Model-based clustering of non-Poisson, non-homogeneous, point process events, with application to neuroscience

PhD Thesis Proposal

Sonia Todorova
Department of Statistics, Carnegie Mellon University

Thesis Committee

Valérie Ventura
Steven Chase
Cosma Schalizi
Rob Kass

Draft Version:

August 31, 2011

Abstract

Neurons communicate by propagating electric pulses called "spikes". Experimental neuroscientists analyze the point process of spike times ("spike train") to study how neural activity relates to behavior. In experiments with behaving subjects, each electrode in the brain receives the combined signal of multiple neurons. Before we can analyze such data, we need to process the signal to identify the spikes of individual neurons, a procedure known as "spike-sorting". The traditional practice is to perform spike-sorting by clustering the characteristic spike voltage waveforms and then to estimate neuron properties, e.g. firing rates as functions of behavioral covariates, from the sorted data. This approach ignores the information available for spike-sorting in the neurons' firing rates and in dependencies between neurons. As a consequence, the traditional procedure yields inconsistent estimates of firing rates and correlations between neurons (Ventura, 2009; Ventura and Gerkin, 2011). The goal of this thesis is to design a spike-sorting method which incorporates in a principled way all variables that contain information about spike identities. This new method would allow us to obtain consistent estimates of neuron parameters such as firing rates and correlations. This requires defining a model for spike identities and designing an algorithm to estimate the model parameters from data.

Contents

1 Introduction

1.1 Neuroscience Background

The topic of this thesis originates in experimental neuroscience. Learning about how the brain works involves studying the relationship between neural activity and behavior. This relationship provides insight into the *neural code*, the way the brain receives, processes and transmits information about the environment and the body. The brain is composed of billions of interconnected neurons. They communicate through electric signals. Neurons use sharp pulses of electric potential called *spikes* to transmit information along the cell body to other neurons in their network. We can measure spikes by inserting an electrode in the brain. Figure 1 shows an illustration of the voltage signal of a neuron. An electrode inside the neuron allows us to measure the electric potential over time. A sharp waveform in the voltage trace (shown in red) indicates a spike. The point process of spike times, known as *spike train*, is a way to record the neuron message. Neuroscientists analyze observed spike trains to study the neural code.

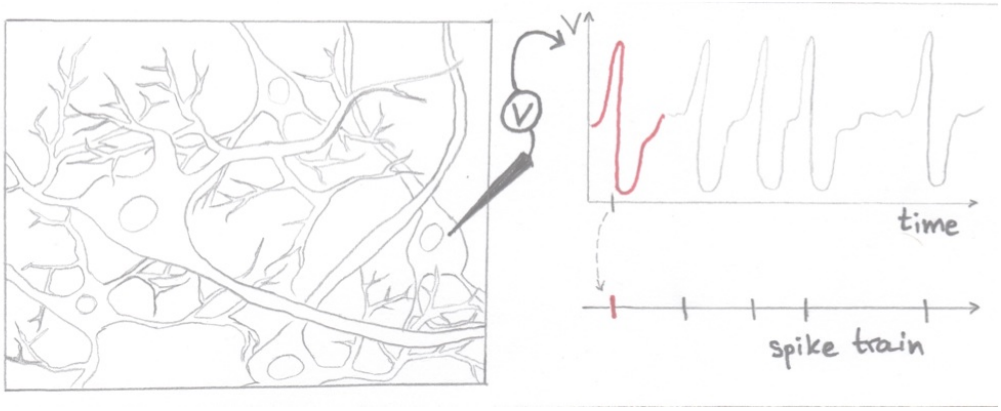

Figure 1: Measuring the electric signal from a neuron. A single spike (highlighted in red) is a waveform on the voltage trace corresponding to an event in the spike train.

One element of the neural code is the frequency of spikes as a function of behavioral covariates, i.e. the *tuning curve* of a neuron. Some neurons produce spikes at elevated rates for specific values of the stimulus. For example, some neurons in the motor cortex fire much more rapidly when the movement is in their preferred direction versus any other direction (Georgopoulos, 1982). We say that these neurons are *tuned* to the direction of movement. To illustrate how we can measure this tuning relationship consider the following hypothetical experiment. Suppose we have a subject from a species with well-studied brain anatomy. We have determined that a "press button" region of the brain is responsible for pressing mechanical buttons. We ask the subject to press a button repeatedly with varying intensity while we record the neural activity in the "press button" region of her brain with an electrode (Figure 2). The pressure on the button takes on three different

values (Figure 2b). For each value of pressure we calculate the observed firing rate and plot firing rate versus pressure (Figure 2c). This "press" neuron example is made-up but the general experimental principle of estimating the tuning relationship is faithful to real practice. Figure 3 shows a tuning curve from the Georgopoulos (1982) experiment. In this experiment a monkey moves his arm from the center of a circle to eight points on the circle. The tuning curve is based on the firing rates of an arm-related neuron observed over five repetitions of the experiment. The observations of the tuning relationship for the eight directions are used to fit a smooth tuning curve. We can test our estimate of the tuning relationship by using it to estimate the covariate (e.g. direction of motion) from the electric signals in the subject's brain. Predicting the value of behavioral covariates from neural activity is known as *decoding*. A successful method for decoding makes it possible to design a brain-machine interface which translates commands from the brain to a robotic arm (Donoghue, 2002). The use of brain-machine interfaces for prosthetic devices requires accurate and efficient decoding algorithms which can produce real-time output.

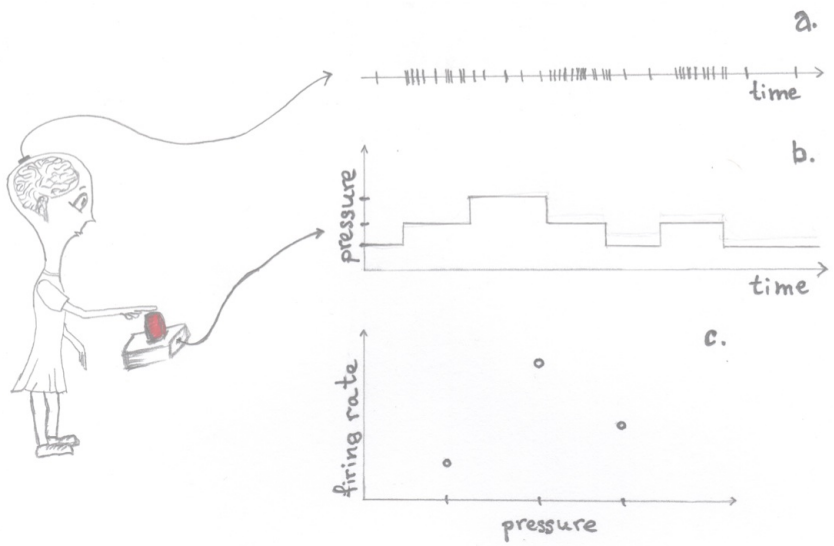

Figure 2: Tuning in a hypothetical experiment. A subject presses a button while we record from the "press button" region of her brain. (a) Recorded spike train of one "press" neuron. (b) Recorded pressure on the button over time. (c) Observed tuning curve for this neuron.

The tuning of individual neurons represents a *rate code*. It is possible that the language of the brain also contains a more intricate *temporal code*, namely that the exact spike pattern of an ensemble of neurons carries information (Harvey et al, 2009; Buzsàki, 2004). An important feature of a *temporal code* is the correlation between spike trains of different neurons. Accounting for correlations enhances the understanding of the neural code and can improve the quality of decoding (Pillow et al, 2008).

The study of the neural code relies on analyzing the spike trains of an ensemble of

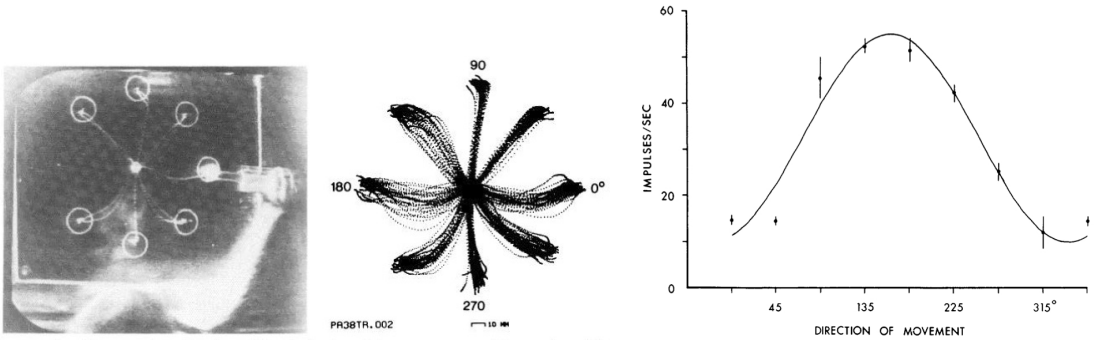

Figure 3: Georgopoulos (1982) experiment. Monkey moves from the center of a circle to eight points on the circle (left). The kinematic covariate here is the position of the arm identified by its trajectory on the screen (middle). (Right) spike impulses per second versus direction of movement for five repetitions of the experiment and fitted sinusoidal tuning curve.

neurons. We can measure the electric activity of a neuron by inserting an electrode inside the cell body. This provides a high-quality signal but eventually kills the cell. The established practice in experiments with behaving subjects is to use extracellular electrodes instead which does not kill cells. Often in the form of arrays, such electrodes can record the activity of hundreds of neurons simultaneously, making it possible to study how the activity of an ensemble of neurons relates to behavior. When placed outside the cell, an electrode records the combined signal from multiple neurons. Before we can analyze the recordings we need to isolate the spike trains of single neurons. This processing step is called *spike-sorting*.

1.2 Spike-sorting

Spike-sorting is the procedure of extracting the spike trains of individual neurons from the signal recorded on an extracellular electrode. This requires determining how many neurons are recorded on the electrode and which neuron produced each spike. We call this source neuron the *identity* of a spike. Figure 4a shows a sample voltage signal recorded on an extracellular electrode. The waveforms of some spikes have very similar shapes. When we overlay all waveforms (Figure 4b) we see two distinct characteristic shapes. This is not surprising because neurons produce approximately the same waveform every time they spike. The traditional method for spike-sorting uses this fact and consists of clustering the recorded waveforms. The identity of each spike corresponds to the cluster to which its waveform belongs. This is illustrated in Figure 4b-d. The observed waveforms can be clustered into two groups by characteristic shape (Figure 4b). This clustering determines the identity of each spike in the electrode spike train (Figure 4c) and provides the spike trains of the two neurons recorded on the electrode (Figure 4d). After spike-sorting the resulting neuron spike trains can be analyzed to estimate tuning curves, correlations, and other relationships of interest. Section 2.1 gives more detail about traditional spike-sorting.

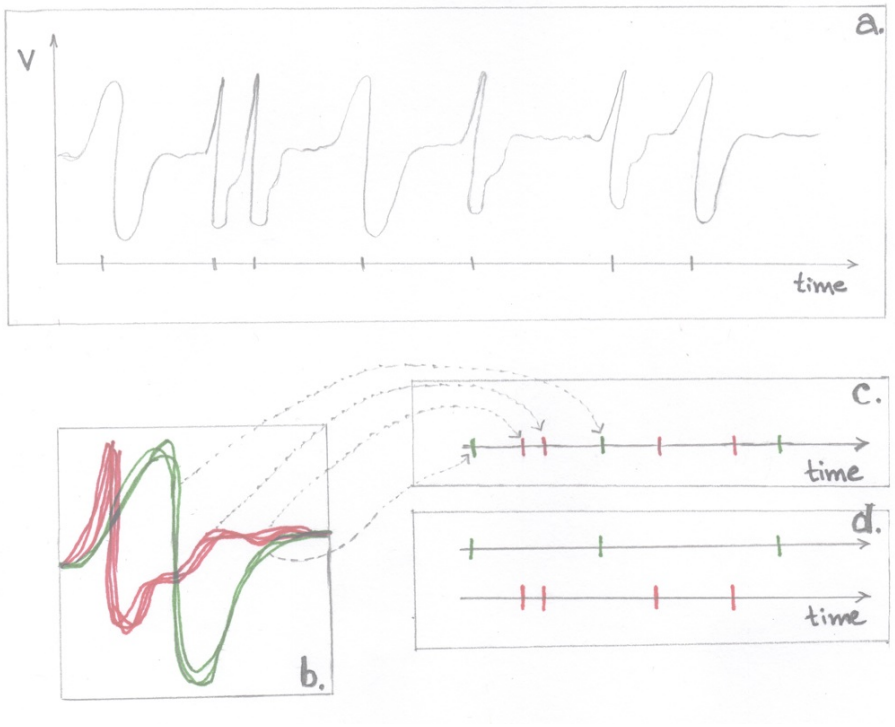

Figure 4: Traditional spike-sorting consists of clustering the recorded waveforms. (a) Voltage trace on an extracellular electrode. The time of each spike is marked on the x-axis denoting the electrode spike train. (b) Overlaid waveforms form two groups by shape (one group highlighted in red and the other in green) (c) Electrode spike train with spike identities corresponding to the clustering in (a). (d) Spike trains of the two neurons.

1.3 Spike-sorting assumptions

This method relies on implicit assumptions. First, it assumes that a neuron's waveforms follow the same distribution throughout the course of the experiment. However, when neurons spike at high frequency for a prolonged period of time, the amplitude of the waveforms decreases. This phenomenon is called attenuation. Thus, assuming that the waveform distribution is constant regardless of past activity is a reasonable simplification only when firing rates are low. Other factors may contribute to changes in the waveform distribution as a function of time such as drifting of electrodes (Calabrese and Paninski, 2011).

Second, the traditional method sorts spikes one at a time, which assumes that spike identities are independent, and in particular that the identity of a spike does not depend on the identities of the spikes recorded prior to it, i.e. the history of the spike train. This is equivalent to assuming that neuron spike trains are Poisson processes. However, neurons' refractory periods prevent them from firing immediately after a spike, which contradicts the Poisson assumption. Another implication of sorting each spike independently is that neuron spike trains are assumed to be independent of each other. This conflicts with the

belief that correlations betweens spike trains are a part of the neural code: we not only know that some spike trains are correlated, but often the very goal of the experiment is to study these correlations. Thus, it is not principled to ignore them for spike-sorting.

1.4 Improvements on traditional spike-sorting

Traditional spike-sorting uses only waveform information, ignoring any available behavioral covariates. Ventura (2009a) shows that the estimates of tuning curves based on data sorted in this way are biased. This is because tuning provides information about the spike identities. Consider a simple example: suppose a neuron record spikes from two neurons tuned to the direction of arm movement. Neuron A spikes mostly when the arm moves to the left and neuron B spikes mostly when the arm moves to the right. Suppose the waveforms of the two neurons form two clusters but 5% of the spikes are misclassified. The misclassified spikes from neuron A are the ones with waveforms most closely resembling waveforms from neuron B. Since the classification method does not depend on the covariate (direction of motion) most misclassified spikes from neuron A occur when the neuron spikes the most, i.e. when the arm moves to the right. Therefore, spike-sorting based on waveform information only results in bias in the estimated tuning relationship and is expected to reduce the quality of decoding. Ventura (2009a) proves this result formally and demonstrates it in a simulation. Ventura (2009b) introduces an algorithm which removes the bias by using tuning information for spike-sorting. This algorithm is described in Section 2.2. The effect of biased tuning curves in real applications is yet to be explored. This is part of my thesis. See Section 3.1 for more detail.

Flaws in spike-sorting can result in biased estimates of correlation (Cohen and Kohn, 2011). Ventura and Gerkin (2011) prove that this is a result of ignoring the correlation between spike trains when spike-sorting. They propose ensemble sorting and prove that it removes bias in subsequent analyses. They illustrate the method for a directed two-neuron model, deriving the specific spike-sorting rule. For more detail see Section 2.3. Part of my research consists of deriving the spike-sorting rule for a general ensemble of neurons and developing an algorithm to fit the model parameters to data by maximum likelihood.

The implementation of a general algorithm to estimate the parameters of the correlation model from data is part of my research goals (see Section 3.2).

We propose to develop a model for spike-sorting which incorporates in a principled way all variables that contain information about spike identities. This would require an extension of the Ventura (2009b) algorithm. The complete method will include waveform information, tuning information, and spiking history of multiple spike trains. Section 2 reviews relevant methodology from the literature. Section 3 presents preliminary results and Section 4 details the steps required to fully develop the desired methodology.

2 Literature

Here we review the statistical framework for spike-sorting.

We start with some notation.

An electrode records n spikes over the course of an experiment between time $t = 0$ and $t = T$. Denote the recorded waveforms by the vector WF.

$$WF = (WF_1, WF_2, ..., WF_n)$$

Let ST be the vector of spike times.

$$ST = (ST_1, ST_2, ..., ST_n)$$

Let C denote the behavioral covariates. We record

$$C(t), t \in (0, T)$$

Let X denote the vector of unobserved spike identities. Let K be the number of neurons recorded on the electrode. Note that K is also unknown.

$$X = (X_1, X_2, ..., X_n); \quad \forall j, \quad X_j \in \{1, 2, ..., K\},$$

$X_j = k$ if spike j (with waveform WF_j, recorded at time ST_j) was produced by neuron k.

The goal of spike-sorting is to determine the spike identities given the observed data. We take a statistical approach and determine the posterior probability

$$P(X_j = k|WF, ST, C), \ \forall j, k.$$

The table below summarizes the spike-sorting methods discussed in previous sections and the goals of this thesis.

Spike-sorting method	*Modeled quantity*	*Section*
Traditional spike-sorting	$P(X_j = k\|WF_j)$	2.1
Waveform and tuning (Ventura, 2009b)	$P(X_j = k\|WF_j, C)$	2.2
Waveform and correlation (Ventura and Gerkin, 2011)	$P(X_j = k\|WF, ST)$	2.3
Goal of this thesis	$P(X_j = k\|WF, ST, C)$	3.2, 4

2.1 Traditional spike-sorting method consists of clustering the recorded waveforms

For a review of spike-sorting see Lewicki (1998), Sahani (1999), and Pouzat (2005).

Traditional model-based spike-sorting determines the identity of each spike based only on its waveform measurement $P(X_j = k|WF_j)$. This implies that $X_j|WF_j$, for $j = 1, 2, ..., N$ are assumed to be independent.

Figure 5 displays all waveforms recorded on an electrode. This is a common way to visualize waveform clusters. In this example the overlaid voltage traces follow two distinct

shapes. Visual inspection is often used to aid and to monitor more formal spike-sorting procedures (Wood et al., 2004).

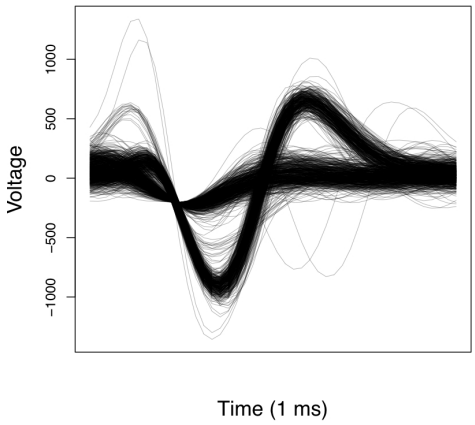

Figure 5: The waveform data WF collected on a single electrode: points of WF_j traced by a line and overlaid for all j. (*Data curtesy to the Doug Weber Lab*)

Spike-sorting consists of clustering the observed waveforms WF_j. In order to reliably estimate a clustering rule from the available data, we use a lower-dimensional representation. Examples include reducing the data to the minimum or maximum value, the amplitude, the width, or the first few coefficients in a principal component or wavelet decomposition.

A variety of clustering methods show good spike-sorting results. Examples include both nonparametric techniques such as k-means (Chah, 2011) and superparamagnetic clustering (Quiroga, 2004) and parametric methods such as mixture models (Lewicki, 1994). We prefer the model-based approach to clustering because it provides a statistically principled solution with well-studied optimality properties.

Using Bayes rule, we can write

$$P(X_j = k|WF_j = w) = \frac{P(X_j = k, WF_j = w)}{f(WF_j = w)} = \frac{f(WF_j = w|X_j = k)P(X_j = k)}{\sum_k f(WF_j = w|X_j = k)}$$

We use the fact that the waveforms of one neuron follow the same distribution. Let the distribution of waveforms w from neuron k have the probability density function $f_k(w)$. Let $P(X_j = k) = \pi_k$ be the proportion of spikes from neuron k. Then, the above becomes

$$P(X_j = k|WF_j = w) = \frac{\pi_k f_k(w)}{\sum_{j=1}^{K} \pi_k f_k(w)}$$

This implies a mixture distribution for the waveform data WF_j, namely

$$f(WF_j = w) = \sum_{j=1}^{K} \pi_k f_k(w), \ \forall j$$

It is difficult to estimate the parameters of a mixture distribution by directly maximizing the likelihood. This is why we use an expectation maximization (EM) algorithm.

We augment the waveform data WF with the latent identities X. Then,

$$f(WF_j = w, X_j = k) \ = \ P(X_j = k) f(WF_j = w | X_j = k) \ = \ \pi_k f_k(w).$$

Let each distribution let f_k be determined by parameters ψ_k. The parameters of the model are $\Pi \ = \ (\pi_1, ..., \pi_K)$ and $\Psi = (\psi_1, ..., \psi_K)$. The EM algorithm has a simple form when the mixture components f_k are Gaussian. For easier notation, let $f_k(WF_j; \psi_k)$ be one-dimensional Gaussian with parameters $\psi_k = (\mu_k, \sigma_k^2)$. Then, the EM algorithm takes the following form

EM Algorithm for Gaussian Mixture

1. Initialize by picking values for $\hat{\Pi}^{(0)}$ and $\hat{\Psi}^{(0)}$. Set $r \leftarrow 0$.

2. (E-step) Calculate the responsibility of WF_j to the component k

$$\gamma_{kj} = P(X_j = k | WF_j; \hat{\Pi}^{(r)}, \hat{\Psi}^{(r)}) = \frac{\pi_k^{(r)} f_k(WF_j; \hat{\psi_k}^{(r)})}{f(WF_j; \hat{\Pi}^{(r)}, \hat{\Psi}^{(r)})}$$

3. (M-step) Update the parameters:

$$\hat{\mu_k}^{(r+1)} = \frac{\sum_j \gamma_{kj} WF_j}{\sum_j \gamma_{kj}}$$

$$\hat{(\sigma_k^2)}^{(r+1)} = \frac{\sum_j \gamma_{kj} (WF_j - \hat{\mu_k}^{(r+1)})^2}{\sum_j \gamma_{kj}}$$

$$\hat{\pi}_k^{(r+1)} = \frac{1}{T} \sum_j \gamma_{kj}$$

$r \leftarrow r + 1$

4. Repeat steps 2 and 3 until convergence.

(Hastie, Tibshirani, and Friedman, 2001)

This algorithm is commonly used to sort spikes into clusters in the space of their first few principal components. The assumptions that each cluster has a Gaussian distribution in the space of principal components can be relaxed to a t-distribution (Shoham, 2003). In

both cases an exact EM algorithm exists, i.e. all the calculations can be done analytically. Despite the strong assumptions this sorting method is effective (Stein et al., 2004).

2.2 Using covariate information for spike-sorting (Ventura, 2009b)

So far we have seen spike-sorting based solely on the shape of the waveforms. Ventura (2009b) proposes an automatic spike-sorting method which incorporates tuning information.

We can model the spike train of neuron k as a Poisson process with intensity $\lambda_k(c)$, (Kass and Ventura, 2001) where c is the value of behavioral covariates. The mixture proportion π_k of spikes from neuron k depends on the covariates through the firing intensity $\lambda_k(c)$.

$$f(WF_j = w|C = c) = \sum_{k=1}^{K} \pi_k(c) f_k(w|c) = \sum_{k=1}^{K} \pi_k(c) f_k(w)$$

We can assume that $f_k(w|c) = f_k(w)$ because a neuron's waveforms follow the same distribution regardless of the value of behavioral covariates. The posterior probability that waveform WF_j comes from neuron k when we observe covariates c is

$$P(X_j = k|WF_j = w, C = c) = \frac{\pi_k(c) f_k(w)}{\sum_{k=1}^{K} \pi_k(c) f_k(w)}$$

An exact EM algorithm to fit this model exists. It consists of iterating the estimation of waveform parameters with the estimation of tuning curve parameters (Ventura, 2009b).

Let $\lambda_k(c; \theta_k)$ be the tuning curve of neuron k. The model for the waveforms is a mixture of Gaussians with parameters Ψ. The goal is to find maximum likelihood estimates of the parameters (Θ, Ψ). To apply the EM algorithm we need to calculate the conditional expectation of the augmented log-likelihood

$$\begin{aligned} logf(WF, ST, X|C; \Theta, \Psi) &= logf(WF|ST, X, C; \Theta, \Psi) + logf(ST, X|C; \Theta, \Psi) = \\ &= logf(WF|X; \Psi) + logf(ST, X|C; \Theta) = \\ &= logf(WF, X; \Psi) + logf(ST, X|C; \Theta) - logf(X), \end{aligned}$$

using Bayes' rule and the definition of each distribution.

This decomposition allows us to calculate EM updates for Ψ and Θ iteratively.

The Ventura (2009b) algorithm is defined as follows

EM Algorithm for Joint Waveform and Tuning Model (Ventura, 2009b)

1. Initialize by picking values for $\hat{\Psi}^{(0)}$ and $\hat{\Theta}^{(0)}$. Set $r \leftarrow 0$.

2. **Waveform E-step** Calculate responsibilities of each waveform WF_j, to the waveform cluster for neuron k,

$$\gamma_{kj} = P(X_j = k|WF_j, ST_j, C(ST_j); \hat{\Theta}^{(r)}, \hat{\Psi}^{(r)})$$

3. **Waveform M-step** Update the waveform parameters

$$\hat{\mu}_k^{(r+1)} = \frac{\sum_j \gamma_{kj} WF_j}{\sum_j \gamma_{kj}}$$
$$\hat{\sigma^2}_k^{(r+1)} = \frac{\sum_j \gamma_{kj}(WF_j - \hat{\mu}^{(r+1)})^2}{\sum_j \gamma_{kj}},$$

4. **Tuning E-step** Calculate the responsibility of each spike to each neuron k

$$e_{kj} = P(X_j = k|WF_j, ST_j, C(ST_j); \hat{\Theta}^{(r)}, \hat{\Psi}^{(r+1)}) = \gamma_{kj}$$

5. **Tuning M-step** Regress e_{kj} on C, to obtain $\hat{\Theta}^{(r+1)}$, $j \leftarrow r + 1$.

6. Repeat steps 2 - 5 until convergence.

This procedure produces unbiased estimates of the tuning curves unlike traditional spike-sorting (Ventura, 2009a).

This algorithm spike-sorts based on waveform and tuning information but assumes that all spikes are independent. Section 2.4 describes some of the ways to model the dependence of a neuron's spike train on history and on the activity of other neurons.

2.3 Using correlation between spike trains for spike-sorting (Ventura and Gerkin, 2011)

Ventura and Gerkin (2011) show that traditional waveform based spike-sorting yields biased estimates of correlation (more specifically, rates of coincident spiking). They also prove that this bias can be avoided if spike identities X are estimated jointly as opposed to independently in what they refer to as "ensemble sorting". They demonstrate this method by determining the spike-sorting rule for a two-neuron directed model with known parameters. This is a special case of the general modeling framework I will use in my work.

The methods we discussed so far model the spike train of a neuron as a Poisson process. The refractory period of a neuron is one violation of the Poisson assumption. Another reason the Poisson model is not always adequate is that it does not allow the activity of a neuron to depend on other neurons. To accommodate such dependence we use a more

general point process framework. We can define a counting process $N(t)$ by the conditional intensity function $\lambda(t|H(t))$, where $H(t)$ is the history of the process until time t. The conditional intensity

$$\lambda(t|H(t)) = \lim_{\Delta \to 0} \frac{P(N(t+\Delta t) - N(t) = 1|H(t))}{\Delta}$$

completely defines the process (Daley and Vere-Jones, 2003).

This framework allows us to define the firing rate $\lambda_k(t|H(t))$ of a neuron k, as a function of the neuron's history and the history of other neurons. For example, we can model a dependence on the time since the last spike. Let $G_k(t)$ be the time elapsed between the last spike of neuron k and time t. For a model where the rate of neuron k depends only on the time of its last spike we define

$$\lambda_k(t|H(t)) = \lambda_k(t|G_k(t)).$$

Kass and Ventura (2001) use a multiplicative model $\lambda_k(t|G_k(t)) = \alpha(t)\beta(G_k(t))$ which they fit by standard regression techniques. To enforce a refractory period of length η we can restrict $\beta(.)$ to 0 over the interval $(0, \eta)$.

To extend this approach to a general ensemble of K neurons, let

$$\lambda_k(t|H(t)) = \alpha_k(t) \prod_{j=1}^{K} \beta_k(G_j(t)).$$

See Section 2.4 for a literature overview of joint spiking models.

Ventura and Gerkin (2011) treat the case of one electrode recording the activity of two neurons: A and B. Neuron A has a conditional intensity function $\lambda_A(t|H(t)) = \lambda_1$, and neuron B has a conditional intensity function $\lambda_B(t|H(t)) = \lambda_2\beta(G_A(t))$ where $\beta(g) = 1.I\{g > \nu\} + q.I\{g \leq \nu\}$. Therefore, neuron A spikes as a Poisson process with constant rate $\lambda_1$and neuron B spikes as a Poisson process with rate $\lambda_3 = q\lambda_2$ for ν seconds after a spike from neuron A and rate λ_2 the rest of the time. We refer to this model as the "neuron A influences neuron B" model. Figure 6 shows a sample spike train from the model.

If we discretize time into 1-millisecond bins, so that each bin contains no more than one spike, we can write the electrode spike train as a vector $z = (z_1, z_2, ..., z_B)$, where $z_l = 1$ if the l^{th} time bin contains a spike and 0 otherwise. The influence of neuron A on neuron B extends over ν bins. The probability that a bin contains a spike from neuron A is

$$p_1 = 1 - e^{-\lambda_1}.$$

The probability that a bin contains a spike from neuron B is

$$\begin{aligned} p_2 =& (1 - e^{-\lambda_2})P(\text{A did not spike in the last } \nu \text{ bins}) + (1 - e^{-\lambda_3})P(\text{A spiked in the last } \nu \text{ bins}) \\ =& (1 - e^{-\lambda_2})(1 - p_1)^\nu + (1 - e^{-\lambda_3})(1 - (1 - p_1)^\nu) \\ =& (1 - e^{-\lambda_2})e^{-\lambda_1\nu} + (1 - e^{-\lambda_3})(1 - e^{-\lambda_1\nu}) \end{aligned}$$

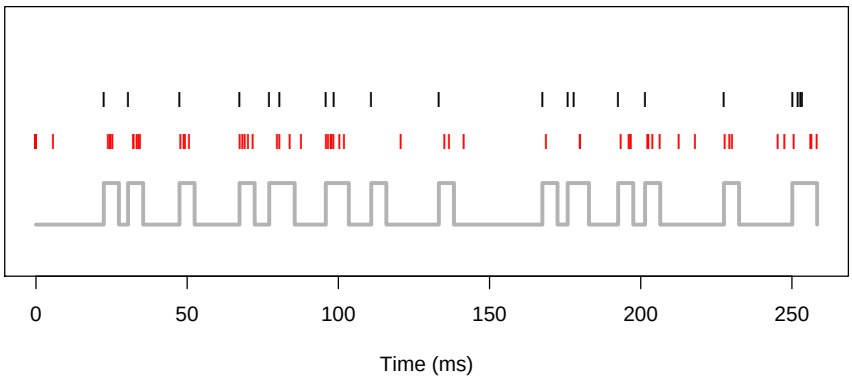

Figure 6: A sample from the "neuron A influences neuron B" model with parameters $\lambda_1 = 0.1$, $\lambda_2 = 0.05$, $\lambda_3 = 0.6$, $\nu = 5$. Spike train of neuron A (black), spike train of neuron B (red), neuron B intensity function (grey).

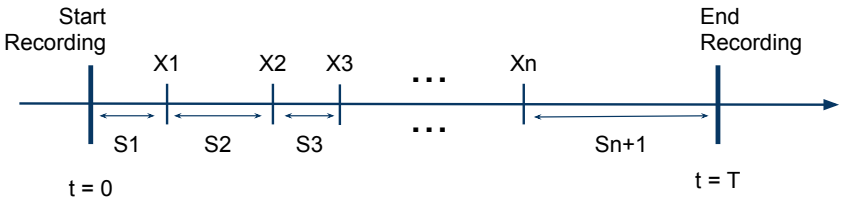

Figure 7: Observed electrode spike train in terms of inter spike intervals S_j and latent spike identities X_j. S_1^* and S_{n+1}^* denote the partially observed inter spike intervals at the two ends of the recording window.

Denote the inter spike intervals by $S = (S_1, S_2, ..., S_n, S_{n+1})$ (Figure 7). In terms of the spike times, $S_j = ST_j - ST_{j-1}$, for $j = 2, 3, ..., n$, $S_1 = ST_1$, and $S_{n+1} = T - ST_n$. Note that the experiment runs from time $t = 0$ to time $t = T$.

Sorting the spikes jointly requires calculating $P(X = i|WF, ST)$ for every possible vector of identities i, $i = (i_1, i_2, ..., i_n)$, where $i_j \in \{1, 2, ..., K\}$, $\forall j$. Here we outline the steps involved in calculating $P(X = i|ST)$. The waveform information can then be added as in Ventura (2009b). The distribution of X_j is determined by the history of the process $(S_1, X_1, ..., X_{j-1}, S_j)$ because it indicates the time past between the last spike of neuron A and the current time ST_j. If more than ν seconds have passed, $P(X_j = 1) = \lambda_1/(\lambda_1 + \lambda_2)$, otherwise, $P(X_j = 1) = \lambda_1/(\lambda_1 + \lambda_3)$. Similarly, the conditional distribution of $(S_j|X_{j-1}, S_{j-1}, ..., X_1, S_1)$ is a known function of the model parameters. This suggests that we can calculate the joint distribution $P(X = i, S = s)$ as a product of conditional

probabilities

$$\begin{aligned} P(X=i,S=s) =& f(s_{n+1}|i_n,s_n,i_{n-1},...,i_1,s_1) \\ & .P(i_n|s_n,i_{n-1},s_{n-1},...,i_1,s_1) \\ & .f(s_n|i_{n-1},s_{n-1},i_{n-2},...,i_1,s_1) \\ & \vdots \\ & .P(i_1|s_1) \\ & .f(s_1), \end{aligned}$$

The joint probability $P(X=i,S=s)$ allows us to calculate the posterior probability of each identity vector i as

$$P(X=i|S=s) = \frac{P(X=i,S=s)}{f(S=s)} = \frac{P(X=i,S=s)}{\sum_i P(X=i,S=s)}$$

And finally, the posterior probability of individual spike identities is the marginal probability

$$P(X_j=k|S=s) = \sum_{i:i_j=k} P(X=i|S=s)$$

2.4 Joint spiking models overview

Modeling an ensemble of neuron spike trains as independent Poisson processes does not allow us to extract information about how the observed neurons work together to encode messages. Joint activity models are essential in analyzing data from a population of neurons. Here we review the use of such models in the literature.

Truccolo et al. (2005) use a point process model for a neuron spike train specified in terms of a conditional intensity function $\lambda(t|\mathcal{H}(t))$. Here $\mathcal{H}(t)$ denotes the spike history, i.e. the value of the process over the interval $(0,t)$. Truccolo et al. use this model to include dependence on the spiking history of an entire ensemble of neurons and covariates such as velocity. They fit the discrete time point process likelihood parameters via a generalized linear model. Limiting the history dependence to short immediate history allows for efficient computation. The paper demonstrates the use of the model for decoding and explores its statistical advantages for goodness-of-fit calculations, residual analysis and model selection.

Loglinear point process models are also used to evaluate synchrony. Kass et al. (2011) study the probability of synchronous or precisely lagged spikes. For example, to study the synchronous behavior of two neurons with spike trains $X^1(t)$ and $X^2(t)$, they define the quantity of interest $P^{1,2}_{a,b}(s,t|\mathcal{H}) = P(X^1(s)=a, X^2(t)=b|\mathcal{H})$, where $\mathcal{H}$ represents the spiking history of the neuron population. Kass et al. propose four practical restrictions of the general model:

(i) only allow low-order interactions,
(ii) assume that $P^{1,2}_{a,b}(s,t)$ and $P^{1,2}_{a,b}(s,t|\mathcal{H})$ are smooth in t,
(iii) restrict $\mathcal{H}$ to a neuron's own spiking history, and
(iv) adapt loglinear model methodology for estimation.

The authors introduce a test for dependence between spike trains defining $\zeta(t)$ such that $P^{1,2}_{1,1}(t,t) = P^1_1(t)P^2_1(t)\zeta(t)$. If $X^1(t)$ and $X^2(t)$ are independent, $\zeta(t) = 1, \forall t$.

Modeling correlated activity has proven to be important for decoding as well (Pillow et al., 2008). There the authors include filters to model spike-train dependency on the recent spiking history of a population of neurons. This is a natural extension of the linear-nonlinear-Poisson (LNP) cascade model (Paninski, 2004). The general LNP model has the form $p(spike|\vec{y}) = f_\theta(K\vec{y})$, where $\vec{y}$ is the stimulus, K is a linear operator with low rank, and θ is the parameter of the nonlinear f which maps $K\vec{y}$ to the probability of a spike. Cascade models could be motivated by an analogy with the biophysical mechanisms behind spike generation. Despite the flexibility in model parameters, finding maximum likelihood estimates for these models is highly tractable (Paninski, 2004). Herzfeld and Beardsley (2010) use a population temporal filter for decoding directly from the electrode spike trains avoiding spike-sorting. Their method requires less computations and can produce results comparable to those from traditional methods which require spike-sorting as an initial step.

3 Preliminary Work

3.1 Spike-sorting using waveform and tuning information

Ventura (2009b) shows that spike-sorting using waveform and tuning information yields unbiased estimates of tuning curves. The algorithm for estimating spike identities and tuning at the same time was only tested on simulation data. I implemented the Ventura (2009b) algorithm and the traditional spike-sorting algorithm and applied them to data to compare the resulting tuning curve estimates.

Comparing the performance of the two algorithms on real data is not straight-forward because we do not know the true identities of the spikes. Some experiments record both intracellular and extracellular signals (e.g, Harris et al., 2000) which provides the true identities of some of the spikes in extracellular recordings. However, I was not able to find such data set which also contains behavioral covariate information. Instead, I used data from an experiment at the Doug Weber lab. The data contains extracellular recordings from the cat dorsal root ganglion and the position of the hind leg in terms of hip, knee and ankle joint angles (Stein et al., 2004). I constructed test data from the recordings on two different electrodes. I selected one well-isolated cluster of waveforms from each of the two electrodes and combined them. This guarantees that the test data contains signals from two different neurons because the range of electrodes does not overlap.

I used a Gaussian model in the first two principal components for the waveforms and splines to smooth the tuning curves in both the Ventura (2009b) and the traditional spike-sorting method. Figure 8 shows the estimated tuning curves from the traditional spike-

sorting (PC), from the Ventura (2009b) algorithm (PC + tuning), and from the true spike identities (True). The tuning curves obtained via the Ventura(2009b) combined waveform and tuning method appear sharper and closer to the truth.

There are several factors to consider before drawing any conclusions from this comparison. The appearance of the tuning curves is greatly influenced by the model used and the distribution of the data. Here we use splines for a flexible nonparametric model and select the position of the knots based on the true tuning curves. For this data set even a low misclassification rate can result in a big change in the spline fit. Lastly, the performance of the algorithm is sensitive to starting values.

A more systematic approach is to compare the results of decoding using each method. This is a direct application of available methodology. We can use a full data set without modifications. The decoding results allow us to assess which algorithm provides a better use of the data for practical applications. This comparison is part of our future work. See Section 4 for details.

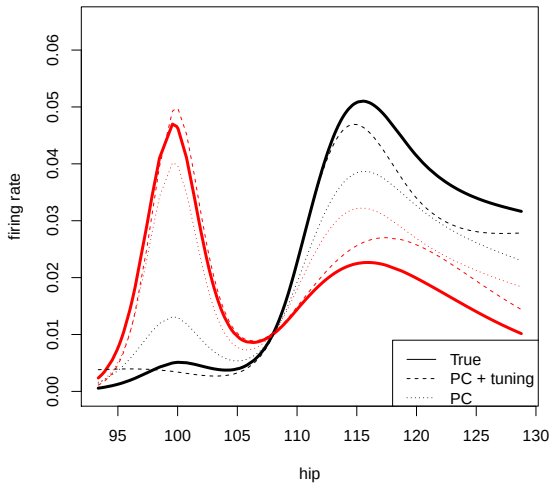

Figure 8: Tuning curves for two neurons (red and black) obtained by traditional spike-sorting (PC), Ventura (2009b) waveform and tuning algorithm (PC + tuning), and by smoothing the true tuning curves (True).

3.2 Algorithm to fit the parameters of the "Neuron A influences neuron B model" from Ventura and Gerkin (2011)

We developed an EM algorithm to fit the parameters of the "Neuron A influences neuron B" model described in Section 2.3. The parameters of the model are $\Theta = (\lambda_1, \lambda_2, \lambda_3, \nu)$. At this stage we assume that ν is known, because it is a nonlinear parameter, and estimate $\lambda_1, \lambda_2, \lambda_3$. Let $k = 1$ denote neuron A and $k = 2$ neuron B.

In order to define the EM algorithm we need to:

1. Specify the augmented data log-likelihood $l_0(S, X; \Theta)$ in terms of the model parameters Θ.
2. (E-Step) Find the conditional expectation $Q(\Theta, \hat{\Theta}) = \mathbb{E}[l_0(S, X; \Theta)|S; \hat{\Theta}]$.
3. (M-Step) Find the value of Θ which maximizes $Q(\Theta, \hat{\Theta})$.

In the algorithms discussed so far, we were able to write Q in terms of the *responsibilities* γ_{kj}, i.e. the probability that a spike j spike was generated by neuron k. This is the same quantity we use to sort the spikes. When using only waveform information, responsibilities depend only on the waveform values (Section 2.1). When tuning is included, responsibilities also depend on the covariates(Section 2.2). In either case, we were able to calculate responsibilities independently for each spike. However, when we do not have independence, the identity of a spike influences the identities of other spikes. Therefore, the joint responsibility of all spikes to a vector of identities is not necessarily the product of the individual neuron responsibilities. The necessary calculations to determine joint responsibilities depend on the model. It is sufficient to calculate the joint responsibility of all spikes to all possible identities, $P(X = i|S)$, for all possible identity vectors i. For K neurons and n spikes this means calculating K^n responsibilities. Such an approach is not feasible for a realistic data application. We need to use a model which allows us to calculate Q without computing all possible joint responsibilities separately.

3.2.1 Special case: $\nu = 0$

In the special case when $\nu = 0$, the two neurons are independent with rates λ_1 and λ_2. Then,

$$\begin{aligned} l_0(S, X; \lambda_1, \lambda_2, \nu = 0) =& \sum_{j=1}^{n} I\{X_j = 1\} \left(log\lambda_1 - (\lambda_1 + \lambda_2)S_j\right) + \\ &+ \sum_{j=1}^{n} I\{X_j = 2\} \left(log\lambda_2 - (\lambda_1 + \lambda_2)S_j\right) - (\lambda_1 + \lambda_2)S_{n+1} = \\ &= log\lambda_1 \sum_{j=1}^{n} I\{X_j = 1\} + log\lambda_2 \sum_{j=1}^{n} I\{X_j = 2\} - (\lambda_1 + \lambda_2) \sum_{j=1}^{n+1} S_j. \end{aligned}$$

Therefore, the E-Step of the EM algorithm is

$$\begin{aligned} Q(\Theta, \hat{\Theta}) &= \mathbb{E}[l_0(S, X)|S; \hat{\Theta}] = \\ &= log\lambda_1 \sum_{j=1}^{n} \mathbb{E}[I\{X_j = 1\}|S; \hat{\Theta}] + \\ &+ log\lambda_2 \sum_{j=1}^{n} \mathbb{E}[I\{X_j = 2\}|S; \hat{\Theta}] - (\lambda_1 + \lambda_2) \sum_{j=1}^{n+1} S_j = \\ &= (log\lambda_1) \frac{n\hat{\lambda}_1}{\hat{\lambda}_1 + \hat{\lambda}_2} + (log\lambda_2) \frac{n\hat{\lambda}_2}{\hat{\lambda}_1 + \hat{\lambda}_2} - (\lambda_1 + \lambda_2) \sum_{j=1}^{n+1} S_j \end{aligned}$$

The M-step update for λ_1 is:

$$\underset{\lambda_1}{\text{argmax}}\ Q(\Theta, \hat{\Theta}) \ = \ \frac{\hat{\lambda}_1}{\hat{\lambda}_1 + \hat{\lambda}_2} \frac{n}{\sum S_i} \ = \ \gamma_1 \frac{n}{\sum S_i},$$

where γ_1 is the probability that a spike belongs to neuron A. Iterating these updates always yields the same values of $\hat{\lambda}_1$ and $\hat{\lambda}_2$ because the likelihood of the data is the same for all $\hat{\lambda}_1$ and $\hat{\lambda}_2$, such that $\hat{\lambda}_1 + \hat{\lambda}_2 = n/\sum S_j$. We need to fix one of the two rates to make the model identifiable.

3.2.2 General case

The augmented log-likelihood $l_0(S, X; \Theta)$ can be calculated in the same way as in Ventura and Gerkin (2011).

$$\begin{aligned} l_0(S, X; \Theta) &= l(S_{n+1}|X_n, S_n, X_{n-1}, ..., S_2, X_1, S_1) + \\ &+ l(X_n|S_n, X_{n-1}, S_{n-1}, ..., S_2, X_1, S_1) + ... + l(X_1|S_1) + l(S_1), \end{aligned}$$

where $l(.)$ is the log-likelihood of the denoted random variable.

This decomposition allows us to calculate each component exactly because it only involves conditional distributions given the entire past.

In Section 2.3 we showed how to calculate the joint posterior probability of spike identities given the observed inter spike intervals, $P(X = i|S = s)$, for any vector of identities i. Let

$$\gamma_i = P(X = i|S = s).$$

Define $n_u(i|S = s)$ to be the number of spikes in the data from a neuron spiking as a Poisson process with rate λ_u for inter spike intervals s and spike identities i. Note that

given a spike identities vector i and the observed spike train S we can partition the interval of time $(0, T)$ into periods when neuron B has rate λ_2 and periods when it has rate λ_3, i.e. all periods of ν seconds after a spike j such that $i_j = 1$ (i.e. a spike from neuron A). To calculate $n_2(i|S = s)$ and $n_3(i|S = s)$, we count the number of spikes j with $i_j = 2$ and ST_j in a period with rate λ_2 and λ_3 respectively. Note that $n_1(i|S = s)$ is the count of spikes from neuron A in the vector of identities i, i.e. $|\{j : i_j = 1\}|$. Then, $n_1(i|S = s) + n_2(i|S = s) + n_3(i|S = s) = n$. For example, for the spike train in Figure 9, $n_1(i|S = s) = 5$, $n_2(i|S = s) = 2$, $n_3(i|S = s) = 10$.

Similarly, define $T_u(i|s = s)$ to be the cumulated period of time when we record from a neuron with rate λ_u. We have $T_1(i|S = s) = T$, $\forall i$. We calculate $T_3(i|S = s)$ as the total period of time when neuron B spikes with rate λ_3 and $T_2(i|S = s) = T - T_3(i|S = s)$.

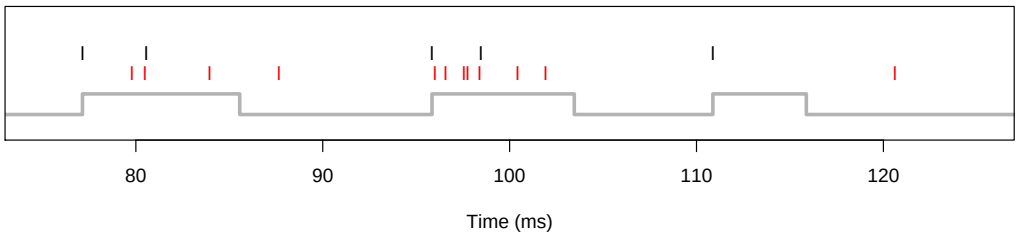

Figure 9: A subsample of the spike train in Figure 6. Spikes from neuron A (black), spikes from neuron B (red), and firing rate of neuron B (grey).

The M-step updates for each λ_u are

$$\underset{\lambda_u}{\text{argmax}}\ Q(\Theta, \hat{\Theta}) \ = \ \frac{\sum_i \gamma_i n_u(i|S = s)}{\sum_i \gamma_i T_u(i|S = s)}$$

3.2.3 Implementation

I implemented the algorithm to estimate λ_1 when $\lambda_2, \lambda_3, \nu$ are known. Figure 10 shows the estimates for λ_1 during the first ten iterations for five runs of the EM algorithm on the same data with different starting values. All runs of the algorithm converge to the same estimate, which exceeds both the true value of λ_1 and the MLE of λ_1 for this data set.

Figure 11 summarizes the estimates of λ_1 over 8179 spike trains simulated from the model with parameters $\lambda_1 = 0.1$, $\lambda_2 = 0.05$, $\lambda_3 = 0.2$, $\nu = 5$, $T = 30$. For these runs the value of ν is fixed at the truth and the values of λ_2 and λ_3 are the MLE given the true labels for each spike train.

The results suggest that there is bias in the estimates of λ_1 under this framework. One possibility is that this is due to bias in the MLE of λ_2 and λ_3 for spike trains of this length and we need to use much longer spike trains to obtain unbiased estimates. We need to

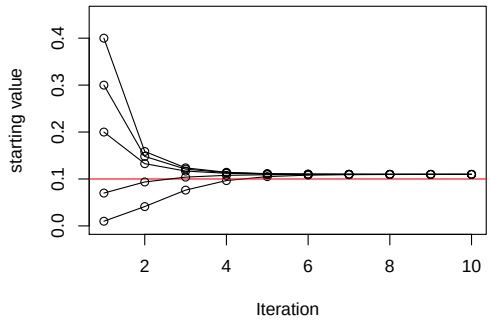

Figure 10: Estimate for λ_1 over the first ten iterations for five runs of the EM algorithm with different starting values. The data is one spike train from the "Neuron A influences neuron B" model with parameters $\lambda_1 = 0.1$ (red line), $\lambda_2 = 0.05$, $\lambda_3 = 0.2$, $\nu = 5$, $T = 50$.

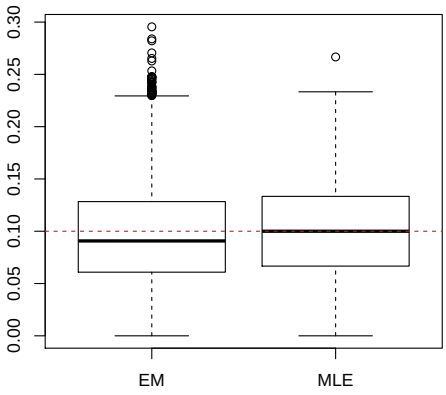

Figure 11: Estimates of λ_1 from 8179 spike trains simulated from the "Neuron A influences neuron B model" with parameters $\lambda_1 = 0.1$, $\lambda_2 = 0.05$, $\lambda_3 = 0.2$, $\nu = 5$, $T = 30$. (EM) estimates of λ_1 from the EM algorithm with λ_2 and λ_3 fixed at their MLE for each spike train. (MLE) maximum likelihood estimate of λ_1 calculated using the true identity of the spikes

determine if the MLE is unbiased or only asymptotically unbiased in this case. [Results from longer spike trains to be included in the next draft.]

The implementation of the algorithm for longer spike trains requires taking advantage of the fact that for any inter spike interval S_j such that $S_j > \nu$, $(S_1, X_1, S_2, ..., S_{j-1}, X_{j-1})$ are jointly independent from $(X_j, S_{j+1}, X_{j+1}, ..., X_n)$, as shown in Ventura and Gerkin (2011). Thus, we can calculate the components of the joint likelihood separately for parts of the observed spike train and multiply them together to calculate Q. This reduces the number

of responsibilities to calculate from K^n to a number less than nK^m, where m is the largest number of consecutive inter spike intervals less than ν. In real data electrode spike trains contain thousands of spikes but the length of the history effect ν with respect to the inter spike intervals of neurons is low assuring that m is also low and the computational time of this procedure is constrained.

4 Research Plan

The goal of this thesis is to extend current spike-sorting methodology to incorporate in a principled way all available variables which contain information about spike identities.

Here are the steps we intend to follow.

Methodology

4.1 Complete the algorithm for the "Neuron A influences neuron B" model

The "Neuron A influences neuron B" is a toy example we want to understand well before proceeding. It provides a simple framework which can be generalized to a more realistic model. We have an algorithm to estimate the posterior probability of spike identities given the observed spike train $P(X_j = k|S),\ \forall j$. The next step is to extend the algorithm to also include waveform data and obtain $P(X_j = k|S, WF)$. For stationary waveform clusters, this can be done by iterating updates for the waveform distribution parameters with updates for the firing rate parameters similarly to Ventura (2009a).

Then, we will allow the distribution of waveform to depend on history to model attenuation. This can be done by modeling the amplitude decay of spike j as a function $\rho(S_j)$ of the time past since the previous spike.

Lastly, we will use simulated data to verify that the algorithm provides unbiased estimates of the model parameters.

4.2 Extend the algorithm to a general population of neurons

Generalizing the "Neuron A influences neuron B" model involves allowing the firing rates to depend on a neuron's own history, on the activity of neurons recorded on other electrodes, and on behavioral covariates. Also, we need a procedure for estimating the number of neurons recorded on each electrode. One possible solution is to use a penalized likelihood approach such as BIC.

The choice of models for the waveform clusters, tuning curves, and correlations is often motivated by computational convenience. In some cases, there are established models, e.g. cosine models for tuning to velocity. We need to formally determine a class of appropriate models for each modeled relationship and incorporate model selection into the algorithm. This can be done by iteratively updating the parameters and the choice model inside the EM algorithm.

A simulation study will determine if the algorithm provides unbiased estimates of the model parameters.

Applications

4.3 Explore the performance of the method on real data

As discussed in Section 3.1, testing the method on real data is difficult because the spike identities are not observed. Instead, we propose to test the performance of the algorithms by the quality of decoding in brain machine interfaces. The first step is to compare the quality of decoding using the Ventura (2009b) algorithm and using traditional spike-sorting on real data. We intend to use data from an experiment similar to the one it Georgopoulos (1982) conducted at the Andrew Schwartz lab (see Fraser et al. (2009) for details on the experiment).

We are also interested in evaluating the proposed methodology in terms of the quality of correlation estimates it provides. An appropriate test data set can come from a region where neuron interactions are well studied, such as the retinal ganglion (Pillow et al., 2005).

4.4 Evaluate and improve computational efficiency

The complete proposed methodology is more computationally intensive than traditional spike-sorting. After determining in what settings the improvement provided by the proposed algorithm justifies the added computational cost, we can identify possible simplifications to make the algorithm more useful in practice. If the algorithm proves valuable for the neuroscience community, we will optimize the implementation for speed and make it user-friendly.

References

Brown EN, Kass RE, Mitra PP. 2004. Multiple neural spike train analysis: state-of-the-art and future challenges. Nat Neurosci 5:456-61.

Buzs̀aki G. 2004. Large-scale recordings of neural ensembles. Nat Neurosci 7:446-51.

Calabrese A, Paniniski L. 2011. Kalman filter mixture model for spike sorting of non-stationary data. J Neurosci Meth 196:159-69.

Cohen MR and Kohn A. 2011. Measuring and interpreting neuronal correlations. Nat Neurosci 14:811-819.

Delay D and Vere-Jones D. 2003. An Introduction to the Theory of Point Processes. New York: Springer-Verlag.

Delescluse M, Pouzat C. 2006. Efficient spike-sorting of multi-state neurons using inter-spike intervals information. J Neurosci Meth 150:16-29.

Donoghue JP. 2002. Connecting cortex to machines: recent advances in brain interfaces. Nat Neurosci 5:1085-8.

Fraser GW, Chase SM, Whitford A, Schwartz AB. 2009. Control of a brain-computer interface without spike sorting. J Neural Eng 6(5):055004.

Georgopoulos AP, Kalaska JF, Caminiti R, and Massey JT. 1982. On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. J Neurosci. 2:1527-37.

Grün S. 2009. Data-driven significance estimation for precise spike correlation. J Neurophysiol 101:1126-40.

Harris KD, Henze DA, Csicsvari J, Hirase H, Buzsaki G. 2000. Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. J Neurophysiol 84:401-14.

Harvey CD, Collman F, Dombeck DA, and Tank DW. 2009. Intracellular dynamics of hippocampal place cells during virtual navigation. Nature 461:08499.

Hastie T, Tibshirani R, and Friedman J.2001. Elements of statistical learning: Data mining, inference and prediction. Berlin: Springer-Verlag.

Henze DE. 2000. Intracellular features predicted by extracellular recordings in the hippocampus *in vivo*. J Neurophysiol 24:272-85.

Ito H, Maldonado PE, Gray CM. 2010. Dynamics of stimulus-evoked spike timing correlations in the cat lateral geniculate nucleus. J Neurophysiol 104:3276-92.

Herzfeld D J, Beardsley S A. 2010. Improved multi-unit decoding at the brain-machine interface using population temporal linear filtering. J Neural Eng 7:046012.

Kass R, Ventura V, 2001. A Spike Train Probability Model. Neural Comp 13:1713-20.

Kass RE, Ventura V, Brown EN. 2005. Statistical Issues in the analysis of neuronal data. J Neurophysiology 94:8-25.

Kass R, Kelly R, Loh W-L. 2011. Assessment of synchrony in multiple neural spike trains using loglinear point process models. Annals of Applied Statistics. To appear.

Kelly, R, Kass R, Smith M A, Lee TS. 2010. Accounting for network effects in neuronal responses using L1 regularized point process models. Advances in Neural Information Processing Systems. In Press.

Lewicki MS. 1994. Bayesian modeling and classification of neural signals. Neural Comp 6:1005-30.

Lewicki MS. 1998. A review of methods for spike sorting: the detection and classification of neural action potentials. Network 9:R53-78.

Okatan M, Wilson M, Brown E. 2005. Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. Neural Comput 17:1927-61.

Paniniski L. 2004. Maximum likelihood estimation of cascade point-process neural encoding models. Network: Comput Neural Syst 15:243-62.

Paninski L, Brown EN, Iyengar S, Kass RE. 2009. Statistical models of spike trains. In Stochastic Methods in Neuroscience, (Liang C, Lord GJ eds).Oxford. Clarendon Press 278-303.

Pillow J, Paninski L, Shlens J, Simoncelli E, Chichilnisky E. 2005. Modeling multi-neuronal responses in primate retinal ganglion cells. Comp Sys Neur'05.

Pillow J, Shlens J, Paninski L, Sher A, Litke AM, Chichilnisky EJ, Simoncelli EP. 2008. Spatio-temporal correlations and visual signaling in a complete neuronal population. Nature 454(7207):995-9.

Pouzat C, Mazor O, Laurent G. 2002. Using noise signature to optimize spike-sorting and to assess neuronal classification quality. J Neurosci Meth 122:43-57.

Pouzat C. 2005. Technique(s) for spike-sorting. In: Dalibard J, editor. Methods and models in neurophysics. Berlin: Springer-Verlag. 729-86.

Quiroga R, Nadasdy Z, Ben-Shaul Y. 2004. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. Neural Comp 16:1661:87.

Sahani M. 1999. Latent variable models of neural data analysis. PhD Thesis, California Institute of Technology: Pasadena.

Shoham S, Fellows M R, Normann R A. 2003. Robust, automatic spike sorting using mixtures of multivariate t-distributions. J. Neurosci Meth. 127:111-22.

Stein R B, et al. 2004. Coding of position by simultaneously recoded sensory neurons in the cat dorsal root ganglion. J Physiol 560:883-96.

Truccolo W, Eden U, Fellows M, Donoghue J, Brown E. 2005. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. J Neurophysiol 93: 1074-89.

Ventura V, Cai C, Kass RE. 2005. Statistical assessment of time-varying dependency between two neurons. J Neurophysiol 94:2940-7.

Ventura V. 2008. Spike train decoding without spike sorting. Neural Comp 20:923-63.

Ventura V. 2009a. Traditional waveform based spike sorting yields biased rate code estimates. PNAS. 106:6921-26.

Ventura V. 2009b. Automatic spike sorting using tuning information. Neural Comp 21:2466-501.

Wood F, Black M, Vargas-Irwin C, Fellows M, and Donoghue J P. 2004. On the variability of manual spike sorting. IEEE transactions on biomedical engineering. 51:912-8.