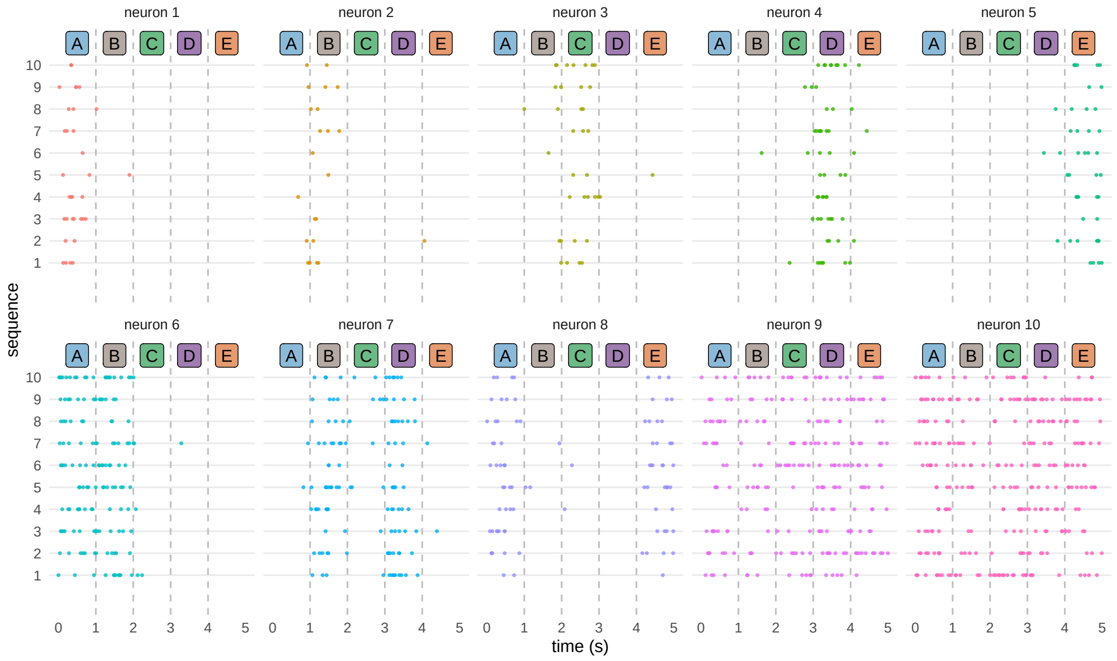


Figure 4.5: Simulated spiking activity of a network of 10 neurons. The x-axis ranges from 0 to 5 seconds, representing the duration of five stimuli, with each stimulus lasting for one second. The y-axis consists of 10 rows, each corresponding to one of the 10 sets of simulated spike train data. Each subplot in the figure represents the spiking activity of an individual neuron. The network includes inhibitory and excitatory connections, with some neurons acting as inhibitors and others as excitatory agents. The figure illustrates distinct firing patterns of different neurons in response to different stimuli. For instance, Neuron 1 only fires during the presentation of stimulus A and inhibits the firing of neurons $2, 3, 4$, and 5 during this stimulus. Similarly, Neuron 2 is active during odor B and inhibits the firing of neurons $1, 3, 4, 5$, and 8. Neuron 3, during stimulus C, inhibits the firing of neurons $1, 2, 4, 5, 6, 7$, and 8. The influence of the remaining neurons follows a similar pattern, with each neuron exhibiting specific firing patterns during certain stimuli while inhibiting the firing of other neurons.

cess model (NL-MHP) with inhibition. By conducting this comparative analysis, we aim to assess the effectiveness and superiority of the SMHP model in capturing the complex dynamics of the neuronal activity. To assess the impact of inhibitory effects in the dataset, we specifically select the self-excitation only MHP model as one of the comparison models. This choice allows us to observe how this model performs when faced with such effects. However, our main focus is on comparing the performance of the SMHP model with that of the NL-MHP model. The NL-MHP model includes all interactions among the neurons and serves as the primary benchmark for evaluating the SMHP model. Our objective is to demonstrate that the SMHP model achieves similar performance to the NL-MHP model while significantly reducing computational time.

We start by separating the spikes in each sequence based on their stimulus type, resulting in 50 spike trains representing 10 different sequences of A, B, C, D, and E stimuli. Next, we fit the four models (MHP, pFHP, NL-MHP, and SMHP) to each of these trials, estimating the point-wise posterior medians of the CIFs for each neuron. This yields 50 matrices of size 1000×10 (corresponding to each of the 10 simulated spike trains and stimulus types). Afterwards, we utilize these posterior median CIFs as predictors and their corresponding stimuli as the response variable to train various classification models. Specifically, we divide the data into a $60\% - 40\%$ train-test split and employ 10-fold cross-validation to optimize the penalty term in the penalized multinomial logistic regression model [60] as our chosen classification model. Table 4.1 and Figure 4.6 present numerical and graphical summaries of the results of this procedure, illustrating the performance of these models using three different metrics: accuracy, F1 score, and ROC-AUC. The NL-MHP model achieves the best classification metrics, indicated by italic font. However, the SMHP model exhibits comparable classification performance to the NL-MHP, highlighted in bold font. It is expected that the NL-MHP model performs well, but this superiority comes at a considerable computational cost. The fitting time (CPU time) of the model fitting procedures is depicted in Figure 4.7. As anticipated, the pFHP model exhibits the shortest fitting time, which is explained in detail in Chapter 3. On the other hand, the NL-MHP model requires the longest fitting time due to its non-linearity and consideration of all possible interactions among neurons. The reduced computational time of

Metric	Model	Mean	95% CI
Accuracy	MHP	0.775	(0.690, 0.860)
	pFHP	0.775	(0.747, 0.803)
	NL-MHP	*0.813*	(0.666, 0.959)
	SMHP	**0.800**	(0.673, 0.927)
F1 score	MHP	0.768	(0.683, 0.852)
	pFHP	0.775	(0.743, 0.806)
	NL-MHP	*0.801*	(0.649, 0.954)
	SMHP	**0.798**	(0.672, 0.924)
ROC-AUC	MHP	0.959	(0.927, 0.992)
	pFHP	0.963	(0.926, 1.001)
	NL-MHP	*0.980*	(0.954, 1.005)
	SMHP	**0.973**	(0.947, 0.998)

Table 4.1: Comparison of the performance of several models: the Self-Modulating Hawkes Process (SMHP) model in Equation 4.3, the vanilla Multivariate Hawkes Process (MHP), the Pooled Flocking Hawkes Process (pFHP) from Chapter 3 Section 3.3.3, and the Non-Linear multivariate Hawkes Process (NL-MHP) in Equation 4.2. Among these models, the best classification metrics are achieved by the NL-MHP model, which is indicated by italic font. The mean classification performance metric of the SMHP model is highlighted in bold font. The column corresponding to the confidence intervals is achieved by repeating the entire data generating and classification process multiple times.

the SMHP model confirms its ability to achieve strong classification performance while utilizing fewer computational resources.

Lastly, we compare the computational time of the SMHP model with the other models using varying sizes of neuronal networks. We consider five different network sizes: $M = 10, 20, 30, 40, 50$. For each network size, we simulate multiple datasets for each stimulus and fit each of the aforementioned models, recording their respective fitting times. The results are presented in Table 4.2 and illustrated in Figure 4.8. Figure 4.8 demonstrates that the fitting time of the NL-MHP model is more than two orders of magnitude longer than that of the SMHP model for small network sizes. Additionally, for larger network sizes, the NL-MHP model still requires approximately an order of magnitude more time compared to the SMHP model.

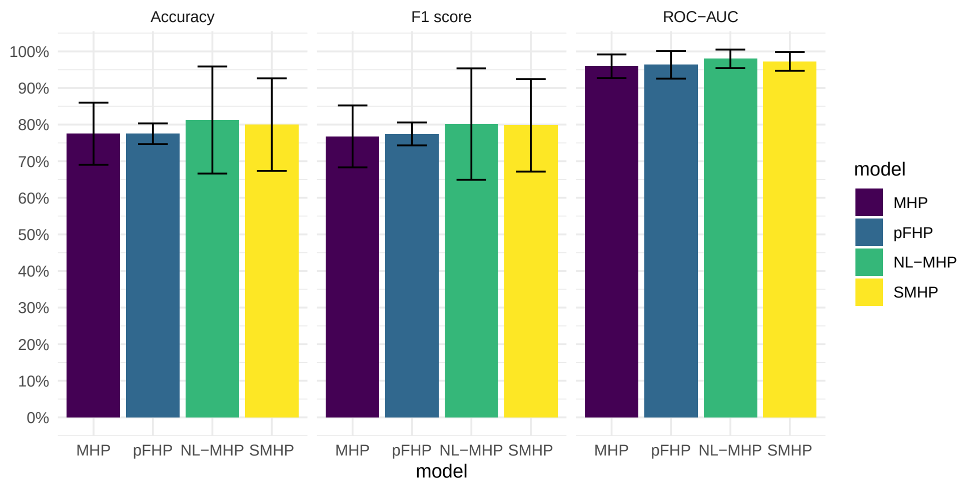


Figure 4.6: The performance of several models, including the Self-Modulating Hawkes Process (SMHP) model, the Multivariate Hawkes Process (MHP), the Pooled Flocking Hawkes Process (pFHP) from Chapter 3 Section 3.3.3, and the Non-Linear multivariate Hawkes Process (NL-MHP), was compared. Among these models, the NL-MHP and SMHP achieve similar performace metrics. The error bands show the uncertainty of these performance metrics.

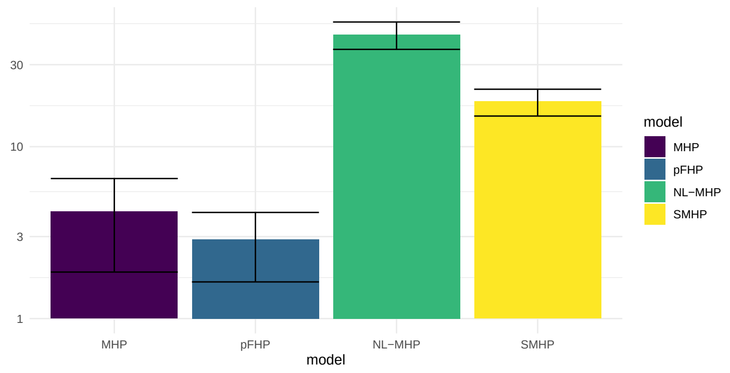


Figure 4.7: Computational time (CPU time) of the model fitting procedures. The pFHP model has the shortest fitting time, while the NL-MHP model has the longest fitting time due to its non-linearity and consideration of all possible interactions among neurons. The SMHP model achieves good classification performance (as seen in Table 4.1)with reduced computational time, indicating its efficiency in utilizing computational resources.

	Number of neurons				
model	10	20	30	40	50
pFHP	(0.114, 0.119)	(0.114, 0.118)	(0.113, 0.222)	(0.113, 0.317)	(0.137, 0.52)
MHP	(0.113, 0.214)	(0.115, 0.421)	(0.24, 1.72)	(0.439, 1.88)	(0.943, 6.7)
NL-MHP	(30.7, 62.6)	(104, 247)	(164, 481)	(247, 790)	(89.3, 1390)
SMHP	(0.125, 0.433)	(0.566, 3.39)	(1.04, 19.1)	(4.04, 94.7)	(2.92, 204)

Table 4.2: Computational (CPU) time for fitting the models on simulated data sets with different number of neurons using MAP estimates. The intervals in each cell correspond to the 95% confidence intervals of the fitting times in seconds. Corresponding CPU times using MCMC (not included in the plot) are significantly higher, but they follow a similar pattern. The NL-MHP model exhibits a fitting time that is more than two orders of magnitude longer than the SMHP model for small network sizes. Furthermore, even for larger network sizes, the NL-MHP model still requires approximately an order of magnitude more time compared to the SMHP model.

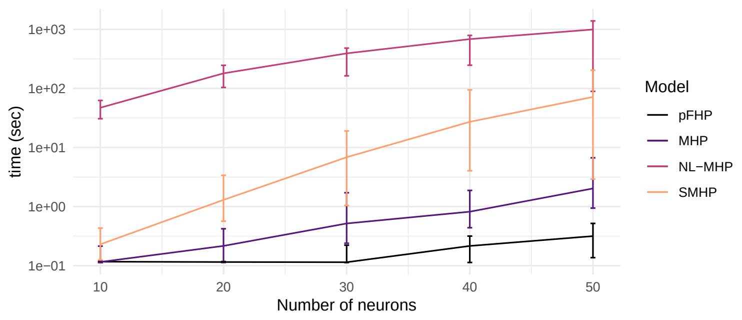


Figure 4.8: The CPU time required for fitting the models to simulated datasets with varying numbers of neurons using MAP estimates is presented. It is worth noting that the corresponding CPU times using MCMC (not depicted in the plot) are significantly higher, although they exhibit a similar pattern. The NL-MHP model demonstrates significantly longer fitting times than the SMHP model, exceeding two orders of magnitude for small network sizes. This difference persists for larger network sizes, with the NL-MHP model requiring approximately an order of magnitude more time than the SMHP model.

4.4.2 Real Data

We now apply our model to real neural data and evaluate their performance in terms of decoding accuracy. The neural activity recordings were obtained from the dorsal CA1 region of the hippocampus during a complex sequence memory task conducted by rats (Shahbaba et al., 2022 [58]). In this task, rats were presented with a repeated sequence of non-spatial events involving odors A, B, C, D, and E. The rats had to classify each stimulus as either "in sequence" (InSeq) or "out of sequence" (OutSeq). The data analysis focuses on a specific rat named SuperChris, and the recordings were obtained using a chronically-implanted hyperdrive with approximately 20 tetrodes. The analysis specifically considers trials where odors A, B, C, and D were presented in sequence and correctly identified by the rats. The spiking activity is extracted within a time window of 100 to 400 milliseconds (ms) after the rat's nose enters the port. This time window captures the period when hippocampal activity primarily reflects the processing of the incoming odor [58]. To balance the classes, odor E trials are excluded from the analysis. By downsampling the spike train data, we obtain 164 trials (55 A's, 41 B's, 37 C's and 31 D's), each containing around 40 neurons and lasting for 300 ms. We further select the neurons that have a firing rate greater than 5×10^{-3} throughout the entire experiment, resulting in 15 neurons.

For each trial, different models are fitted using ADVI, and the pointwise posterior mean of the conditional intensity function (CIF) is calculated for each neuron. These fitted CIF matrices serve as predictors in a multinomial LASSO model [60], which is a penalized multinomial logistic regression, to classify the odor stimuli. Since the focus is on extracting information from the neurons using the FHP models, a simple classification model is employed. A $60\% - 40\%$ train-test split is used, and the penalty term in the classification model is tuned using 10-fold cross-validation. The final model is then evaluated on the test dataset to measure the decoding accuracy.

Table 4.3 presents the performance metrics of different models in decoding odor stimuli based on neural activity recorded from the hippocampus during a complex sequence memory task per-

Model	Accuracy	ROC-AUC	F1 Score
MHP	0.702	0.884	0.618
pFHP	0.687	0.853	0.596
NL-MHP	0.746	0.896	0.722
SMHP	0.702	0.833	0.645

Table 4.3: Performance metrics of different models in decoding odor stimuli based on neural activity recorded from the dorsal CA1 region of the hippocampus during a complex sequence memory task performed by SuperChris. The models evaluated include the Multivariate Hawkes Process (MHP), the Pooled Flocking Hawkes Process (pFHP), the Non-Linear multivariate Hawkes Process (NL-MHP), and the Self-Modulating Hawkes Process (SMHP). Accuracy, area under the receiver operating characteristic curve (AUC), and F1 score are used as performance metrics.

formed by the rat named SuperChris. The models evaluated include the Multivariate Hawkes Process (MHP), the Pooled Flocking Hawkes Process (pFHP), the Non-Linear multivariate Hawkes Process (NL-MHP), and the Self-Modulating Hawkes Process (SMHP). The metrics assessed are accuracy, area under the receiver operating characteristic curve (AUC), and F1 score. Among the models evaluated, as expected, the NL-MHP model achieved the highest accuracy of 0.746, closely followed by the SMHP model with an accuracy of 0.702. It is worth noting that the SMHP model achieves this high performance in a shorter amount of time compared to the NL-MHP model.

4.5 Discussion and Conclusion

In conclusion, we have examined the performance of various models, including our proposed Self-Modulating Hawkes Process (SMHP), in decoding neural activity and classifying odor stimuli. Our findings reveal that the SMHP model achieves comparable performance to the NL-MHP model while demonstrating better computational efficiency. This suggests that the SMHP model holds promise for decoding neural information and has potential for practical applications. However, it is important to acknowledge the limitations of the SMHP model. One limitation is its assumption of a self-modulating mechanism, which may not accurately capture complex interactions between neurons in certain scenarios. Additionally, the generalizability of the SMHP model beyond the specific context of decoding odor stimuli needs to be explored. Furthermore, the reliance on variational inference, an approximate method, for obtaining results warrants consideration. Alternative scalable Markov chain Monte Carlo (MCMC) methods, such as Hamiltonian Monte Carlo (HMC) and Variational HMC, can be explored to improve the inference process. The performance of the SMHP model may also vary when applied to different neural datasets or experimental paradigms, necessitating thorough assessment and validation across various domains. Moreover, the interpretability of the model parameters and their ability to uncover meaningful insights about neural dynamics can result in further insights to the intricate dynamics happening in the brain. Despite these limitations, the SMHP model represents a novel approach in the field of neural decoding and paves the way for future research in understanding the temporal organization of neural activity and its relationship to behavior and cognition.

Chapter 5

Bayesian Non-Parametric Point Process Modeling for Neural Decoding

5.1 Introduction

The hippocampus is a brain structure that plays a crucial role in the formation, retrieval, and temporal organization of memories. Within the hippocampus, there are specialized neurons known as *time cells* that fire at specific moments within a cognitive task or experience [19]. These time cells are thought to provide a temporal framework for the encoding and retrieval of memories. In particular, neuroscientists have been interested in understanding how these cells encode information about time and how this information is used by the brain [36]. By analyzing the firing rates of different time cells in a given recorded brain activity, it is possible to estimate or decode the time of the experimental conditions or stimuli that was present during an event or task. In addition to time cells, there are also neurons in the hippocampus that are selective for specific spatial locations, known as *place cells* [9]. Place cells play a key role in spatial navigation and have been extensively studied in rodents and other animals. The firing of place cells is thought to provide a neural representation of the animal's position in its environment, allowing it to navigate effectively. Both time cells and place cells have been the subject of intensive research in recent years, as they are thought to play important roles in the formation and retrieval of memories. By studying the spiking activity of these cells, neuroscientists hope to gain a better understanding of how memories are stored and retrieved in the brain.

One of the primary functions at which the hippocampus is responsible for is the processing of olfactory information. Recent studies have shown the possibility of decoding the identity of odors from the spiking activity of hippocampal neurons [58] [31] [36]. The decoding of odor information from hippocampal spike train data involves several steps. First, the spiking activity of individual neurons is recorded using electrodes implanted on the animal's brain while the animal is exposed to different odors. This activity is then analyzed to identify neurons that respond selectively to specific odors, typically using statistical methods such as Principal Component Analysis (PCA) [3]. Once the odor-selective neurons have been identified, the next step is to decode the identity of the odors from their spiking activity using various machine learning algorithms.

One of the challenges in decoding odor information from hippocampal spike train data is the noise and sparsity of the spiking activity of individual neurons. To alleviate this issue, multiple trials of odor presentation are typically used to increase the signal-to-noise ratio. Additionally, the spiking activity of multiple neurons is often combined to improve the decoding accuracy.

The main motivation of our proposed methodology is to decode the latent time during certain events known as *replays* [47]. Hippocampal replays refer to the phenomenon of the hippocampus replaying previously experienced events during periods of rest or sleep. During replays, the hippocampus appears to reactivate the same pattern of neural activity that was active during the original experience. This replay process is thought to be important for consolidating memories and for extracting commonalities or generalizations from past experiences that can be applied to future situations.

The structure of this paper is as follows. Section 5.2 provides a brief overview of point processes and their log-likelihood computation. Section 5.2.2 gives a brief introduction to Gaussian processes and their reduced-rank approximations. In Section 5.3, we formally introduce our proposed neural encoding and decoding models. We present simulation results in Section 5.4.1 and evaluate the proposed models in terms of their neural decoding accuracy using data from a rat electrophysiology experiment in Section 5.4.2. Finally, we discuss future directions and possible extensions of our method in Section 5.5.

5.2 Background

5.2.1 Point Processes

Point processes [18] are special type of stochastic processes that model the occurrence of events over time or space. The *counting process* associated with a point process, denoted by $N(t)$ for $t \geq 0$, is itself a stochastic process that counts the number of events that have occurred up to time t. Another fundamental tool in the study of point processes is the *conditional intensity function (CIF)* that is defined as the instantaneous rate of events at time t given the history of the events:

$$\lambda(t) = \lim_{h \to 0} \frac{\Pr(N(t+h) - N(t) = 1 \mid \mathcal{H}_t)}{h} \tag{5.1}$$

where $\mathcal{H}_t$ is the history of the events up to time t. In fact, the intensity function fully describes the probability distribution of a point process [18].

Non-Homogeneous Poisson Process (NHPP) are a specific type of point process with a time-varying intensity function $\lambda(t) \geq 0$ that has the following properties:

1. $N(0) = 0$; that is, the process starts fresh with no events

2. $N(t)$ has independent increments; that is, the number of events in non-overlapping intervals are independent

3. $\mathbf{P}\left(N(t,t+h]=1\right)=\lambda(t)h+o(h)$ as $h\to 0$

4. $\mathbf{P}\left(N(t,t+h]\geq 2\right)=o(h)$ as $h\to 0$

In a multivariate point process, events occur in multiple sub-processes, each with its own intensity function. The occurrence of events ineach sub-process may depend not only on its own past history but also on the past history of events in other sub-processes. The intensity function for each sub-process specifies the probability of an event occurring in that sub-process at a given time, conditional on the past history of events in all sub-processes. Suppose we observe the spiking activity of M neurons over the time interval $(0,T]$. Denote the counting process of neuron m by $N_m(t)$. The spiking activity of the ensemble during $(0,t]$ is represented by $\mathbf{N}(t)=(N_1(t),\ldots,N_M(t))$. Then, the m-th sub-process is fully determined by its CIF:

$$\lambda_m(t\mid\mathcal{H}_t)=\lim_{h\to 0}\frac{\Pr(N_m(t+h)-N_m(t)=1\mid\mathcal{H}_t)}{h}\tag{5.2}$$

where $\mathcal{H}_t$ is the history of ensemble activity and the stimuli up to time t. The CIF in Equation 5.2 is interpreted as the instantaneous expected rate of spiking of neuron m at time t conditioned on the history of the ensemble up to time t.

Likelihood of a Point Process

The log-likelihood of a univariate point process $N(t)$ with event times $\{t_i\}_{i=1,\ldots,N(T)}$ observed over the time interval $(0,T]$ is computed as follows:

$$\log\mathcal{L}\left(\boldsymbol{\theta}\mid\{t_i\}_{i=1,\ldots,N(T)}\right)=\sum_{i=1}^{N(T)}\log\lambda(t_i\mid\boldsymbol{\theta})-\int_0^T\lambda(s\mid\boldsymbol{\theta})\,ds\tag{5.3}$$

where $\boldsymbol{\theta}$ contains all of the parameters of the point process model. The term $\Lambda(T) = \int_0^T \lambda(s \mid \boldsymbol{\theta})\, ds$ is known as the *compensator* of the point process.

In the case of a multivariate point process $\mathbf{N}(t)$ with event times $\{t_{m,i}\}_{i=1,\dots,N_m(T),\ m=1,\dots,M}$ observed over the time interval $(0, T]$, the log-likelihood is simply the sum of the marginal log-likelihood of each sub-process. That is,

$$\log \mathcal{L}\left(\boldsymbol{\theta} \mid \{t_{m,i}\}_{i=1,\dots,N_m(T),\ m=1,\dots,M}\right) = \sum_{m=1}^{M} \left(\sum_{i=1}^{N_m(T)} \log \lambda_m(t_{m,i} \mid \boldsymbol{\theta}) - \int_0^T \lambda_m(s \mid \boldsymbol{\theta})\, ds \right) \tag{5.4}$$

Using the log-likelihood function given in Equations 5.3 and 5.4, one can calculate the maximum likelihood estimates to determine the model parameters accurately. Alternatively, one can employ Bayesian inference on the model parameters by using appropriate priors, and use the resulting posterior distributions to perform neural decoding.

5.2.2 Gaussian processes

Gaussian processes (GP) [51] are stochastic processes that characterizes the distribution over a set of random variables. A Gaussian process is a distribution over the set $\mathcal{F} = \{f(x) : x \in \mathcal{X}\}$ for some index set $\mathcal{X}$. An important property of Gaussian processes is that any finite subset of random variables from $\mathcal{F}$, namely $\{f(x_1), f(x_2), \dots, f(x_n)\}$, has a multivariate Gaussian distribution. A Gaussian process is fully characterized by its mean and covariance functions:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{5.5}$$

$$m(x) = \mathbb{E}(f(x)) \tag{5.6}$$

$$k(x, x') = \mathbb{E}\left[(f(x) - m(x)(f(x) - m(x))'\right] \tag{5.7}$$

The covariance function, which needs to be symmetric and positive semi-definite [51], represents our prior assumptions about the functional relationship between variables, including properties like continuity, smoothness, periodicity, and scale. A covariance function is called *stationary* if it only depends on the distance between the input points, $x - x'$. In other words, a stationary covariance function remains unaffected by any translations made in the input space. A covariance function is called *isotropic* if it solely dependents on the difference between input points, as expressed by the norm of the difference: $k(x, x') = k(|x - x'|)$, which implies that the covariance is invariant to both translation and rotation. A common choice for a stationary and isotropic covariance function is the *squared exponential* covariance function defined by:

$$\text{cov}\left(f(x), f(x')\right) = k(x, x') = \alpha^2 \exp\left(-\frac{\|x - x'\|^2}{2\rho^2}\right) \tag{5.8}$$

where the parameter ρ is known as the *length-scale* and is related to the frequency of the functions represented by the Gaussian process in the domain. When ρ is closer to zero, the GP can represent high-frequency functions, while larger values of ρ allow the GP to represent low-frequency functions. On the other hand, the hyperparameter α is referred to as the *marginal standard deviation* and regulates the range of the functions represented by the GP.

5.2.3 Hilbert Space approximate Bayesian Gaussian Processes

Gaussian processes are a powerful tool for modeling complex and non-linear functions, but they can be computationally expensive, particularly when dealing with large datasets. The main computational bottleneck is the inversion of the covariance matrix, which scales as $O(n^3)$ for n data points. This can make the inference process prohibitively slow, especially for real-world problems where the dataset size can be very large. Additionally, Gaussian processes are not very scalable, and their performance can deteriorate when applied to high-dimensional data (high in the number of covariates/features). This is because the covariance matrix becomes much more complex and difficult to estimate in higher dimensions. One common approach to alleviate these issues is to use reduced-rank Gaussian processes, which involve approximating the covariance matrix with a low-rank matrix. This can significantly reduce the computational burden while still retaining much of the modeling power of the original Gaussian process.

Stationary covariance functions can be expressed in terms of their spectral densities [51]. According to *Bochner's theorem*, stationary covariance functions can be represented as the Fourier transform of a positive finite measure. If this measure has a density function, it is referred to as the spectral density of the covariance function [51]. The spectral density functions associated with the squared exponential covariance function is given by:

$$S(s) = \alpha\sqrt{2\pi\rho^2}\exp\left(-\frac{1}{2}\rho^2 s^2\right) \tag{5.9}$$

Solin and Sarkka [59] proposed an approximate Gaussian process method, which involves approximating the covariance operator of a stationary covariance function as a pseudo-differential operator composed of a series of Laplace operators. This pseudo-differential operator is then approximated using Hilbert space techniques on a limited subset of real numbers, subject to certain boundary conditions.

The approximate Gaussian process method restricts the domain to a subset of the real numbers, denoted by Ω, which is defined as $[-L, L] \subset \mathbb{R}$ with L being a positive real number that serves as a boundary condition. Within this restricted space, any stationary covariance function can be expressed as a sum of products between the spectral density function S and the eigenvalues and eigenvectors of the Laplacian operator in Ω. Specifically, the covariance function can be written as:

$$k(x, x') = \sum_{i=1}^{\infty} S\left(\sqrt{\lambda_i}\right)\phi_i(x)\phi_i(x') \tag{5.10}$$

where $\lambda_i \in \mathbb{R}^+$ and ϕ_i are the sets of eigenvalues and eigenfunctions (sinusoidal functions) that are the solution to the following eigenvalue problem:

$$-\nabla^2 \phi_i(x) = \boldsymbol{\lambda}_i \phi_i(x) \quad \text{for} \quad x \in \Omega \tag{5.11}$$

$$\phi_i(x) = 0 \quad \text{for} \quad x \notin \Omega \tag{5.12}$$

The eigenvalues and eigenfunctions are independent of the specific choice of covariance function and are given by:

$$\boldsymbol{\lambda}_i = \left(\frac{i\pi}{2L}\right)^2 \tag{5.13}$$

$$\phi_i(x) = \sqrt{\frac{1}{L}} \sin\left(\sqrt{\boldsymbol{\lambda}_i}(x+L)\right) \tag{5.14}$$

In practice, we truncate the sum in Equation 5.10 to the first q terms.

$$k(x, x') = \sum_{i=1}^{q} S\left(\sqrt{\boldsymbol{\lambda}_i}\right) \phi_i(x)\phi_i(x') = \boldsymbol{\phi}(x)^T \Delta \boldsymbol{\phi}(x') \tag{5.15}$$

where $\boldsymbol{\phi}(x) = (\phi_1(x), \ldots, \phi_q(x))^T$ and $\Delta = \text{diag}\left(S\left(\sqrt{\boldsymbol{\lambda}_1}\right), \ldots, S\left(\sqrt{\boldsymbol{\lambda}_q}\right)\right)$. Then, the Gaussian process $f(\mathbf{x}) \sim \mathcal{GP}(0, k)$ for the input $\mathbf{x} = (x_1, \ldots, x_n)^T$ can be approximated by

$$f(\mathbf{x}) \sim \text{Normal}(0, \Phi\Delta\Phi') \tag{5.16}$$

where $\Phi \in \mathbb{R}^{n \times q}$ is the matrix of eigenfunctions:

$$\Phi = \begin{pmatrix} \phi_1(x_1) & \ldots & \phi_q(x_1) \\ \vdots & \ddots & \vdots \\ \phi_1(x_n) & \ldots & \phi_q(x_n) \end{pmatrix} \tag{5.17}$$

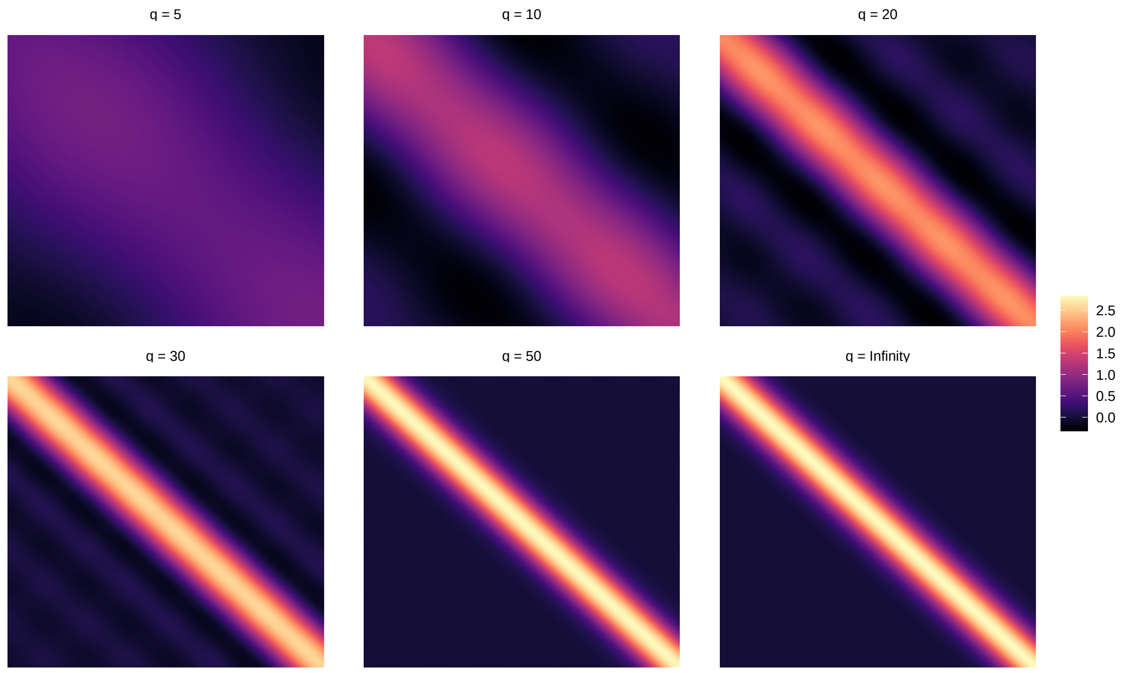


Figure 5.1: The Hilbert space approximation of the squared exponential covariance function ($\alpha = 1, \rho = 0.1$) is plotted for various numbers of eigenfunctions. The subplot in the lower right corner (labeled as q = infinity) represents the exact squared exponential covariance function. Larger number of eigenfunctions produce better approximation to the covariance function.

This is equivalent to

$$f(\mathbf{x}) = \sum_{i=1}^{q} \left(S\left(\sqrt{\lambda_i}\right)\right)^{1/2} \phi_i(\mathbf{x})\beta_i \tag{5.18}$$

$$\beta_i \sim \text{Normal}(0, 1) \tag{5.19}$$

Figure 5.1 shows the Hilbert space approximations to the squared exponential covariance function for different number of eigenfunctions. As long as the input values x are not too close to the boundaries $-L$ and L, using a finite number of q terms in the series expansion should provide a reasonable approximation [54].

5.3 Methods

Our proposed methodology in this chapter focuses on estimating the underlying neuronal firing mechanisms in various experimental conditions through *encoding*, followed by utilizing the estimated firing patterns for *decoding* the stimulus from new and unseen spike train data. Encoding involves training a model that captures the relationship between the stimulus and the spiking activity of the neurons. Once the model is trained, it can be used for decoding the stimulus from new spike train data by using the spiking activity patterns to infer the stimulus that was presented to the neurons.

To introduce the concepts of the encoding and decoding models, let's consider an experiment in which a rat is performing a sequence of cognitive tasks (a sequence of different stimuli), and we are recording spiking activity from an ensemble of M neurons. The experiment involves presenting a sequence of different stimuli to the rat, with each stimulus lasting for a certain fixed duration. Each sequence lasts for T seconds, and we have S sequences in total. This means that the experiment is repeated S times, and each time the spiking activity of M neurons is recorded. Let $\mathcal{Y}_{t,m,s} \in \{0,1\}$ denote the spike at time t in neuron m and sequence s, where $\mathcal{Y}_{t,m,s} = 1$ means the existence of the spike. Note that the time interval $(0, T]$ is discretized to millisecond increments such that at each millisecond bin there is at most one spike. Hence, the 3-dimensional array $\mathcal{Y}$ with dimension $1000T \times M \times S$ contains S sequences of M-variate point processes, each lasting for $1000T$ milliseconds. In the following sections, we will provide a detailed explanation of both the encoding and the decoding model. We will first discuss the encoding model, which is used to learn the underlying neuronal firing mechanisms under different experimental conditions. Then we will discuss the decoding model, which utilizes the learned firing patterns to decode the stimuli from a previously unseen spike train data.

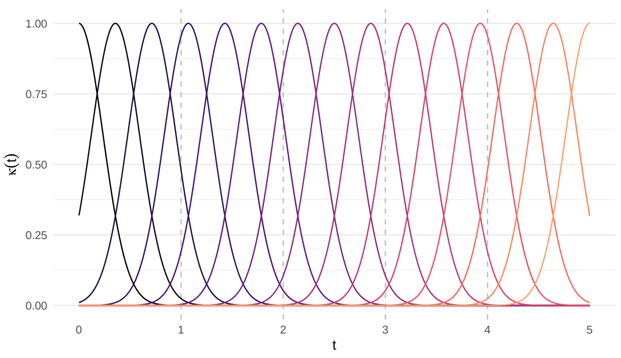


Figure 5.2: 15 RBF kernels with different centers distributed equally over the interval $[0, 5]$ and width $\gamma = 0.1$.

5.3.1 The Point-Process Encoding (PPE) Model

We assume that the M neurons in the sequence $\mathcal{Y}_{:,:,s}$ fire according to non-homogeneous Poisson processes with intensity functions $\lambda_m(t) \geq 0$ for $m = 1, \ldots, M$. The intensity function (firing rate) of neuron m over time is a smooth function that is a linear combination of B Radial Basis Function (RBF) kernels. That is, for $b = 1, \ldots, B$ and for $m = 1, \ldots, M$:

$$\lambda_m(t) = \sum_{b=1}^{B} \beta_{b,m} \kappa_b(t) \tag{5.20}$$

where $\beta_{b,m} \geq 0$ and $\kappa_b(t)$ is the RBF kernel defined by:

$$\kappa_b(t) = \exp\left(-\|t - \mu_b\|^2 / \gamma\right) \tag{5.21}$$

It is crucial to note that the $\beta_{b,m}$ parameters must be positive as the intensity function of a point process has to be non-negative. The μ_b parameters represent the centers for the RBF kernels and are pre-defined at B equidistant locations $(0, T/B, 2T/B, \ldots, T)$, while γ is the fixed shared scale (width) of these RBF kernels. It is worth noting that the intensity function in Equation 5.20 is not a mixture of Gaussian distributions since the $\sum_b \beta_{b,m} \neq 1$. Furthermore, the RBF kernels in Equation 5.21 do not integrate to 1. Figures 5.2 and 5.6 show the RBF basis functions and a set of simulated intensity functions for 10 neurons, respectively.

The encoding model for neuron m in sequence s can be described as follows:

$$\begin{aligned} \mathcal{Y}_{:,m,s} \mid \beta_{\cdot,m} &\sim \text{Poisson-Process}\left(\lambda_m(t)\right) \\ \lambda_m(t) &= \sum_{b=1}^{B} \beta_{b,m}\kappa_b(t) \\ p(\beta_{b,m} \mid \pi, \boldsymbol{\eta}) &\sim \pi p(\beta_{b,m} \mid \eta_{\text{spike}}) + (1-\pi)p(\beta_{b,m} \mid \eta_{\text{slab}}) \end{aligned} \tag{5.22}$$

for $m = 1, \ldots, M$ and $t \in (0, T)$. We put a *spike-and-slab* shrinkage prior [39] on the $\beta_{b,m}$ parameters to shrink the irrelevant parameters to zero. The other parameters get the following priors:

$$\begin{aligned} \pi &\sim \text{Beta}(a, b) \\ \eta_{\text{spike}} &\sim \text{exponential}(c) \\ \eta_{\text{slab}} &\sim \text{exponential}(d) \end{aligned} \tag{5.23}$$

where $\frac{a}{a+b}$ is the prior expected number of irrelevant β coefficients. In particular, we use a Beta$(5, 5)$ prior for the π parameter, resulting in a prior belief that half of the parameters are irrelevant. The η_{spike} and η_{slab} parameters are assigned exponential prior distributions with hyper-parameters c and d, where $c \gg d$. We select c to be large enough such that the majority of the mass of the distribution for η_{spike} is concentrated near zero. Whereas, d is chosen such that the distribution for η_{slab} has considerable mass away from zero. In particular, we chose $c = 1000$ and $d = 1$. The plate diagram of the encoding model is shown in Figure 5.3. The encoding model attempts to learn the intensity functions by estimating the $\beta_{b,m}$ parameters. By examining the posterior distribution of these parameters, we can determine which neuron selectively fires for each stimulus. For example, if $\beta_{b,m}$ is approximately zero, it suggests that the intensity function of neuron m is negligible during the time interval that is covered under the b-th RBF basis function. The inputs to the encoding model include the spikes $\mathcal{Y}$, the RBF hyper-parameters $\boldsymbol{\mu}$ and γ, and the corresponding time information $t \in \{0.001, 0.002, \ldots, T - 0.002, T - 0.001, T\}$.

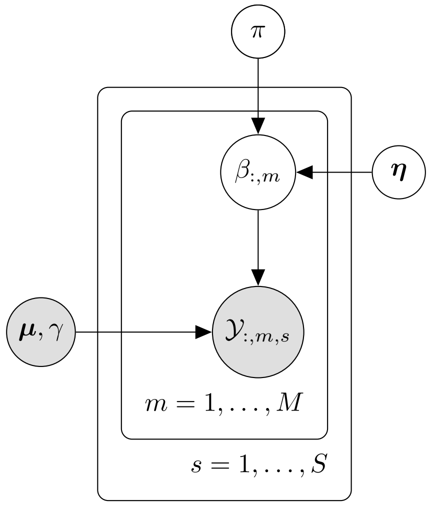


Figure 5.3: Graphical representation of the encoding model in Equations 5.22. $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_B)$, $\boldsymbol{\eta} = (\eta_{\text{spike}}, \eta_{\text{slab}})$

5.3.2 Likelihood Function of the Encoding Model

Recall that the observation window of the spike train is divided into millisecond bins, so in total we have $J = 1000T$ time stamps. Let K be a matrix of size $J \times B$ containing the values of RBF kernels over time:

$$K = \begin{pmatrix} \kappa_1(t_1) & \ldots & \kappa_B(t_1) \\ \kappa_1(t_2) & \ldots & \kappa_B(t_2) \\ \vdots & \ddots & \vdots \\ \kappa_1(t_J) & \ldots & \kappa_B(t_J) \end{pmatrix} \tag{5.24}$$

Let L be a matrix of size $J \times M$ containing the values of the intensity functions over time for the M neurons:

$$L = K\mathcal{B} \tag{5.25}$$

where $\mathcal{B}$ is an $B \times M$ with elements $(\mathcal{B})_{b,m} = \beta_{b,m}$. The m-th column of L contains the values of the intensity function of neuron m; that is, $L_{:,m} = (\lambda_m(0.001), \ldots, \lambda_m(T))'$. The contribution of sequence s to the log-likelihood of the encoding model is as follows:

$$\ell^{(s)}\left(\mathcal{B} \mid \mathcal{Y}_{:,:,s}\right) = \sum_{m=1}^{M} \left(\sum_{j=1}^{J} \mathcal{Y}_{j,m,s} \log \lambda_m(t_j \mid \mathcal{B}_{:,m}) - \int_0^T \lambda_m(s \mid \mathcal{B}_{:,m})\, ds \right) \tag{5.26}$$

where $t_j \in \{0.001, 0.002, \ldots, T-0.002, T-0.001, T\}$. The integral in Equation 5.26 is known as the *compensator* of the point process and is computed as follows:

$$\begin{aligned}
\Lambda_m(T) &= \int_0^T \lambda_m(s \mid \mathcal{B}_{:,m})\, ds && (5.27)\\
&= \int_0^T \sum_{b=1}^{B} \beta_{b,m} \kappa_b(s)\, ds \\
&= \sum_{b=1}^{B} \beta_{b,m} \int_0^T \kappa_b(s)\, ds \\
&= \sum_{b=1}^{B} \beta_{b,m} \int_0^T \exp\left(-\frac{1}{\gamma}(s-\mu_b)^2\right) ds \\
&= \sqrt{\gamma} \sum_{b=1}^{B} \beta_{b,m} \int_{-\mu_b/\sqrt{\gamma}}^{(T-\mu_b)/\sqrt{\gamma}} e^{-u^2}\, du \\
&= \frac{\sqrt{\gamma\pi}}{2} \sum_{b=1}^{B} \beta_{b,m} \left[\text{erf}\left(\frac{\mu_b}{\sqrt{\gamma}}\right) + \text{erf}\left(\frac{T-\mu_b}{\sqrt{\gamma}}\right) \right] && (5.28)
\end{aligned}$$

where $\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\, dt$. Hence, the contribution of sequence s to the log-likelihood function can be written as follows:

$$\begin{aligned}
\ell^{(s)}\left(\mathcal{B}_{:,m} \mid \mathcal{Y}_{:,:,s}\right) = \sum_{m=1}^{M} \Bigg(&\sum_{j=1}^{J} \mathcal{Y}_{j,m,s} \log \lambda_m(t_j \mid \mathcal{B}_{:,m}) && (5.29)\\
&- \frac{\sqrt{\gamma\pi}}{2} \sum_{b=1}^{B} \beta_{b,m} \left[\text{erf}\left(\frac{\mu_b}{\sqrt{\gamma}}\right) + \text{erf}\left(\frac{T-\mu_b}{\sqrt{\gamma}}\right) \right] \Bigg) && (5.30)
\end{aligned}$$

and the log-likelihood for the entire training data set is going to be equal to:

$$\ell\left(\mathcal{B}_{:,m} \mid \mathcal{Y}\right) = \sum_{s=1}^{S} \ell^{(s)}\left(\mathcal{B}_{:,m} \mid \mathcal{Y}_{:,:,s}\right) \tag{5.31}$$

5.3.3 The Point-Process Decoding (PPD) Model

The objective of the decoding model is to predict or decode the stimuli within a previously unseen spike train that contains the spiking activity of the same number of neurons. This unseen spike train data could correspond to brain activity during periods of rest or performance of a different tasks than the one used during the encoding phase. Hence, this test spike train can have a duration that is different from the spike train data used in the encoding model. Alternatively, the decoding model can be used to correlate the spiking activity with the stimuli pattern in order to gain insights into the brain's information processing mechanisms.

Let us consider that P stimuli are presented during a sequence of cognitive tasks between the time interval $(0, T]$, where stimulus p is presented during the time interval $\left(\frac{(p-1)T}{P}, \frac{pT}{P}\right]$ for $p = 1, \dots, P$. This allows us to divide the total observation time $(0, T]$ into P equal-sized sub-intervals, each corresponding to a particular stimulus. Unlike traditional classification tasks, the proposed decoding model does not predict a discrete label for each of the stimuli, but rather a continuous value representing the temporal location of the stimuli within the observation window $(0, T]$. We refer to this continuous value as the *latent time*, denoted by $\boldsymbol{\tau} \in (0, T]$. We can infer which of the stimuli were *replayed* or decoded during the spike train by analyzing the value of $\boldsymbol{\tau}$. For instance, if $\boldsymbol{\tau} \in \left(\frac{(p-1)T}{P}, \frac{pT}{P}\right]$, we can conclude that the first stimulus p was being replayed/decoded in that moment of time.

Let $\mathcal{Y}'_{t',m}$ denote the spike train that we want to decode the temporal information from, where $t' \in (0, T')$ and $m = 1, \dots, M$; that is $\mathcal{Y}'$ is a spike train with duration T'. We assume that the underlying stimuli that evoked the spiking activity of the neurons is continuous and varies smoothly across the time interval $(0, T')$. To decode the latent time $\boldsymbol{\tau}$ of the stimulus, we employ a Gaussian process prior for the functional form of $\boldsymbol{\tau}$. This prior assumes that the decoded latent time varies smoothly across the time interval $(0, T')$. However, we need to apply some constraints to the Gaussian process prior, as the range of τ is limited to $(0, T]$. In addition, we use the posterior point estimates obtained from the encoding model, namely $\hat{\beta}_{b,m}$, $\hat{\kappa}_b$, and $\hat{\lambda}_m$, to decode the latent time of the stimulus.

For neuron m, the decoding model is as follows:

$$\begin{aligned}
\mathcal{Y}'_{t',m} &\sim \text{Poisson-Process}\left(\hat{\lambda}_m(t')\right) \\
\hat{\lambda}_m(t') &= \sum_{b=1}^{B} \hat{\beta}_{b,m}\hat{\kappa}_b(\boldsymbol{\tau}(t')) \\
\boldsymbol{\tau}(t') &= T \cdot \text{sigmoid}(\beta_0 \mathbf{1} + \mathbf{f}(t')) \\
\mathbf{f}(t') &\sim \mathcal{GP}\left(\mathbf{0}, \mathcal{K}\right) \\
\beta_0 &\sim \text{Normal}(0,1)
\end{aligned} \tag{5.32}$$

for $t' \in (0, T']$. The latent function $\mathbf{f}(t')$ is an unconstrained Gaussian process over the time interval $(0, T')$ with a zero mean and squared exponential covariance function $\mathcal{K}$ with marginal standard deviation α and length-scale ρ. The function $\mathbf{f}$ is then passed through a sigmoid$(x) = 1/(1 + e^{-x})$ function and multiplied by T to ensure that the range of the output function, $\boldsymbol{\tau}(t)$, is restricted to the interval $(0, T)$. The parameter β_0 acts as a bias term which moves the latent GP vertically between the $(0, T]$ boundary. The resulting function $\boldsymbol{\tau}(t)$ represents the decoded latent time, indicating the decoded stimuli. Note that the latent Gaussian process $\mathbf{f}(t')$ is approximated using the results in section 5.2.3. We use the following priors for the model parameters: $\beta_0 \sim$ Normal$(0, 1)$, $\rho \sim$ gamma$(1, 1)$, and $\alpha \sim$ gamma$(1, 1)$.

5.4 Results

In this section, we demonstrate the application of the encoding and decoding models through two examples. The first example employs simulated neural spike data to evaluate the performance of the models through goodness-of-git (GOF) tests (for the encoding model) and predictive per-

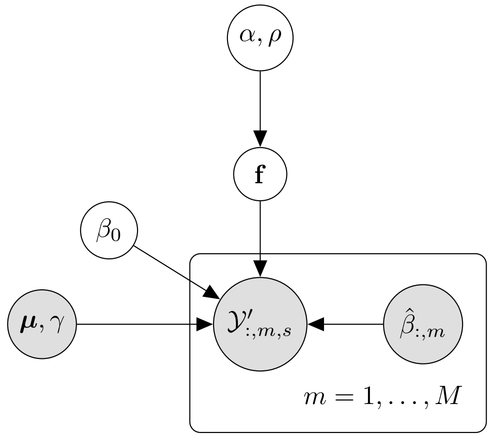


Figure 5.4: Graphical representation of the decoding model in Equations 5.32. The inputs to this model are $\mathcal{Y}'_{:,m,s}$, $\hat{\beta}_{:,m}$, $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_B)$, and γ, which are the unseen test sequence, the posterior point estimates from the encoding model, the RBF kernel centers, and the common RBF kernel width, respectively. α is the marginal standard deviation and ρ is the length-scale of the Gaussian process $\mathbf{f}$.

formance (for the decoding model). Furthermore, we compare the performance of the decoding model to other existing methods for neural decoding. The second example applies the model to an ensemble of neurons recorded from the CA1 region of the hippocampus in rats. The performance of the models on real data is evaluated on a left-out dataset (with known ground truth) as well as on spike trains during inter-sequence periods (without ground truth).

5.4.1 Simulation study

In this simulation study, we demonstrate the key characteristics of the models presented in Equations 5.22 and 5.32. Furthermore, we aim to show that the model parameters can be precisely estimated from a test spike data using the posterior samples from the fitted models.

We simulate intensity functions for $M = 10$ neurons over the time interval $(0, T] = (0, 5]$ and $P = 5$ different stimuli A, B, C, D, and E. To do so, we use $B = 15$ RBF basis kernels with fixed centers that are evenly spaced within the interval $[0, 5]$, with a common width of $\gamma = 0.1$ as depicted in Figure 5.2. In order to control the impact of each stimulus on the intensity functions,

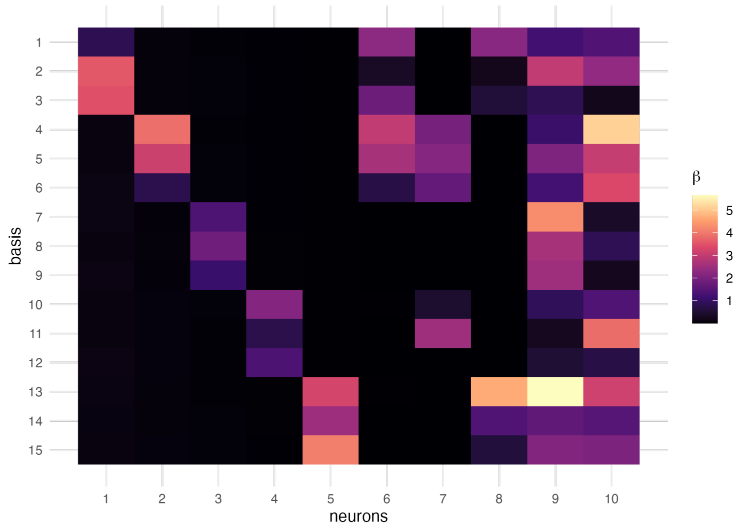


Figure 5.5: The heatmap of the $\mathcal{B}$ matrix. Neuron 1 in only selective for stimulus A, neuron 2 for B, neuron 3 for C, neuron 4 for D, and neuron 5 for E. Furthermore, neuron 6 is selective for stimuli A and B, neuron 7 for stimuli B and D, and neuron 8 for stimuli A and E. Neuron 9 and 10 are not intentionally selective for any specific stimuli. The dark sections of the heatmap correspond to values generated from uniform$(0.07, 0.1)$, while the rest are generated independently from a gamma$(2, 1)$ distribution.

certain neurons are forced to fire only in response to specific stimuli. This is accomplished by intentionally setting certain β parameters to very small values when a neuron is not responsive to a stimulus, and setting them to large values when the neuron is active in response to that stimulus. Specifically, we choose the structure shown in Figure 5.5 for the $\mathcal{B}$ matrix. In this structure, we selectively shrink $\approx 60\%$ of the parameters to very small values to induce sparsity. The resulting simulated intensity functions for the 10 neurons are shown in Figure 5.6.

The simulation of the intensity functions is followed by the simulating spiking activity for a complete sequence of five stimuli through a thinning algorithm (Algorithm 1 in Section 2.1.3). More specifically, the simulated spikes follow a non-homogeneous Poisson process, where the firing rate of each neuron is equivalent to the corresponding intensity function. We simulate $S = 50$ different sequences of spike train data according to the intensity functions depicted in Figure 5.6. The resulting spikes are shown in Figure 5.7.

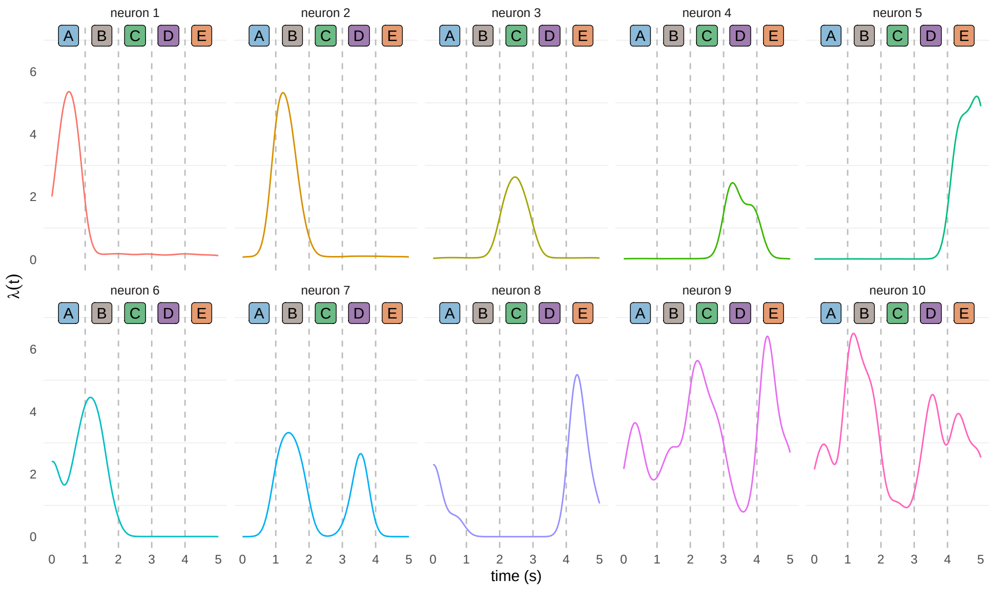


Figure 5.6: Simulated intensity functions of 10 neurons using the β parameters shown in Figure 5.5. Dashed vertical lines represent the separation between stimuli.

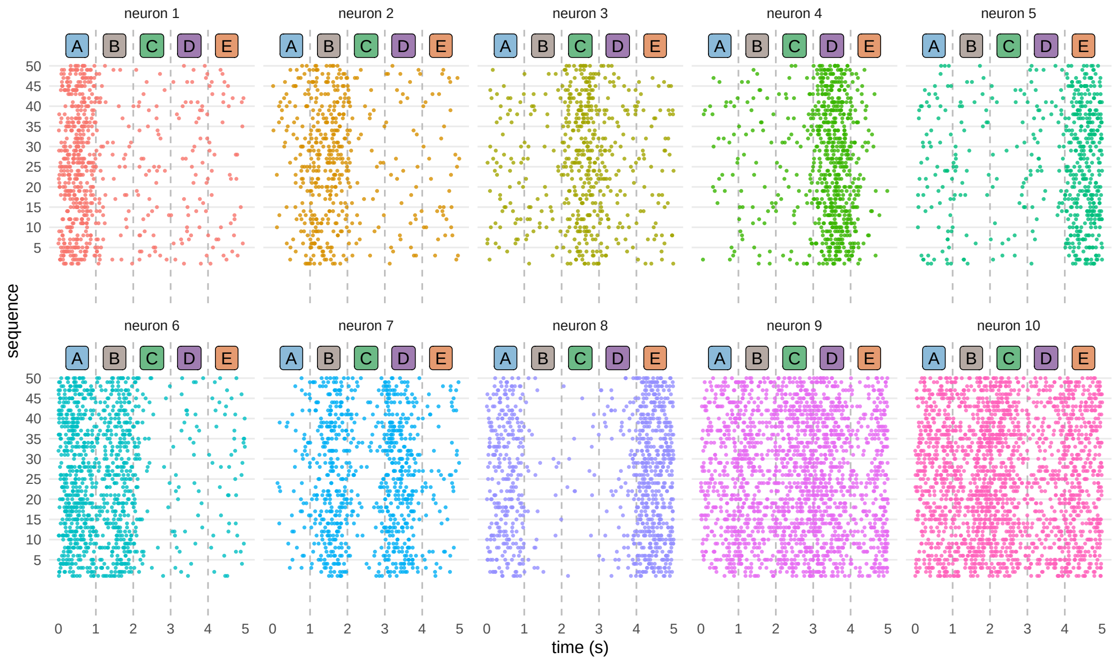


Figure 5.7: $S = 50$ sequences of simulated spikes odors according to the intensity functions in Figure 5.6. Each sub-plot depicts the spikes of a single neuron over time. The spikes in each neuron match the dynamics of its corresponding intensity function in Figure 5.6. Dashed vertical lines represent the separation between stimuli.

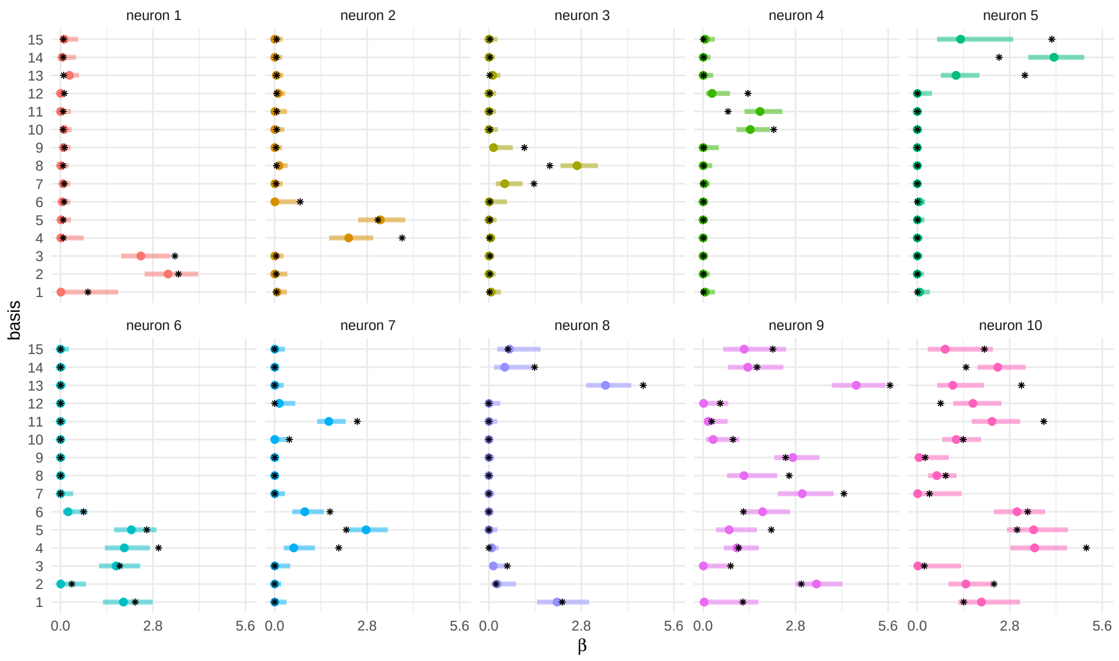


Figure 5.8: Posterior summary of the β parameters of the encoding model. The horizontal bars are the posterior 95% credible intervals color coded based on their corresponding neurons. The colored dots represent the posterior medians, and the black asterisks are the true values of the parameters.

Encoding model

With the simulated spike train data prepared and the ground truth for the intensity functions determined, we can now fit the encoding model in Equation 5.22 and evaluate its performance. Figure 5.8 shows the posterior median and the posterior 95% credible intervals of the $\beta_{b,m}$ parameters. We observe that the posterior distribution of the encoding model parameters exhibits a close match to the actual values selected for the simulation. Due to the spike-and-slab prior imposed on the $\beta_{b,m}$ parameters, the posterior distribution of the model becomes more compressed for the irrelevant parameters. Interestingly, the posterior credible intervals are able to capture approximately 80% of the true values.

The estimated intensity functions, represented by $\hat{\lambda}_m$, are shown in Figure 5.9, where we can observe that the posterior distribution of the intensity functions closely match the ground truth selected during the simulation process.

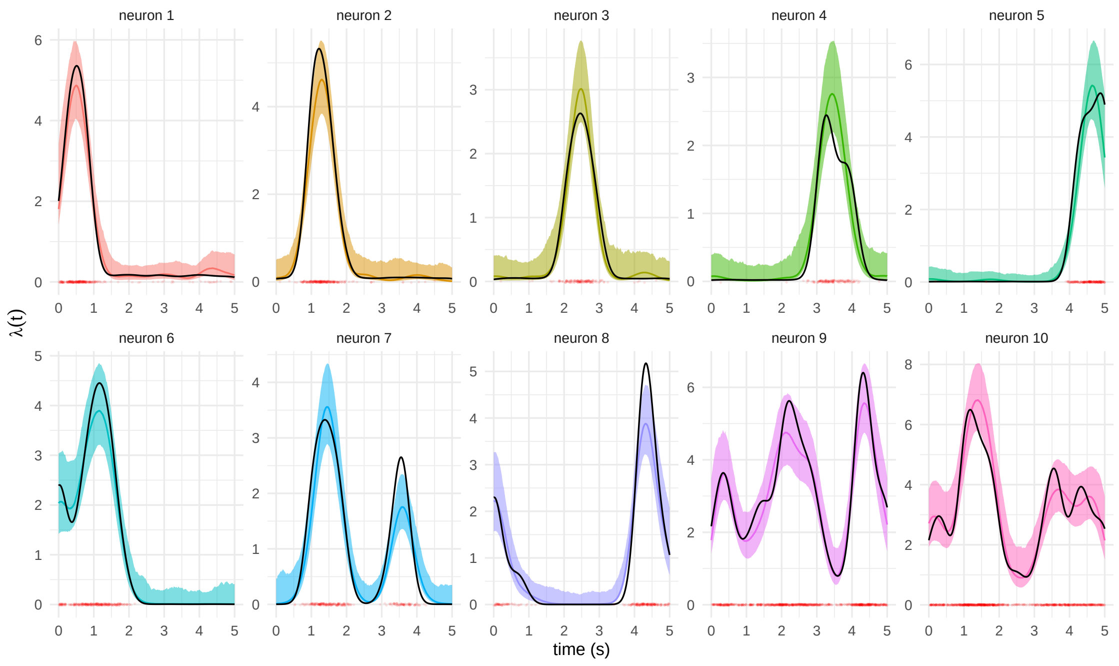


Figure 5.9: The posterior distribution of the intensity functions computed from the encoding model. The colored solid lines are the point-wise posterior median of the $\hat{\lambda}$ values, and the colored regions are the 95% posterior credible intervals. The black curves represent the ground truth.

Choosing the Number of RBF Kernsls

Selecting the number of RBF basis kernels is an important decision to be made before fitting the encoding model. This hyperparameter influences the smoothness of the intensity functions and can significantly impact the model's performance. To identify the optimal number of basis kernels, we can fit the encoding model with different numbers of RBF kernels and use the deviance information criterion (DIC) [23] as a selection criterion.

The DIC is defined by DIC $= -2\ell(\boldsymbol{\theta} \mid \mathcal{Y})$ where $\ell(\boldsymbol{\theta} \mid \mathcal{Y})$ is the log-likelihood of the training dataset. Figure 5.10 shows that the optimal number of RBF basis functions is indeed equal to the one chosen in the simulation.

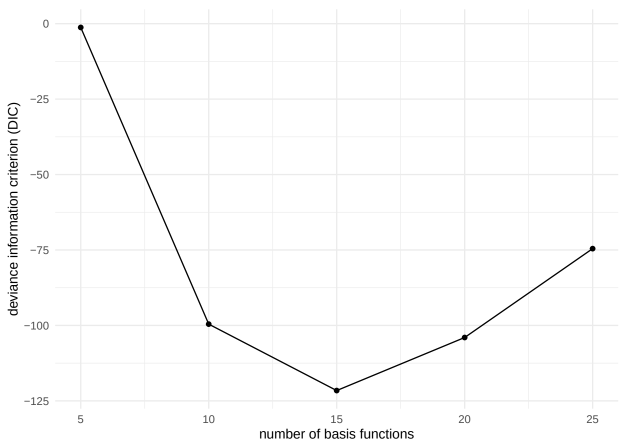


Figure 5.10: The posterior medians of the deviance information criterion (DIC) evaluated for different number of RBF kernels. The minimum DIC is achieved when the number of RBF kernels is exactly equal to the number chosen in the simulation.

Goodness-of-Fit Test

Assessing the adequacy of a point process model in fitting the observed spike data is a crucial practical consideration. The compensator of the point process and the well-known Time-Rescaling Theorem [18] are important tools in conducting this evaluation. The *Time-Rescaling Theorem* states that the for a realization $0 \leq t_1 < t_2 < \cdots < t_{N(T)} \leq T$ from a point process $N(.)$ with the intensity function $\lambda(t) \geq 0$, the sequence $\{\Lambda(t_1), \Lambda(t_2), \ldots, \Lambda(t_{N(T)})\}$ forms a Poisson process with unit rate, where

$$\Lambda(t_k) = \int_0^{t_k} \lambda(u)\, du < \infty \tag{5.33}$$

is known as the compensator. (See [8] for the proof)

To assess the goodness-of-fit of the encoding model, a non-parametric Kolmogorov-Smirnov (KS) test can be conducted by comparing the values $\{\Lambda_m(t_1), \Lambda_m(t_2), \ldots, \Lambda_m(t_{N(T)})\}$ of neuron m to those of a unit-rate Poisson process. As inter-arrival times of events in a unit-rate Poisson process follow an exponential(1) distribution, we can compare the values $\{\Lambda_m(t_k + 1) - \Lambda_m(t_k)\}_{k=1,\ldots,N(T)}$ to an exponential(1) distribution. The quantile-quantile plots of the time-rescaled spikes, along with the p-values of their corresponding Kolmogorov-Smirnov tests, are presented in Figure 5.11.

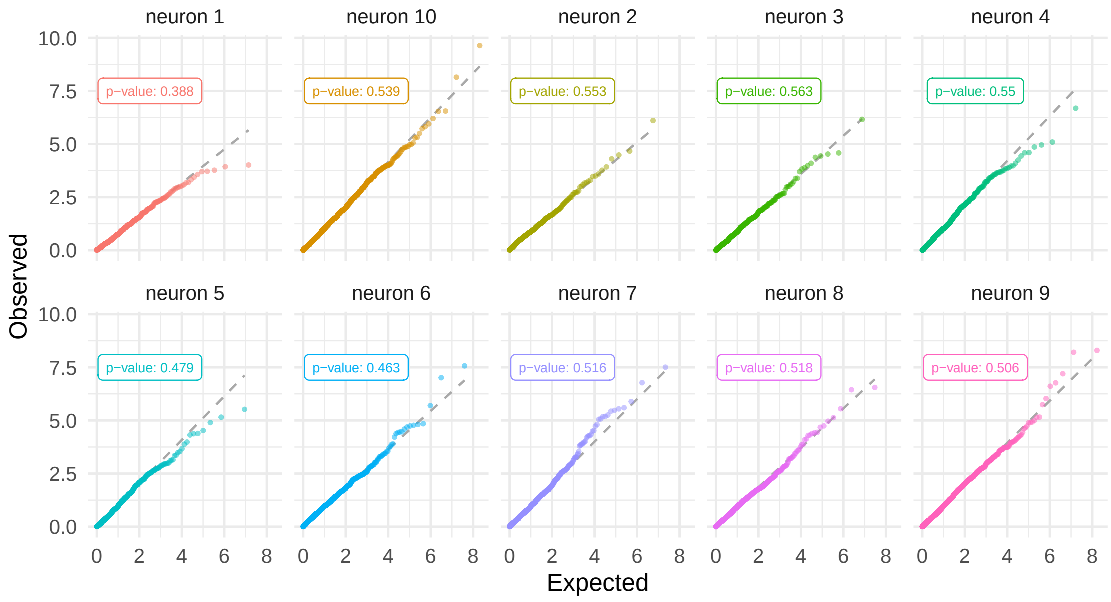


Figure 5.11: Goodness-of-fit quantile-quantile plots for the time-rescaled spikes for the $S = 50$ simulated spike trains. The inter-arrival times of the time-rescaled spikes are compared to the theoretical quantiles of the exponential(1) distribution. The p-values of the Kolmogorov-Smirnov (KS) tests are shown on each sub-plot. Large p-values are indicative of a good fit using the encoding model.

Decoding model

In order to evaluate the accuracy of the decoding model, we use a set of simulated spikes that were not used in the training of the encoding model to infer the latent time, which represents the continuous temporal location of the stimuli. The performance of the decoding model can be accurately assessed as we have access to the ground truth of the stimuli. Figure 5.12 presents the result of applying the decoding model to a new set of simulated test spike trains (previously unseen by the encoding model) that correspond to the intensity functions of a sequence of A, B, C, D, and E stimuli, each lasting for 0.5 seconds, over the time interval $(0, 2.5]$ seconds.

The decoding model accurately estimates the latent time from the spiking activity, as demonstrated by the posterior of the decoded time in Figure 5.12. The progression of the predicted stimuli in the test spike train is also visually verified by the posterior. Furthermore, the accuracy of the decoding

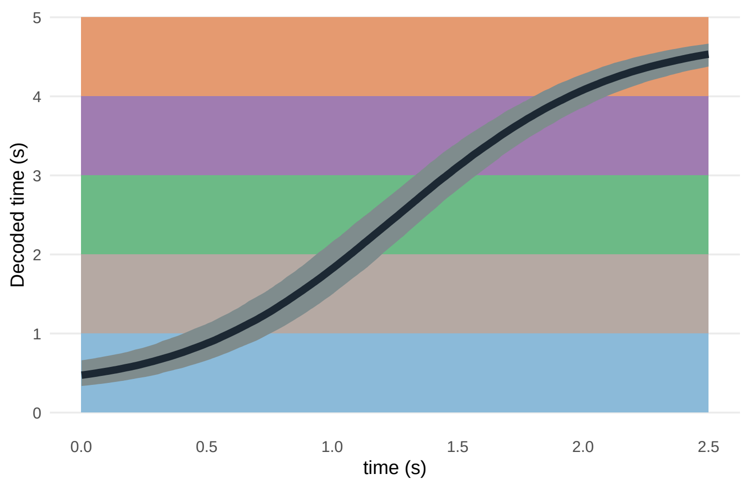
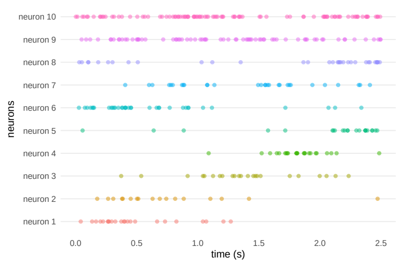
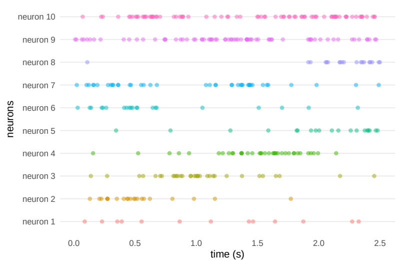


Figure 5.12: The decoded latent time from a sequence of test spikes (decoding accuracy = 91%). The black curves represent the point-wise posterior median of the decoded time, and the gray region around the curve is the posterior 95% pointwise credible intervals. The y-axis in the plots represents the decoded time that spans between 0 and $T = 5 \times 1.2 = 6$ seconds. The x-axis, which spans from 0 to 2.5 seconds, is the duration of spike train data that are simulated. The background contains horizontal stripes to indicate the stimuli: A: 🟦 B: ⬜ C: 🟩 D: 🟪 E: 🟧

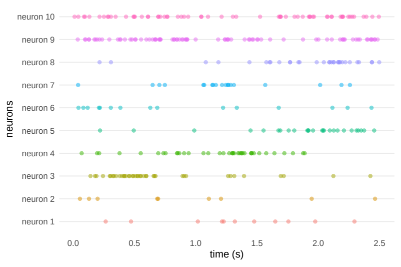
model can be numerically verified by binning the x-axis into 25-ms bins, which results in 100 bins in total, and determining the region of the y-axis where the posterior median falls. This allows us to assess the precision of the decoded time estimates. The proposed methodology was originally designed to decode the latent time of memory replays – brief periods of activity during which the hippocampus reactivates past experiences during periods of rest or sleep. These replay processes are critical for consolidating memories and extracting generalizations from past experiences [47]. The latent time during the replay event represents the temporal order of the stimuli being replayed by the hippocampus. We tested six different replay structures, including ABCDE, BCDE, CDE, EDCBA, CDEAB, and AB, by changing the order of stimuli and generating new test spike train data based on the revised stimulus orders. We then decoded the temporal order of stimuli during the simulated replay events using the proposed methodology. The spike trains corresponding to these replay events and their corresponding posterior decoded latent times are presented in Figures 5.13 and 5.14, respectively. The decoding accuracies for the different replay structures were as follows: $\text{acc}_{\text{ABCDE}} = 91\%$, $\text{acc}_{\text{BCDE}} = 91\%$, $\text{acc}_{\text{CDE}} = 90\%$, $\text{acc}_{\text{EDCBA}} = 73\%$, $\text{acc}_{\text{CDEAB}} = 86\%$,



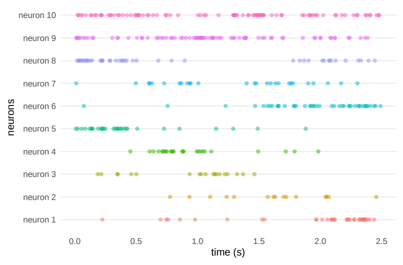
(a) A⇒B⇒C⇒D⇒E



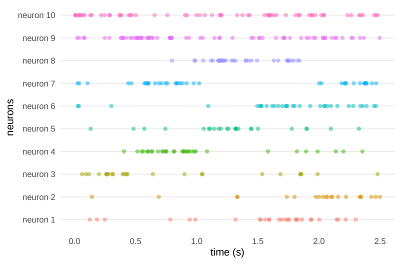
(b) B⇒C⇒D⇒E



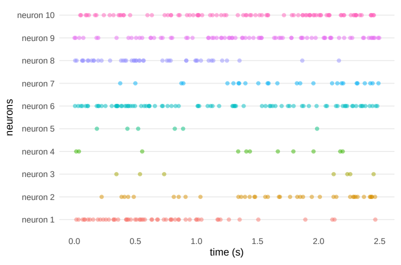
(c) C⇒D⇒E



(d) E⇒D⇒C⇒B⇒A



(e) C⇒D⇒E⇒A⇒B

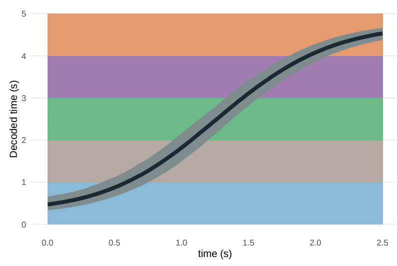


(f) A⇒B

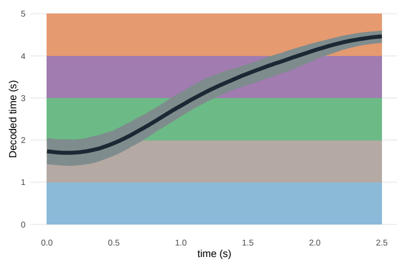
Figure 5.13: Simulated spike train data for 8 neurons for a duration of 1 second generated according to the stimulus patterns described for each subplot.

and $\text{acc}_{\text{AB}} = 83\%$.

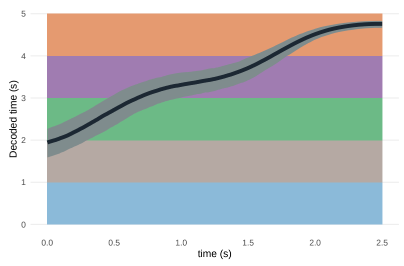
We compared the performance of our decoding model with other existing methods, including multinomial logit regression (MLR), penalized multinomial logit regression (MLR + LASSO), and random forest (RF). First, we trained the encoding model on 40 sequences of simulated spike trains shown in Figure 5.7 and estimated the posterior medians of the intensity functions of each neuron. We then used these estimated posterior intensity functions as input to train the MLR, MLR + LASSO, and RF models where the stimuli labels at each time bin were the response variables. Afterward, we tested the classifiers on the estimated intensity functions of the remaining 10 sequences of simulated spike trains shown in Figure 5.7 and calculated different performance metrics. The results of this comparison are presented in Table 5.1. It is evident that the decoding model outperformed the other methods in terms of accuracy. We also fed the raw spike trains as inputs to these classifiers, but the results were extremely poor. This outcome was anticipated as the raw spike trains are a stochastic manifestation of the underlying intensity functions. This indeed verifies the need for using "firing rate" instead of raw spikes in analyzing spike train data. Further-



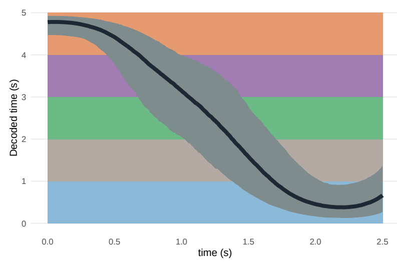
(a) A⇒B⇒C⇒D⇒E



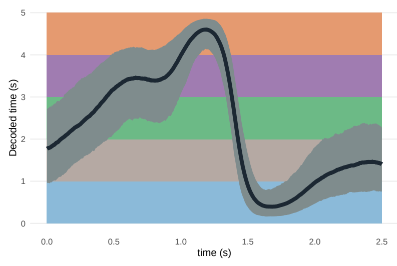
(b) B⇒C⇒D⇒E



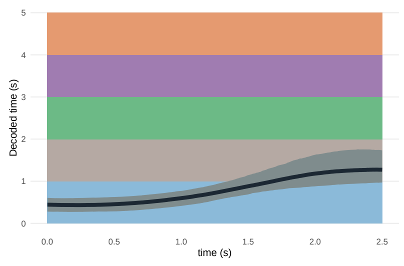
(c) C⇒D⇒E



(d) E⇒D⇒C⇒B⇒A



(e) C⇒D⇒E⇒A⇒B



(f) A⇒B

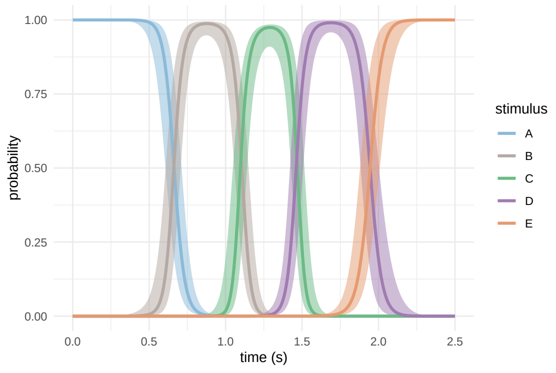
Figure 5.14: Decoding the latent time under multiple conditions. Decoding accuracies for each subplot are a) 91% b) 91% c) 90% d) 73% e) 86% f) 83%. The black curves represent the posterior pointwise median of the decoded time, and the gray regions around these curves are the posterior 95% pointwise credible intervals, representing the uncertainty of the decoded time. The y-axis in the plots represents the decoded time that spans between 0 and $T = 2.5$ seconds. The x-axis, which spans from 0 to 2.5 seconds, is the duration of spike train data that are simulated according to the stimulus patterns described for each subplot. In subplot (a), where the spike train is simulated according to the stimulus pattern ABCDE, we see that the decoded time starts from the blue region (odor A) and traverses all regions up to the orange region (odor E). In subplot (b), corresponding to the stimulus pattern BCDE, the decoded time starts from the region corresponding to odor B and then traverses up to odor E. In subplot (e), corresponding to the stimulus pattern CDEAB, we observe that the decoded time starts from the region corresponding to odor C and then traverses up to odor E. However, it then abruptly falls down to odor A and subsequently climbs back up to odor B. The decoding of the other subplots can be interpreted similarly. The background of each subplot contains horizontal stripes to indicate the stimuli: A: 🟦 B: 🟫 C: 🟩 D: 🟪 E: 🟧

Input data type	Method	Accuracy	F1 score	AUC
PPE + PPD		**(0.910, 0.026)**	**(0.838, 0.088)**	**(0.991, 0.008)**
PPE + Std. Classifiers	LASSO	$(0.760, 0.084)$	$(0.810, 0.095)$	$(0.976, 0.025)$
	RF	$(0.749, 0.080)$	$(0.769, 0.080)$	$(0.960, 0.027)$
	MLR	$(0.728, 0.076)$	$(0.711, 0.160)$	$(0.973, 0.049)$
Smoothed Spikes + Std. Classifiers	LASSO	$(0.363, 0.032)$	$(0.343, 0.04)$	$(0.688, 0.018)$
	RF	$(0.371, 0.033)$	$(0.349, 0.04)$	$(0.695, 0.019)$
	MLR	$(0.364, 0.032)$	$(0.343, 0.04)$	$(0.688, 0.018)$
Raw Spikes + Std. Classifiers	LASSO	$(0.210, 0.001)$	$(0.089, 0.011)$	$(0.521, 0.001)$
	RF	$(0.207, 0.002)$	$(0.090, 0.007)$	$(0.507, 0.002)$
	MLR	$(0.203, 0.002)$	$(0.091, 0.004)$	$(0.515, 0.011)$

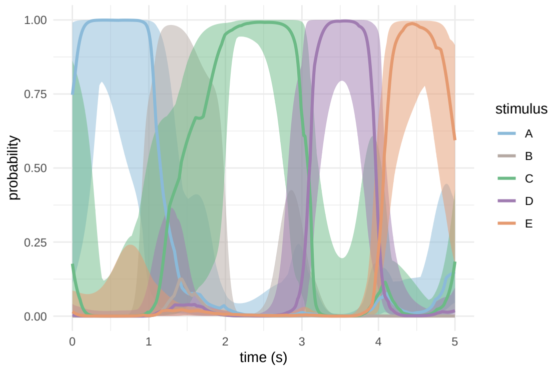
Table 5.1: Comparison of the performance of the decoding model with other existing methods, including multinomial logit regression (MLR), penalized MLR (MLR + LASSO), and random forest (RF), on simulated spike trains. The encoding model was trained on 40 sequences of simulated spike trains, and the estimated posterior intensity functions were used as input to train the classifiers. The classifiers were tested on the estimated intensity functions of the remaining 10 sequences (overall we have 50 simulated spike trains). The values in the parentheses are the mean and the standard deviation, respectively. Our proposed decoding model outperformed the other methods in terms of accuracy. We also used the smoothed spikes (using a convolution of a Gaussian kernel with width $= 20$ and sigma $= 4$) as an input, which resulted in a worse performance compared to the estimated intensities as input. Using the raw spike trains as inputs to the classifiers resulted in extremely poor performance, which confirms the necessity of analyzing spike train data using firing rates instead of raw spikes.

more, we compared these results with a commonly used smoothing technique in neuroscience in which the activity of each neuron, independent of the rest, is smoothed by convolving a Gaussian kernel with the binary spike train data.

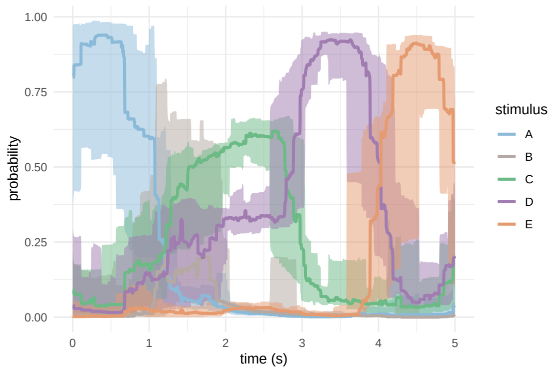
One of the significant benefits of the proposed decoding model is its ability to incorporate the autocorrelation of spikes over time. This property is crucial since it allows for more accurate decoding of the underlying intensity functions that drive the spike train activity. As demonstrated in Figure 5.15, the probabilities of stimuli over time appear as a smooth curve when decoded using the proposed method, compared to the noisy probabilities generated by the MLR, MLR + LASSO, and RF models. This difference in the smoothness of the decoded probabilities is a clear indication of the advantage of incorporating the temporal structure of the spike trains. Additionally, since the hippocampus is known to generate spike trains with temporal dependencies, this property makes the



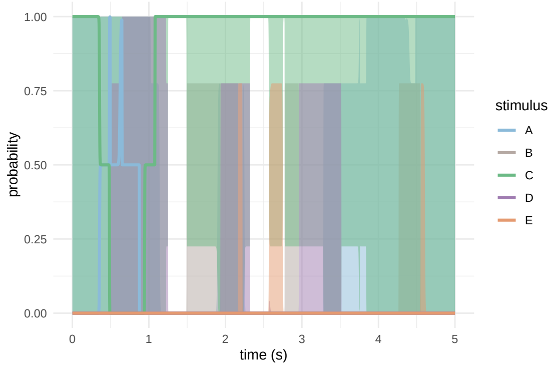
(a) proposed decoding model



(b) MLR + LASSO



(c) RF



(d) MLR

Figure 5.15: Comparing the time series of the decoding/classification probabilities of our proposed decoding model to the MLR, MLR + LASSO, and RF classifiers. The decoding model has the advantage of incorporating autocorrelation of the spikes over time, which results in a smooth curve of probabilities for the stimuli compared to the noisy probabilities generated by other models such as MLR, MLR + LASSO, and RF models.

decoding model particularly well-suited for decoding memory replay events in the hippocampus.

5.4.2 Real Data Experiments

The data set [58] contains recordings of neural activity from the dorsal CA1 region of the hippocampus as rats performed a complex sequence memory task. Briefly, the task involves presenting rats with a repeated sequence of non-spatial events (five stimuli: odors A, B, C, D, and E) at a single port, and requiring them to identify each stimulus as "in sequence" (InSeq; i.e. AB$\underline{C}$...) or "out of sequence" (OutSeq; e.g., AB$\underline{D}$...). Spiking and local field potential (LFP) activity have been recorded using a chronically-implanted hyperdrive with approximately 20 tetrodes. On average, about 40 of the recorded neurons (per rat) are active throughout the experiments.

We focus on the trials of odors A, B, C, D, and E in which the odor is presented InSeq and is correctly identified as such by the animal, and a *complete sequence* refers to a set of five consecutive trials of this type. For each of these trials, the spiking activity are extracted during the time window of $(0, 1.2)$ seconds after the rat's nose enters the port. We concatenate the spiking activity of the individual odors/trials in a complete sequence such that we get a spike train that lasts for $T = 5 \times 1.2 = 6$ seconds, corresponding to 5 odors each lasting for 1.2 seconds. Therefore, a complete sequence is a binary spike train data of multiple neurons firing over a time period of $T = 6$ seconds. In total there are 11 complete sequences of odors A, B, C, D, and E in the dataset. The fitted intensity functions, using a leave-one-out procedure, as well as the posterior distributions of the β parameters of the encoding model are shown in Figures 5.16 and 5.17, respectively. Figure 5.16 provide insight into the firing behavior of each neuron and the uncertainty associated with the intensity function estimates. We can see that the intensity function of each neuron closely matches the spiking activity that is shown below each sub-plot. Figure 5.17 shows that spike-and-slab prior shrinks the irrelevant parameters to zero. It is worth noting the strong relationship between Figures 5.16 and 5.17. Whenever a neuron has an almost flat intensity function, the corresponding β parameters are mostly shrunk to zero. Conversely, neurons with strong firing patterns have non-zero β parameters. This relationship reinforces the importance of selecting neurons based on their firing rates.

It is worth noting that there are longer gaps between the onset of the trials in the dataset, which are greater than 1.2 seconds as seen in Figure 5.18. When we concatenate these trials to each other, we essentially omit a lot of the inter-trial spikes. Inter-trial spikes have been demonstrated to have minimal information that is directly related to the applied stimuli during the experiment [58]. Therefore, the spiking activity of a complete sequence described above is not a continuous representation of the actual dataset. Instead, a complete sequence is composed of different chunks

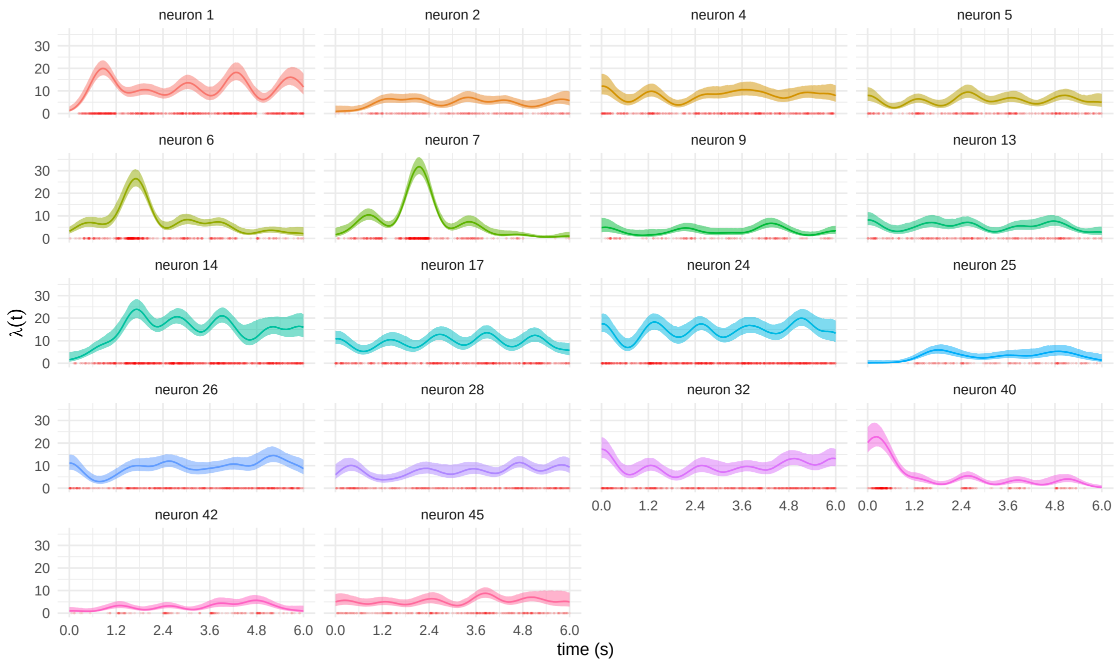


Figure 5.16: Posterior distribution of the intensity function of each neuron in the real data. The solid lines represent the point-wise posterior medians, and the colored regions correspond to the point-wise 95% posterior credible intervals. The spikes are shown with red dots on the x-axis of each plot. The x-axis cover the time interval from 0 to 6 seconds, which corresponds to $5 \times 1.2 = 6$ seconds. That is, the interval $(0, 1.2)$ corresponds to odor A, the interval $(1.2, 2.4)$ corresponds to odor B, etc.

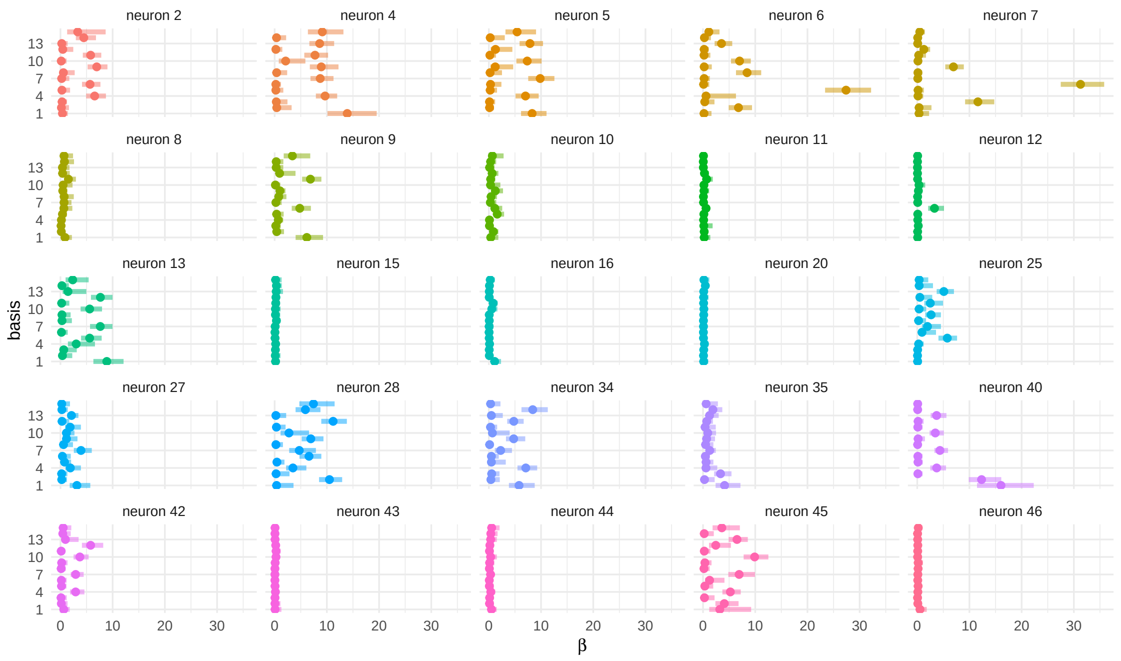


Figure 5.17: Posterior distribution of the β parameters from the encoding model in the real data. The dots correspond to the posterior medians and the horizontal bars correspond to the point-wise 95% posterior credible intervals.

of the data (1.2 seconds each) placed one after the other. We adopt this approach for two reasons. First, the entire dataset is quite large (≈ 3 million rows), and analyzing it in its entirety is computationally infeasible. Second, the hippocampal activity related to odor processing in the rat's brain is restricted to the time interval of odor presentation [58]. Therefore, we focus our analysis on this interval to capture the relevant neural activity.

In order to select the most relevant neurons for our analysis, we computed the firing rates of all the neurons in the dataset over the entire data set and selected those with a firing rate greater than 10^{-3} spikes per second. This resulted in selecting 18 out of the 46 neurons in the dataset, as seen in Figure 5.18. To determine the optimal number of radial basis function (RBF) basis functions for the encoding model, we tested a range of possible values, $(5, 10, 15, \ldots, 40)$, using the deviance information criterion (DIC) metric, ultimately selecting $B = 15$. To verify that the encoding model fit the dataset well, we employed the time-rescaling theorem in Section 5.4.1 and found that all 18 neurons had p-values above 0.09 and an average of 0.276, indicating a good fit.

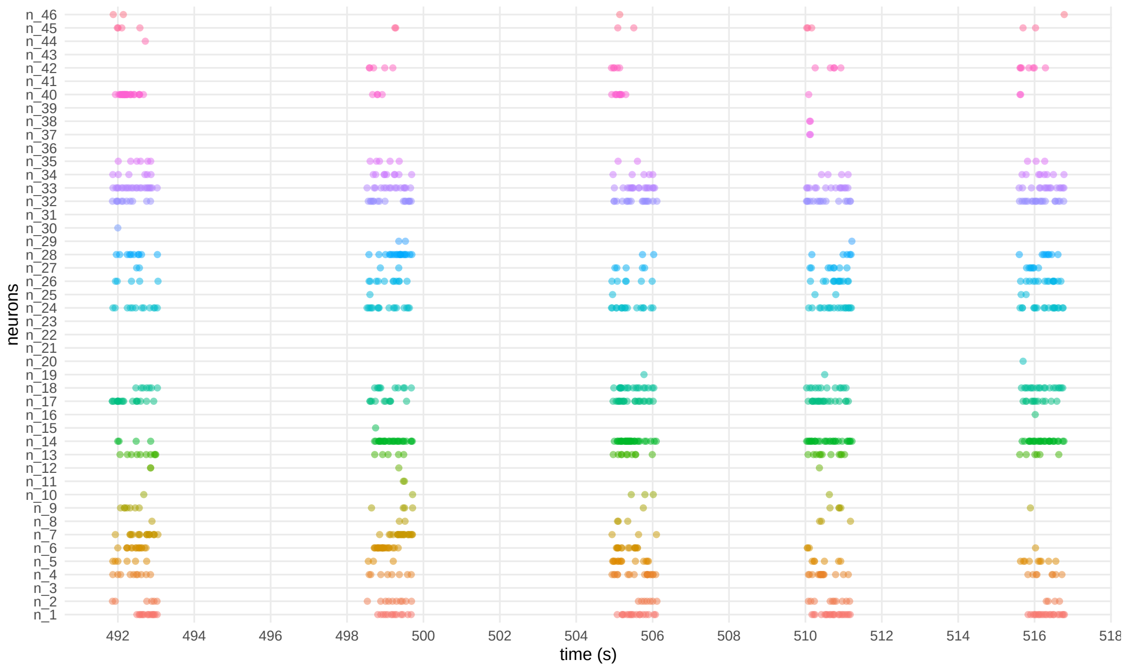
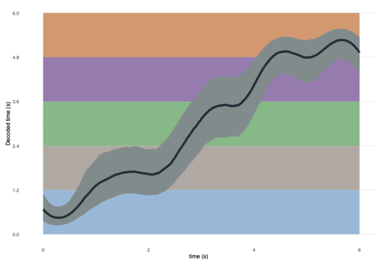


Figure 5.18: Spiking activity for the first complete sequence of InSeq + correct trials of odors A, B, C, D, and E. The y-axis shows 46 neurons that fire over the time interval $(492, 518)$ seconds. Each dot represents a spike. The separation in time between the onset of each of the trials is shown by the gaps in the figure. Some neurons do not fire at all during this sequence.

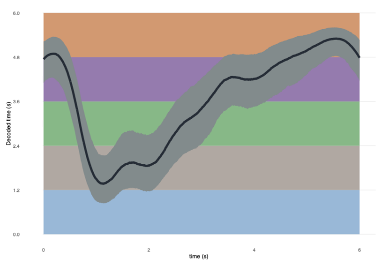
Next, we applied the decoding model to predict the latent time for each of the 11 sequences, and the results are depicted in Figure 5.19. The sub-plots show an overall upward trend for the decoded time, which corresponds to the progression of the odors A, B, C, D, and E over the complete sequence. The ground truth for these sequences is known, and we calculated the accuracy of the decoding results by dividing the x-axis into 50 bins and determining whether the decoded time falls within the correct region on the y-axis. The decoding accuracies for each sub-plot are displayed below them in Figure 5.19.

5.5 Discussion

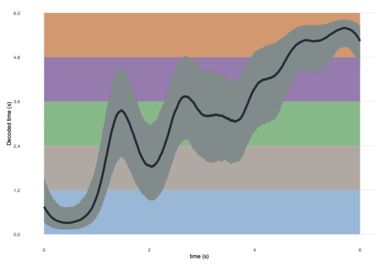
In this paper, we have proposed a flexible multivariate stochastic process model that can capture the underlying dynamics of point processes such as neuron spiking activities. Our proposed approach



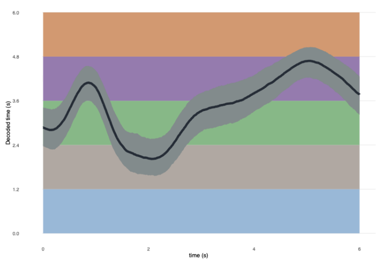
(a) Sequence 1: 80% accuracy



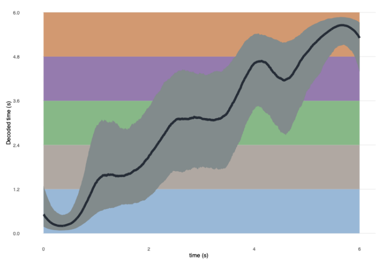
(b) Sequence 2: 70% accuracy



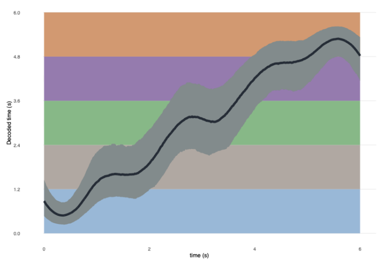
(c) Sequence 3: 78% accuracy



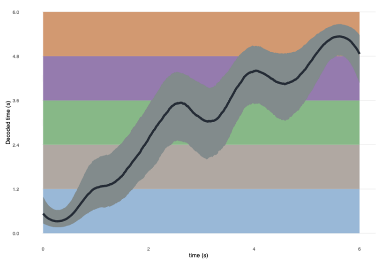
(d) Sequence 4: 54% accuracy



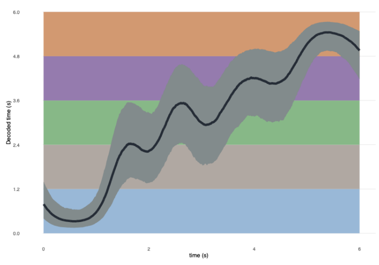
(e) Sequence 5: 90% accuracy



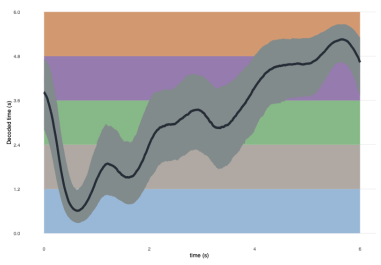
(f) Sequence 6: 88% accuracy



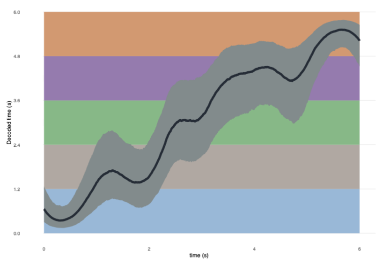
(g) Sequence 7: 80% accuracy



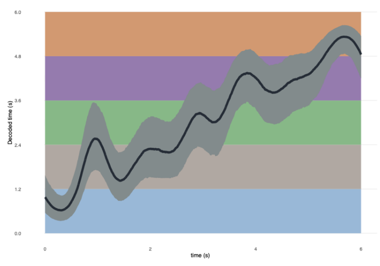
(h) Sequence 8: 94% accuracy



(i) Sequence 9: 72% accuracy



(j) Sequence 10: 82% accuracy



(k) Sequence 11: 80% accuracy

Figure 5.19: Decoded latent time for the 11 complete sequences. The black curves represent the posterior pointwise median of the decoded time, and the gray regions around these curves are the posterior 95% pointwise credible intervals, representing the uncertainty of the decoded time. The y-axis in the plots represents the decoded time that spans between 0 and $T = 5 \times 1.2 = 6$ seconds. The x-axis, which spans from 0 to 6 seconds, is the duration of a complete sequence of odors ABCDE. The decoding accuracies of each complete sequence is shown in the caption of each subplot. The background of each subplot contains horizontal stripes to indicate the stimuli: A: 🟦 B: 🟫 C: 🟩 D: 🟪 E: 🟧

models the firing rates of neurons with a non-homogeneous Poisson process, where the firing rate function is represented as a linear combination of radial basis functions. The parameters of the model are estimated using a Bayesian framework, allowing us to incorporate prior knowledge and quantify uncertainty in the parameter estimates. However, our proposed methods have their limitations, such as the fixed hyperparameters for the RBF centers and width in the encoding model, which could be improved by treating them as parameters to be learned by the model. A possible solution is to generate B RBF kernel centers $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_B)^T$ that are ordered and constrained within the interval $(0, T)$, we start by generating a $B+1$-simplex , denoted by $\boldsymbol{\zeta} \in \Delta^{B+1}$, from a Dirichlet distribution with parameter $\boldsymbol{\alpha}$. Then, the elements of the $\boldsymbol{\mu}$ vector are as follows:

$$\mu_b = \tilde{T} \sum_{i=1}^{B} \zeta_i \quad \text{for} \quad b = 1, \ldots, B \tag{5.34}$$

The $\boldsymbol{\alpha}$ parameter is set as follows:

$$\boldsymbol{\alpha} = \alpha_c \tilde{T} \times (0.6, \underbrace{1.2, \ldots, 1.2}_{B-1 \text{ times}}, 0.6) \tag{5.35}$$

where α_c is a scaling factor (See Figure 5.20).

Furthermore, in our current models, we do not account for the effect of the history of the process on the firing rates of the neurons. In future research, we could explore the use of history-dependent point processes such as Hawkes processes to incorporate this feature. Additionally, we could investigate the use of other basis functions or methods to increase the flexibility of our model without significantly increasing its computational complexity, such as B-splines or smoothing splines.

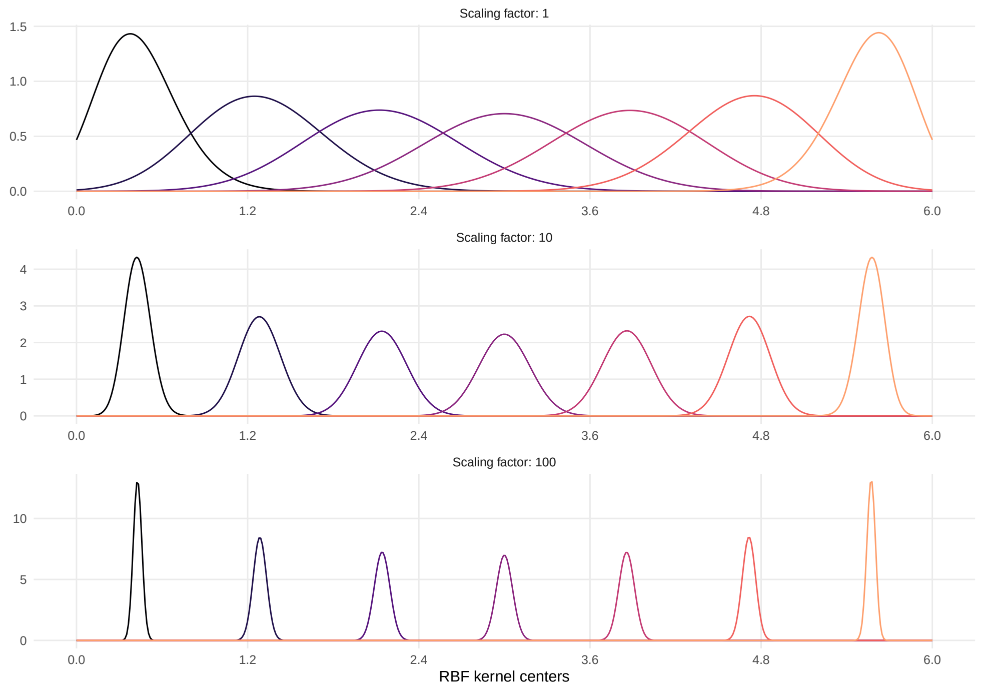


Figure 5.20: The distribution of the centers of the RBF kernels using the $B + 1$-simplex method. The scaling factor α_c controls the prior uncertainty of the RBF centers.

Chapter 6

Future Work

In this work, we explored the modeling and decoding of neural activity patterns using stochastic point processes. We delved into the concept of multivariate point processes and discussed how they provide a suitable framework for studying neuronal spiking activity, considering the inherent nature of such activity as point processes themselves. In Chapter 3, we examined different Hawkes process models and their applications in understanding neural dynamics. We highlighted the strengths of our proposed Hawkes process models in successfully capturing the firing rates of neurons and estimating their intensity functions. In Chapter 4, we extended the Hawkes process models to incorporate inhibitory effects in the spiking activity of an ensemble of neurons. We demonstrated that our SMHP model achieves comparable performance to a fully connected non-linear multivariate Hawkes process model while significantly reducing computational time. By incorporating inhibitory interactions, our model offers a more comprehensive representation of neural dynamics observed in the spiking activity of hippocampal neurons.

Our proposed methodologies in Chapter 3 have shown significant promise and offer potential for improvement to overcome their limitations and enhance their capabilities. One key aspect to consider is the utilization of alternative scalable inference methods to enhance the accuracy and efficiency of the modeling process. While the results presented in this paper relied on variational inference, which is an approximate method, exploring scalable Markov chain Monte Carlo (MCMC) methods such as Hamiltonian Monte Carlo (HMC) and Variational HMC can yield more accurate and reliable results. Another direction for improvement is to incorporate time-varying base intensity functions in the models presented in Chapter 3. Currently, the base intensity of each neuron remains constant during the observation interval. By introducing time-varying functions such as splines or Gaussian processes, we can capture the dynamic nature of neuronal activity more accurately and account for temporal variations in firing rates. Furthermore, drawing inspiration from convolutional neural networks (CNNs), we can explore the use of multiple filters to capture the influence of an ensemble of neurons in diverse ways. This approach can provide a more comprehensive understanding of neural activity patterns and improve the model's ability to decode underlying stimulus patterns. Furthermore, future extensions of our model can focus on integrating multiple data modalities, such as spike trains and local field potentials (LFPs). By incorporating LFPs as marks in a Marked Multivariate Hawkes Process (MHP), where the LFP values during each neuron's spike act as additional information influencing the conditional intensity function (CIF), we can gain a richer representation of neural dynamics and uncover new insights.

The SMHP model in Chapter 4 has certain limitations that should be acknowledged. Its assumption of a self-modulating mechanism may not accurately capture complex interactions between neurons in certain scenarios. Moreover, the generalizability of the model beyond decoding odor stimuli needs to be investigated. Furthermore, the performance of the SMHP model may vary when applied to different neural datasets or experimental paradigms, necessitating comprehensive assessment and validation across various domains. Despite these limitations, the interpretability of the model parameters and their potential to reveal meaningful insights about neural dynamics provide a promising avenue for further understanding the intricate dynamics of the brain. The SMHP model represents a novel approach in the field of neural decoding and opens up new avenues for future research in unraveling the temporal organization of neural activity and its relationship to behavior and cognition.

Improving the interpretability and adaptability of the model is also essential in Chapter 5. Treating the fixed hyperparameters for RBF centers and width as learnable parameters can enhance their effectiveness in capturing underlying patterns. The generation of RBF kernel centers using a simplex method (described in Section 5.5), controlled by a scaling factor, can provide a more flexible and adaptable framework for capturing the firing rates of the neurons.

Bibliography

[1] T. A. Allen and N. J. Fortin. The evolution of episodic memory. *Proceedings of the National Academy of Sciences*, 110(Supplement 2):10379–10386, 2013.

[2] I. Apostolopoulou, S. Linderman, K. Miller, and A. Dubrawski. Mutually regressive point processes. *Advances in Neural Information Processing Systems*, 32, 2019.

[3] C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[4] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[5] B. Bócsi, D. Nguyen-Tuong, L. Csató, B. Schoelkopf, and J. Peters. Learning inverse kinematics with structured prediction. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 698–703. IEEE, 2011.

[6] A. Bonnet, M. M. Herrera, and M. Sangnier. Maximum likelihood estimation for hawkes processes with self-excitation or inhibition. *Statistics & Probability Letters*, 179:109214, 2021.

[7] P. Brémaud and L. Massoulié. Stability of nonlinear hawkes processes. *The Annals of Probability*, pages 1563–1588, 1996.

[8] E. N. Brown, R. Barbieri, V. Ventura, R. E. Kass, and L. M. Frank. The time-rescaling theorem and its application to neural spike train data analysis. *Neural computation*, 14(2):325–346, 2002.

[9] E. N. Brown, L. M. Frank, D. Tang, M. C. Quirk, and M. A. Wilson. A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *Journal of Neuroscience*, 18(18):7411–7425, 1998.

[10] R. L. Burden, J. D. Faires, and A. M. Burden. *Numerical analysis*. Cengage learning, 2015.

[11] G. Buzsáki. Neural syntax: cell assemblies, synapsembles, and readers. *Neuron*, 68(3):362–385, 2010.

[12] G. Buzsáki and D. Tingley. Space and time: the hippocampus as a sequence generator. *Trends in cognitive sciences*, 22(10):853–869, 2018.

[13] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1), 2017.

[14] Y. Chen. Likelihood function for multivariate hawkes processes. *Preprint (5 pages) available on https://www. math. fsu. edu/ychen/research/HawkesLikelihood. pdf*, 2016.

[15] Y. Chen. Thinning algorithms for simulating point processes. *Florida State University, Tallahassee, FL*, 2016.

[16] D. Clewett, S. DuBrow, and L. Davachi. Transcending time in the brain: How event memories are constructed from experience. *Hippocampus*, 29(3):162–183, 2019.

[17] J. P. Cunningham and B. M. Yu. Dimensionality reduction for large-scale neural recordings. *Nature neuroscience*, 17(11):1500–1509, 2014.

[18] D. J. Daley, D. Vere-Jones, et al. *An introduction to the theory of point processes: volume I: elementary theory and methods*. Springer, 2003.

[19] H. Eichenbaum. Time cells in the hippocampus: a new dimension for mapping memories. *Nature Reviews Neuroscience*, 15(11):732–744, 2014.

[20] M. Eichler, R. Dahlhaus, and J. Dueck. Graphical modeling for multivariate hawkes processes with nonparametric link functions. *Journal of Time Series Analysis*, 38(2):225–242, 2017.

[21] T. F. Freund and G. Buzsáki. Interneurons of the hippocampus. *Hippocampus*, 6(4):347–470, 1996.

[22] A. E. Gelfand and A. F. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409, 1990.

[23] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. CRC press, 2013.

[24] J. P. Gonzalez, S. E. Cook, T. Oberthür, A. Jarvis, J. A. Bagnell, and M. B. Dias. Creating low-cost soil maps for tropical agriculture using gaussian processes. 2007.

[25] J. Gonzalvez, E. Lezmi, T. Roncalli, and J. Xu. Financial applications of gaussian processes and bayesian optimization. *arXiv preprint arXiv:1903.04841*, 2019.

[26] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97—-109, 1970.

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

[28] A. G. Hawkes. Hawkes processes and their applications to finance: a review. *Quantitative Finance*, 18(2):193–198, 2018.

[29] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[30] R. E. Kass, U. T. Eden, E. N. Brown, et al. *Analysis of neural data*, volume 491. Springer, 2014.

[31] F. Kloosterman, S. P. Layton, Z. Chen, and M. A. Wilson. Bayesian decoding using unsorted spikes in the rat hippocampus. *Journal of neurophysiology*, 2014.

[32] A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic differentiation variational inference. *arXiv preprint arXiv:1603.00788*, 2016.

[33] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[34] P. W. Lewis and G. S. Shedler. Simulation of nonhomogeneous poisson processes by thinning. *Naval research logistics quarterly*, 26(3):403–413, 1979.

[35] C. J. MacDonald, K. Q. Lepage, U. T. Eden, and H. Eichenbaum. Hippocampal "time cells" bridge the gap in memory for discontiguous events. *Neuron*, 71(4):737–749, 2011.

[36] J. R. Manns, M. W. Howard, and H. Eichenbaum. Gradual changes in hippocampal activity support remembering the order of events. *Neuron*, 56(3):530–540, 2007.

[37] H. Mei and J. M. Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems*, 30, 2017.

[38] P. Miller. *An introductory course in computational neuroscience*. MIT Press, 2018.

[39] T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the american statistical association*, 83(404):1023–1032, 1988.

[40] G. O. Mohler, M. B. Short, P. J. Brantingham, F. P. Schoenberg, and G. E. Tita. Self-exciting point process modeling of crime. *Journal of the american statistical association*, 106(493):100–108, 2011.

[41] R. M. Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[42] Y. Ogata. On lewis' simulation method for point processes. *IEEE transactions on information theory*, 27(1):23–31, 1981.

[43] Y. Ogata. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical association*, 83(401):9–27, 1988.

[44] S. W. Oh, J. A. Harris, L. Ng, B. Winslow, N. Cain, S. Mihalas, Q. Wang, C. Lau, L. Kuan, A. M. Henry, et al. A mesoscale connectome of the mouse brain. *Nature*, 508(7495):207–214, 2014.

[45] J. Olinde and M. B. Short. A self-limiting hawkes process: Interpretation, estimation, and use in crime modeling. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 3212–3219. IEEE, 2020.

[46] E. Pastalkova, V. Itskov, A. Amarasingham, and G. Buzsaki. Internally generated cell assembly sequences in the rat hippocampus. *Science*, 321(5894):1322–1327, 2008.

[47] B. E. Pfeiffer. The content of hippocampal "replay". *Hippocampus*, 30(1):6–18, 2020.

[48] J. Piironen and A. Vehtari. Sparsity information and regularization in the horseshoe and other shrinkage priors. *Electronic Journal of Statistics*, 11(2):5018–5051, 2017.

[49] P. Poduval, A. Zakeri, F. Imani, H. Alimohamadi, and M. Imani. Graphd: Graph-based hyperdimensional memorization for brain-like cognitive learning. *Frontiers in Neuroscience*, page 5, 2022.

[50] N. G. Polson and J. G. Scott. On the half-cauchy prior for a global scale parameter. *Bayesian Analysis*, 7(4):887—-902, 2012.

[51] C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2005.

[52] A. D. Redish. Vicarious trial and error. *Nature Reviews Neuroscience*, 17(3):147–159, 2016.

[53] P. Reynaud-Bouret, V. Rivoirard, and C. Tuleau-Malot. Inference of functional connectivity in neurosciences via hawkes processes. In *2013 IEEE global conference on signal and information processing*, pages 317–320. IEEE, 2013.

[54] G. Riutort-Mayol, P.-C. Bürkner, M. R. Andersen, A. Solin, and A. Vehtari. Practical hilbert space approximate bayesian gaussian processes for probabilistic programming. *Statistics and Computing*, 33(1):17, 2023.

[55] M.-A. Rizoiu, Y. Lee, S. Mishra, and L. Xie. Hawkes processes for events in social media. In *Frontiers of multimedia research*, pages 191–218. 2017.

[56] C. P. Robert, G. Casella, and G. Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.

[57] D. L. Schacter, D. R. Addis, D. Hassabis, V. C. Martin, R. N. Spreng, and K. K. Szpunar. The future of memory: remembering, imagining, and the brain. *Neuron*, 76(4):677–694, 2012.

[58] B. Shahbaba, L. Li, F. Agostinelli, M. Saraf, K. W. Cooper, D. Haghverdian, G. A. Elias, P. Baldi, and N. J. Fortin. Hippocampal ensembles represent sequential relationships among an extended sequence of nonspatial events. *Nature communications*, 13(1):1–17, 2022.

[59] A. Solin and S. Särkkä. Hilbert space methods for reduced-rank gaussian process regression. *Statistics and Computing*, 30(2):419–446, 2020.

[60] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[61] W. Truccolo, U. T. Eden, M. R. Fellows, J. P. Donoghue, and E. N. Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of neurophysiology*, 93(2):1074–1089, 2005.

[62] N. Van Strien, N. Cappaert, and M. Witter. The anatomy of memory: an interactive overview of the parahippocampal–hippocampal network. *Nature reviews neuroscience*, 10(4):272–282, 2009.

[63] C. Zhang, B. Shahbaba, and H. Zhao. Variational hamiltonian monte carlo via score matching. *Bayesian Analysis*, 13(2):485–506, 2018.