Automatic Spike Sorting Using Tuning Information

Valérie Ventura
vventura@stat.cmu.edu
Department of Statistics and the Center for the Neural Basis of Cognition, Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.

Current spike sorting methods focus on clustering neurons' characteristic spike waveforms. The resulting spike-sorted data are typically used to estimate how covariates of interest modulate the firing rates of neurons. However, when these covariates do modulate the firing rates, they provide information about spikes' identities, which thus far have been ignored for the purpose of spike sorting. This letter describes a novel approach to spike sorting, which incorporates both waveform information and tuning information obtained from the modulation of firing rates. Because it efficiently uses all the available information, this spike sorter yields lower spike misclassification rates than traditional automatic spike sorters. This theoretical result is verified empirically on several examples. The proposed method does not require additional assumptions; only its implementation is different. It essentially consists of performing spike sorting and tuning estimation simultaneously rather than sequentially, as is currently done. We used an expectation-maximization maximum likelihood algorithm to implement the new spike sorter. We present the general form of this algorithm and provide a detailed implementable version under the assumptions that neurons are independent and spike according to Poisson processes. Finally, we uncover a systematic flaw of spike sorting based on waveform information only.

1 Introduction

Extracellular electrodes are commonly used to monitor populations of neurons. The signal collected at an electrode is a mixture of activities from different neurons, corrupted by noise. Spike sorting consists of finding out how many neurons contributed to the recorded data and determining which neurons produced which spikes. Many spike-sorting solutions have been proposed, all based on clustering characteristic features of the recorded spike waveforms (see Lewicki, 1998, for a review). Spike-sorting papers generally focus on two broad problems: feature selection and clustering techniques. Features can be raw waveform measurements or projections onto lower-dimensional spaces, such as principal components. Full or reduced measurements are numerical vectors that can be treated as points in a high-dimensional space. Spike sorting is achieved by identifying clusters in this cloud of points, which correspond to single neurons. Clustering techniques are many and range from ad hoc procedures (e.g., Fee, Mitra, & Kleinfeld, 1996), nonparametric techniques (e.g., k-means; Salganicoff, Sarna, Sax, & Gerstein, 1988), neural networks (Ohberg, Johansson, Bergenheim, Pedersen, & Djupsjobacka, 1996), to model-based clustering using mixtures of distributions (e.g., Lewicki, 1994; Sahani, Pezaris, & Andersen, 1997; Shoham, Fellows, & Normann, 2003). In this letter, we focus on the latter approach, often referred to as automatic spike sorting, because it requires less subjective judgment to classify spikes, provides a statistically complete and efficient solution to the spike clustering problem, and is well suited to process many recording electrodes in a short amount of time. This approach has been evaluated in an experiment and validated with simultaneous intracellular and extracellular recordings, and it appears to perform well in many situations (Harris, Henze, Csicsvari, Hirase, & Buzsaki, 2000).

Spike sorting is typically the preliminary step to understanding how neurons represent information about covariates such as stimuli in sensory studies, behavioral correlates in motor studies, or, more generally, environmental parameters. This often involves estimating firing rates, tuning curves or functions, receptive fields, and so on as functions of these covariates, using the spike-sorted data. However, when these covariates modulate the neurons' firing rates, they too provide information about spike identities, which thus far has been ignored for the purpose of spike sorting. To see this, imagine an electrode that records two neurons whose tuning curves are modulated by some covariate c. Imagine that the first neuron spikes only when $c_1 < c < c_2$ and that the second neuron spikes only when $c_2 < c < c_3$, so they never spike together. Traditional spike sorters will assign a spike recorded at the electrode to one of the two neurons based on features of the recorded waveform, which might lead to a misclassification error if the neurons' waveform clusters overlap. But we can classify spikes with perfect confidence if we use c for spike sorting. Indeed, if we observe that $c_1 < c < c_2$ when a spike is detected at the electrode, then the spike must have been produced by neuron 1. If $c_2 < c < c_3$, then it is neuron 2 that must have spiked.

We propose a new automatic spike sorter that combines spike waveform information and tuning information. Because it uses more information, it yields lower spike misclassification rates than traditional automatic spike sorters do. The proposed method does not require additional assumptions; only its implementation is different. It essentially consists of performing spike-sorting and tuning estimation simultaneously rather than sequentially, as is currently done. We present the general form of the EM algorithm we developed to estimate the new spike sorter and provide a detailed implementable version under the assumptions that neurons are independent and spike according to Poisson processes. We illustrate the new sorter and its properties based on several simulated experiments.

2 Methods

Consider a single electrode that records I neurons. We collect the electrode spike train (EST) by thresholding its bandpassed voltage signal. When a spike is detected at time t, we record waveform measurements, which we denote by a_t. The methods presented in this letter apply generally, so a_t can be the raw waveform or any reduction, such as principal components (PCs). We also record the values c_t of covariates thought to modulate neurons that are spiking. For example, in section 3.1, $c_t = d_t$ is the angular direction of a moving hand in a movement decoding experiment, while in section 3.3, $c_t = t$ is the experimental time of a designed experiment. Once the electrode spike train is recorded, we divide time in bins of length δ that are small enough so that at most one spike can occur in a bin. Our illustrations all use $\delta = 1$ msec. Then our observed data are a spike train $\mathbf{z} = (z_t, t = 1, \ldots, T)$, where $z_t = 1$ (0) indicates that the electrode voltage did (did not) exceed the threshold at time t, the recorded waveform measurements $\mathbf{a} = (a_t, t = 1, \ldots, T)$[1], and the covariates $\mathbf{c} = (c_t, t = 1, \ldots, T)$.

The EST $\mathbf{z}$ is the aggregate of the I neurons' spike trains $\mathbf{y_i} = (y_{it}, t = 1, \ldots, T)$, $i = 1, \ldots, I$. The $\mathbf{y_i}$'s are not observed directly, and it is the purpose of spike sorting to estimate them by assigning spikes to neurons based on classifying the waveform measurements a_t. To develop our algorithm, it is useful to associate with a spike at t, the unobserved I-dimensional binary latent vector $x_t = (x_{1t}, \ldots x_{It})$, where $x_{it} = 1$ (0) means that neuron i did (did not) spike at t. When $z_t = 0$ (no spike was recorded at t), x_t is a vector of zeros (no neuron spiked). When $z_t = 1$, all we know is that x_t is not identically zero, and we let $\mathcal{X}$ denote the set of $(2^I - 1)$ distinct values x_t can take, which give all possible subsets of the I neurons spiking approximately together to produce a spike at t. For example, if a spike is detected at an electrode that records $I = 2$ neurons, the combinations of neurons that could have produced that spike can take $2^2 - 1 = 3$ values, $x = (1, 0) \equiv 10$, $(0, 1) \equiv 01$ or $(1, 1) \equiv 11$, depending on whether neuron 1 or 2 spiked alone or together. The complete set of latent variables is $\mathbf{x} = (x_t, t = 1, \ldots, T)$. In statistical jargon, $(\mathbf{z}, \mathbf{x})$ is a latent marked point process with $\mathbf{x}$ as the unobserved marking variable.

2.1 Classic Automatic Spike Sorting Based on Waveform Information.

Automatic spike sorting assumes that waveform features (WFs) a_t originate from one of several components, which typically correspond to I different neurons. To explicitly allow for neurons spiking together, we instead assume that a_t originates from one of $\text{card}(\mathcal{X}) = 2^I - 1$ components, each corresponding to a combination $x \in \mathcal{X}$ of neurons spiking approximately

[1]Waveform measurements are seldom collected when the electrode voltage is below threshold ($z_t = 0$). Missing values for a_t when $z_t = 0$ do not affect our method.

together to produce a spike. This choice is important to identify not only joint spikes but also individual neurons' spikes when the EST is contaminated by noise, as we illustrate later. Assuming that each spiking combination x accounts for a proportion π_x^* of the spikes, so that $\sum_{x \in \mathcal{X}} \pi_x^* = 1$, and that the distribution of spike WFs from that combination has distribution f_x^*, the probability distribution of WFs is

$$f^*(a) = \sum_{x \in \mathcal{X}} \pi_x^* f_x^*(a). \tag{2.1}$$

The superscript asterisk designates the true proportions and true distributions that give rise to the data, which distinguish them from the models we will later fit to the data. Estimation is treated in sections 2.3 and 2.4. Automatic spike sorting relies on Bayes' rule to obtain the probability that a spike with WF a was produced by x, that is,

$$P^*(x \mid a) = \frac{\pi_x^* f_x^*(a)}{f^*(a)}, \tag{2.2}$$

where the denominator is equation 2.1 and the numerator is its summands. The spike is then assigned to the combination $x^\dagger$ with the highest posterior probability,

$$x^\dagger(a) = \arg\max_{x \in \mathcal{X}} P^*(x \mid a),$$

with corresponding allocation for neuron $i = 1, \ldots, I$, the ith component of $x^\dagger(a)$, $x_i^\dagger(a)$. We apply this spike-sorting rule to each spike observed at the electrode and denote by $\mathbf{x}_i^\dagger$ the estimated spike train of neuron i.

The result of spike sorting is rarely perfect. In the ideal case, when the variabilities of each neuron's WFs are smaller than the variability between WFs from different neurons, WFs will form distinct clusters and neurons can be isolated perfectly. Then $\mathbf{x}_i^\dagger = \mathbf{y}_i$, the true spike train of neuron i. More often, clusters overlap so that the WFs that belong to overlapping regions cannot be classified with perfect confidence. Spike misclassification errors are unavoidable. However, if equation 2.1 is indeed the distribution of the data, the Bayes classification rule in equation 2.2 is known to be optimal, which means that it has the lowest spike misclassification rate of all spike-sorting rules (see e.g., Wasserman, 2004). But as we show in the next section, equation 2.1 does not use all the information in the data, so that the traditional automatic spike sorter is not fully efficient.

2.2 Incorporating Tuning Information for Spike Sorting. Traditional automatic spike sorting starts with the mixture in equation 2.1 to describe the distribution of spike WFs. This description ignores the information provided by firing rate modulating covariates c. We adopt the same approach to develop our spike sorter, but we let the distribution of WFs depend on c. This gives

$$f^*(a \mid c) = \sum_{x \in \mathcal{X}} \pi_x^*(c) f_x^*(a \mid c),$$

where $f_x^*(a \mid c)$ is the distribution of WFs produced by neuron combination x when the covariate takes value c. It reduces to $f_x^*(a)$, since waveforms are characteristic of the neurons that produce them and so should not depend on behavioral correlates. If c includes the experimental time of an experiment, this reduction also implies that spike waveform dynamics are assumed to be stationary. We comment on the nonstationary case in the discussion. The term $\pi_x^*(c)$ is the probability that neuron combination x spikes when the covariate takes value c. It is a function of the neurons' firing rates. To see that, consider an electrode that records $I = 2$ neurons that have true tuning curves $\lambda_i^*(c)$, $i = 1, 2$, with $\lambda_i^*(c)$ expressed in spikes per msec. Suppose that the duration of neurons' waveforms is such that substantially different waveforms are recorded whenever two spike peaks are separated by less than γ msec. Then the probability that neuron i spikes when the covariate is c is $\lambda_i^*(c)$, and the probabilities that this spike is contaminated or not by a spike from the other neuron are $2\gamma\lambda_j^*(c)$ and $1 - 2\gamma\lambda_j^*(c)$, $j \neq i$, respectively. Therefore, if we assume that neurons spike independently, the probability $\pi_{10}(c)$ that neuron 1 spikes alone is proportional to $\lambda_1^*(c)[1 - 2\gamma\lambda_2^*(c)]$. Listing all neuron combinations gives

$$\pi_x^*(c) \propto \begin{cases} \lambda_1^*(c)[1 - 2\gamma\lambda_2^*(c)] & \text{if } x = (1, 0) \\ \lambda_2^*(c)[1 - 2\gamma\lambda_1^*(c)] & \text{if } x = (0, 1) \\ 2\gamma\lambda_1^*(c)\lambda_2^*(c) & \text{if } x = (1, 1), \end{cases} \tag{2.3}$$

where $\propto$ stands for "proportional to." The proportionality constant is such that the probabilities sum to one. This constant is not one, because we do not consider the combination $x = (0, 0)$ when a spike is recorded at the electrode. To illustrate equation 2.3, suppose that both neurons spike independently according to Poisson processes with constant rates 100 Hertz. Suppose also that waveforms are 2 msec long and that a single, substantially different, waveform is recorded whenever two spikes overlap by 0.5 msec or more (i.e., whenever two spike peaks are within $\gamma = 1.5$ msec). Consider a 1 second interval in which neuron 1 happens to fire 100 times. Then the set of intervals formed by taking times that lie within 1.5 msec of the peak of a spike from neuron 1 occupies a total of 300 msec. Thus, on average, 300 of 1000 of the spikes from neuron 2 will overlap with spikes from neuron 1. A typical case would thus have 30 overlaps and 70 clean spikes from each of the two neurons. These numbers match equation 2.3; indeed, $\lambda_1^*(c)[1 - 2\gamma\lambda_2^*(c)] = \lambda_2^*(c)[1 - 2\gamma\lambda_1^*(c)] = 0.1[1 - 2 \times 1.5 \times 0.1] = 0.07$ and $2\gamma\lambda_1^*(c)\lambda_2^*(c) = 2 \times 1.5 \times 0.1 \times 0.1 = 0.03$, which gives the correct ratio of single and joint spikes.

Generalizing to I independent neurons, we obtain

$$\pi_x^*(c) = \frac{1}{\kappa^*(c)} \prod_{i=1}^{I} (2\gamma)^{-1} [2\gamma\lambda_i^*(c)]^{x_i} [1 - 2\gamma\lambda_i^*(c)]^{(1-x_i)}, \tag{2.4}$$

where x_i are the I components of x, and $\kappa^*(c)$ is such that $\sum_{x \in \mathcal{X}} \pi_x^*(c) = 1$, which yields

$$\kappa^*(c) = (2\gamma)^{-1} \left(1 - \prod_{i=1}^{I} (1 - 2\gamma\lambda_i^*(c))\right). \tag{2.5}$$

Equation 2.5 is also one minus the probability that no neuron spikes in a time bin of duration 2γ, rescaled by 2γ. It is therefore the probability of observing a spike at the electrode when the covariate has value c, in units of spikes per msec. That is, $\kappa^*(c)$ is the electrode's firing rate.

Putting this together, the distribution of WFs conditional on c is

$$f^*(a \mid c) = \sum_{x \in \mathcal{X}} \pi_x^*(c) f_x^*(a), \tag{2.6}$$

where $\pi_x^*(c)$ are defined by equation 2.4 and 2.5, and $f_x^*(a)$ are the same WFs' distributions we used in equation 2.1. Equation 2.6 specifies a different mixture distribution for the WFs for each value of c. In contrast, equation 2.1 describes the distribution of the WFs if we ignore the information provided by c. If neurons are not tuned to c, then equation 2.6 reduces to equation 2.1 mathematically and logically. Note also that the integral of $\pi_x^*(c)$ over the observed values of c is equal to π_x^* in equation 2.1, which will be useful to compare spike-sorting schemes.

Just as in the previous section, automatic spike sorting is carried out by using Bayes' rule to calculate the probability that a spike with WF a was produced by neuron combination x, given that the covariate has value c,

$$P^*(x \mid a, c) = \frac{\pi_x^*(c) f_x^*(a)}{f^*(a \mid c)}, \tag{2.7}$$

and the spike is allocated to the neurons in $x^\dagger$ such that

$$x^\dagger(a \mid c) = \arg\max_x P^*(x \mid a, c),$$

with corresponding allocation for neuron $i = 1, \ldots, I$ the ith component $x_i^\dagger(a \mid c)$.

As with traditional spike sorters, perfect spike classification is rare. However, the proposed spike sorter incorporates all the information available about spike identities, and it is derived from the optimal Bayes' rule in equation 2.7, so it has the lowest spike misclassification rate of all classification rules. In particular, it has a lower spike misclassification rate than the traditional automatic spike sorter described in section 2.1.

2.2.1 Special Case: Classic and Proposed Approaches Are Equivalent When Waveform Features Clusters Are Perfectly Separated. When WFs clusters are perfectly separated, a spike with WF a has $f_x^*(a) = 0$ for all $x \in \mathcal{X}$, except for the particular x that produced the spike. Hence equations 2.2 and 2.7 both equal one if the spike was produced by neuron combination x, and zero otherwise. This shows that the new spike-sorting rule does not depend on the covariate c, so it reduces to the traditional spike-sorting rule. Tuning function estimates will thus be the same under both approaches, given the same tuning model.

2.2.2 Automatic Spike Sorting Based on Tuning Information Only. Current spike-sorting methods use waveform information and ignore tuning information. The reverse could be done, as in Ventura (2008). To do that, we assume that f_x^*, $x \in \mathcal{X}$, are all equal, so that they carry no discriminating information about waveforms. Then equation 2.7 becomes

$$P^*(x \mid c) = \pi_x^*(c), \tag{2.8}$$

which implies that a spike recorded when the covariate takes value c is allocated to the combination of neurons that has the highest probability of spiking when the covariate is c, regardless of the spike waveform measurements. To illustrate this, consider the introductory example once more: an electrode records two neurons whose tuning curves are such that $\lambda_1^*(c) = 0$ except when $c_1 < c < c_2$, and $\lambda_2^*(c) = 0$ except when $c_2 < c < c_3$. Imagine that a spike is recorded at the electrode when c takes some value $c_0 \in [c_2, c_3]$. Then using equations 2.4 and 2.8, we calculate $P^*(x = 10 \mid c_0) \propto 0 \times [1 - \lambda_2^*(c_0)] = 0$, $P^*(x = 01 \mid c_0) \propto \lambda_2(c_0) \times [1 - 0] > 0$, and $P^*(x = 11 \mid c_0) \propto 0 \times \lambda_2^*(c_0) = 0$. The value of x that maximizes equation 2.8 is therefore $x^\dagger(c_0) = 01$, so that the spike is allocated to neuron 2, the correct decision. Figure 3 provides another example of spike sorting based only on tuning information.

2.3 Estimation of the Waveform-Based Spike-Sorting Rule. So far we have used the true proportions π_x^*, distributions f_x^*, and tuning curves λ_i^*. In practice, these are unknown and must be estimated from data.

We approximate $f_x^*(a)$ by $f_x(a; \psi_x)$ indexed by parameters ψ_x, for example, normal distributions with means and variance-covariance matrices $\psi_x = (\mu_x, \Sigma_x)$. We approximate $\lambda_i^*(c)$ by functions $\lambda_i(c; \theta_i)$ that depend on parameters θ_i. Although the f^*s and λ^*s play parallel roles for spike sorting, WFs are qualitatively invariant to the experiment that produce them, and models for their distributions have long been evaluated. In contrast, neurons' tuning properties are not only experiment specific, they are typically of paramount interest in practice. Therefore, while it is reasonable to use parametric models for f_x, λ_i should be nonparametric functions unless more information is available. (This point is discussed further in section 3.) We denote by $\Theta = (\theta_i, i = 1, \ldots I)$, $\Pi = (\pi_x, x \in \mathcal{X})$, and $\Psi = (\psi_x, x \in \mathcal{X})$ the combined vectors of parameters. We estimate Θ, Π, and Ψ by the method of maximum likelihood (ML), because it makes the most efficient use of data (Kass, Ventura, & Brown, 2005).

The ML estimate of $(\Pi, \Psi) = (\pi_x, \psi_x, x \in \mathcal{X})$ needed to estimate equation 2.1 is the value that maximizes the likelihood function $L(\Pi, \Psi)$, defined as the joint distribution of the observed data. The data consist of the WFs **a**, so

$$L(\Pi, \Psi) = f(\mathbf{a}; \Pi, \Psi). \tag{2.9}$$

If we assume that WFs a_t are independent, equation 2.9 reduces to the product over time bins of $f(a_t; \Pi, \Psi)$, the model we chose to estimate equation 2.1, evaluated at a_t. Likelihoods that arise from mixtures such as equation 2.1 are well known to be difficult to optimize, and a latent variable approach is often preferred. We use as latent variables the neuron combinations $x_t = (x_{1t}, \ldots, x_{It}) \in \mathcal{X}$ that could have produced the spike at time t, and use an expectation-maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977) to maximize the log of $L(\Pi, \Psi)$. This is a classic iterative algorithm, which we now give in its general form. Suppose that $(\Pi^{(k)}, \Psi^{(k)})$ are the current parameter values and that we want to update them to $(\Pi^{(k+1)}, \Psi^{(k+1)})$, with the eventual aim of reaching the ML estimator $(\widehat{\Pi}, \widehat{\Psi})$. The EM algorithm obtains $(\widehat{\Pi}, \widehat{\Psi})$ by iteratively:

1. (**E-step**): Calculating

$$Q\big(\Pi, \Psi, \Pi^{(k)}, \Psi^{(k)}\big) = E\big(\log f(\mathbf{a}, \mathbf{X}; \Pi, \Psi) \mid \mathbf{a}; \Pi^{(k)}, \Psi^{(k)}\big), \tag{2.10}$$

   the expectation of the augmented data log likelihood $\log f(\mathbf{a}, \mathbf{x}; \Pi, \Psi)$ with respect to the distribution of the latent variables **X** given the observed data **a**, at current estimates $(\Pi^{(k)}, \Psi^{(k)})$
2. (**M-step**): Maximizing $Q(\Pi, \Psi, \Pi^{(k)}, \Psi^{(k)})$ with respect to (Π, Ψ) to obtain update parameter values $(\Pi^{(k+1)}, \Psi^{(k+1)})$

Mixtures of normal distributions are often used for $f(a; \Pi, \Psi)$, although Harris et al. (2000) and Shoham et al. (2003) provide convincing evidence that t-distributions might be are more appropriate. We use univariate normal distributions throughout this letter for algorithmic simplicity. It would be straightforward to use instead t or other distributions deemed more appropriate, provided an EM algorithm existed or could be developed to estimate their parameters. For a mixture of univariate normal distributions, the general EM algorithm given above reduces to:

0. Pick starting values $(\Pi^{(0)}, \Psi^{(0)})$. Repeat steps 1 and 2 until convergence.
1. (**E-step**): For each recorded WF a_t, with associated latent variable X_t, calculate the responsibilities of a_t to each of the mixture components,

$$w_{xt} = P\big(X_t = x \mid a_t, \Pi^{(k)}, \Psi^{(k)}\big) = \frac{\pi_x^{(k)} f_x\big(a_t; \psi_x^{(k)}\big)}{f\big(a_t; \Pi^{(k)}, \Psi^{(k)}\big)}, \quad x \in \mathcal{X},$$

   which is the evaluation of equation 2.2 at $(\Pi, \Psi) = (\Pi^{(k)}, \Psi^{(k)})$.
2. (**M-step**): Update the parameter estimates: for $x \in \mathcal{X}$, calculate

$$\mu_x^{(k+1)} = \sum_t w_{xt} a_t \Big/ \sum_t w_{xt}$$

$$\big(\sigma_x^2\big)^{(k+1)} = \sum_t w_{xt}\big(a_t - \mu_x^{(k+1)}\big)^2 \Big/ \sum_t w_{xt}$$

$$\pi_x^{(k+1)} = P(X_t = x) = \sum_t w_{xt}/N,$$

   where the summations are over time bins that contain a spike and N is the total number of spikes.

References for the normal mixture EM are numerous (see e.g., Hastie, Tibshirani, & Friedman, 2001, sec. 8.5). The case of a mixture of t-distributions was treated by Shoham et al. (2003) in the context of spike sorting. Figures 2A and 8A show step-by-step runs of this algorithm applied to two simulated data sets.

2.4 Estimation of the Waveform and Tuning-Based Spike-Sorting Rule. Estimation of the proposed spike classification rule involves estimation of Θ and Ψ, the parameters that index the model for equation 2.6. The ML estimates of (Θ, Ψ) maximize the joint distribution of the data,

$$L(\Theta, \Psi) = f(\mathbf{z}, \mathbf{a} \mid \mathbf{c}; \Theta, \Psi),$$

where the data consist not only of the WFs $\mathbf{a}$, but also of the electrode spike train $\mathbf{z}$, which contains tuning information. We use the EM algorithm with latent variables $\mathbf{X}$ to maximize $L(\Theta, \Psi)$. Suppose that $\Theta^{(k)}$ and $\Psi^{(k)}$ are the current parameter values and we want to update them to $\Theta^{(k+1)}$ and $\Psi^{(k+1)}$. The EM algorithm obtains the ML estimate of Θ and Ψ by iteratively:

1. (**E-step**): Calculating

$$Q\big(\Theta, \Psi, \Theta^{(k)}, \Psi^{(k)}\big) = E\big(\log f(\mathbf{z}, \mathbf{a}, \mathbf{X} \mid \mathbf{c}; \Theta, \Psi) \mid \mathbf{z}, \mathbf{a}; \Theta^{(k)}, \Psi^{(k)}\big), \tag{2.11}$$

   the expectation of the augmented data log likelihood with respect to the distribution of the latent variables $\mathbf{X}$ given the data $(\mathbf{z}, \mathbf{a})$, at current estimates $\Theta^{(k)}$ and $\Psi^{(k)}$.
2. (**M-step**): Maximizing $Q(\Theta, \Psi, \Theta^{(k)}, \Psi^{(k)})$ with respect to Θ and Ψ to obtain parameter updates $\Theta^{(k+1)}$ and $\Psi^{(k+1)}$.

This algorithm is general and has the same form as the algorithm given in the previous section.

Next we show that this algorithm reduces to two intertwined EM algorithms, one of which is a basic extension of the algorithm in the previous section and the other a basic extension of the EM algorithm in Ventura (2008). We first use basic laws of probability to rewrite the augmented data likelihood as

$$f(\mathbf{z}, \mathbf{a}, \mathbf{x} \mid \mathbf{c}; \Theta, \Psi) = f(\mathbf{a} \mid \mathbf{z}, \mathbf{x}, \mathbf{c}; \Theta, \Psi)\ f(\mathbf{z}, \mathbf{x} \mid \mathbf{c}; \Theta, \Psi).$$

The second term reduces to $f(\mathbf{z}, \mathbf{x} \mid \mathbf{c}; \Theta)$ since without knowledge of $\mathbf{a}$, the distributions of the spike trains $\mathbf{z}$ and $\mathbf{x}$ depend on only Θ and the covariates $\mathbf{c}$. The first term reduces to $f(\mathbf{a} \mid \mathbf{x}, \mathbf{c}; \Theta, \Psi)$ since knowing $\mathbf{x}$ completely specifies $\mathbf{z}$; indeed, for all t, $z_t = 1$ if and only if $x_{it} = 1$ for some i. Now given a spike generated by neuron combination x, its WF a has distribution $f_x(a; \psi_x)$ independent of spike rate parameters Θ. Hence, $f(\mathbf{a} \mid \mathbf{x}, \mathbf{c}; \Theta, \Psi)$ reduces to $f(\mathbf{a} \mid \mathbf{x}, \Psi)$. For practical reasons, we rewrite $f(\mathbf{a} \mid \mathbf{x}, \Psi) = f(\mathbf{a}, \mathbf{x}; \Psi)/f(\mathbf{x}; \Psi)$, where $f(\mathbf{x}; \Psi)$ reduces to $f(\mathbf{x})$ since without knowledge of $\mathbf{a}$, spike train probabilities do not depend on Ψ. Putting this together gives

$$f(\mathbf{z}, \mathbf{a}, \mathbf{x} \mid \mathbf{c}; \Theta, \Psi) = f(\mathbf{a}, \mathbf{x}; \Psi) f(\mathbf{z}, \mathbf{x} \mid \mathbf{c}; \Theta)/f(\mathbf{x}),$$

which we use to rewrite equation 2.11 as

$$\begin{aligned} Q\big(\Theta, \Psi, \Theta^{(k)}, \Psi^{(k)}\big) = {} & Q_1\big(\Psi, \Theta^{(k)}, \Psi^{(k)}\big) + Q_2\big(\Theta, \Theta^{(k)}, \Psi^{(k)}\big) \\ & + Q_0\big(\Theta^{(k)}, \Psi^{(k)}\big), \end{aligned} \tag{2.12}$$

where

$$Q_1\big(\Psi, \Theta^{(k)}, \Psi^{(k)}\big) = E\big(\log f(\mathbf{a}, \mathbf{X}; \Psi) \mid \mathbf{z}, \mathbf{a}; \Theta^{(k)}, \Psi^{(k)}\big), \tag{2.13}$$

$$Q_2\big(\Theta, \Theta^{(k)}, \Psi^{(k)}\big) = E\big(\log f(\mathbf{z}, \mathbf{X} \mid \mathbf{c}; \Theta) \mid \mathbf{z}, \mathbf{a}; \Theta^{(k)}, \Psi^{(k)}\big), \tag{2.14}$$

and $Q_0(\Theta^{(k)}, \Psi^{(k)}) = -E(\log f(\mathbf{X}) \mid \mathbf{z}, \mathbf{a}; \Theta^{(k)}, \Psi^{(k)})$. The decomposition in equation 2.12 suggests that the EM algorithm given above can be reduced to iteratively (1) calculating and maximizing equation 2.13 with respect to Ψ, which is similar to the EM algorithm of section 2.3, and (2) calculating and maximizing equation 2.14 with respect to Θ, which is similar to the EM algorithm for mixtures of spike trains developed in Ventura (2008). The term Q_0 is a constant with respect to Ψ and Θ, so does not contribute to finding the maximum likelihood. The decomposition of the EM algorithm in two partial steps will yield the ML estimates for Ψ and Θ. Indeed, it is a special case of the partial M-step approach justified by the variational formulation of Hathaway (1986) and Neal and Hinton (1998).

The only additional element we need to completely specify the algorithm is the distribution of the latent variables $\mathbf{X}$ given data $\mathbf{z}$, $\mathbf{a}$, covariates $\mathbf{c}$, and parameters Θ, Ψ. Assuming that at time t, the WF a_t and neuron spiking probabilities do not depend on the past, we simplify the conditional joint distribution to the product of the marginal distributions over time bins, that is, $P(\mathbf{x} \mid \mathbf{z}, \mathbf{a}, \mathbf{c}; \Theta, \Psi) = \prod_{t=1}^{T} P(x_t \mid z_t, a_t, c_t; \Theta, \Psi)$. We first treat the trivial case: given $z_t = 0$ (no spike at t), then $x_t = 0$ (no neuron spiked) with probability one. Given $z_t = 1$, the probability that $x_t = 0$ is zero. Otherwise,

$$P(x_t \mid a_t, z_t = 1, c_t; \Theta, \Psi) = \frac{\pi_{x_t}(c_t; \Theta) f_{x_t}(a_t; \psi_{x_t})}{f(a_t, z_t = 1 \mid c_t, \Theta, \Psi)},$$

which is the probability specified in equation 2.7 for $x = x_t$.[2]

We now have all the elements to run the EM algorithm to find the ML estimate of (Ψ, Θ), which in turn provides estimates of $f^*(a \mid c)$ in equation 2.6 and of the neurons' tuning curves $\lambda_i^*(c)$. In the particular case when $f^*(a \mid c)$ is assumed to be a mixture of univariate normal distributions, this algorithm to reduces to:

0—Pick starting values $\Psi^{(0)}$ and $\Theta^{(0)}$. Repeat steps 1 and 2 until convergence.

1—**Update** Ψ

1a. (**E-step**): For each WF a_t, with associated latent variable X_t, calculate the responsibilities of a_t to each of the WF mixture components,

$$w_{xt} = P\big(X_t = x \mid a_t, z_t = 1, c_t; \Theta^{(k)}, \Psi^{(k)}\big), \quad x \in \mathcal{X},$$

by evaluating the estimation model for equation 2.7 at $(\Theta^{(k)}, \Psi^{(k)})$.

[2]Equation 2.7 misses the explicit condition $z_t = 1$. However, this condition was implicit since then we considered only time bins such that $z_t = 1$

1b. (**M-step**): Update $\Psi^{(k)}$ to $\Psi^{(k+1)}$: for $x \in \mathcal{X}$, calculate

$$\mu_x^{(k+1)} = \sum_t w_{xt}\, a_t \Big/ \sum_t w_{xt}$$

$$\left(\sigma_x^2\right)^{(k+1)} = \sum_t w_{xt}\left(a_t - \mu_x^{(k+1)}\right)^2 \Big/ \sum_t w_{xt},$$

where the sums are over the time bins that contain a spike ($z_t = 1$).

2—**Update** Θ

2a. (**E-step**): For each spike $z_t = 1$, with associated latent variable X_t, calculate the responsibilities of z_t to each of the I neurons,

$$\begin{aligned} e_{it} &= E\left(Y_{it} \mid a_t, z_t = 1, c_t; \Theta^{(k)}, \Psi^{(k+1)}\right) \\ &= P\left(Y_{it} = 1 \mid a_t, z_t = 1, c_t; \Theta^{(k)}, \Psi^{(k+1)}\right) \\ &= \sum_{x\, s.t.\, x_i = 1} P\left(X_t = x \mid a_t, z_t = 1, c_t; \Theta^{(k)}, \Psi^{(k+1)}\right), \end{aligned}$$

with probabilities in the summand given by the estimation model for equation 2.7, and summation over the 2^{I-1} values of $x = (x_1, \ldots, x_I)$ that have $x_i = 1$. Given $z_t = 0$, we have trivially

$$e_{it} = E(Y_{it} \mid a_t, z_t = 0, c_t; \Theta^{(k)}, \Psi^{(k+1)}) = 0.$$

2b. (**M-step**): For $i = 1, \ldots, I$, regress e_{it} on c_t, $t = 1, \ldots, T$ to obtain $\theta_i^{(k+1)}$, the parameter of the tuning curve $\lambda_i(c, \theta_i)$.

To distinguish this algorithm from the EM algorithm of the previous section, we refer to it as the linked EM algorithm, since it merges two partial EM algorithms. Note that steps 1 and 2 can be executed in any order. The particular form of step 1 assumes that waveforms are normally distributed. Distributions other than normal can be used provided an EM algorithm exists to estimate their parameters. Step 2 assumes that spike trains are Poisson. Alternatives are discussed in section 4.

Figures 2B, 8B, and 12 show step-by-step runs of the linked EM algorithm and also illustrate that neurons' tuning functions can be estimated before the data are spike-sorted.

2.4.1 Special Case: Perfectly Separated Waveform Features Clusters. As mentioned earlier, traditional and proposed approaches yield the same spike-sorted data in theory when WFs clusters are separated. This remains true in practice, provided the WFs' distributions f_x are not too badly misspecified. Indeed in that case, after convergence of the EM algorithms of sections 2.3 and 2.4, the responsibility of WF a_t to each of the mixture components is $w_{xt} = 1$ if a_t was produced by neuron combination x, and $w_{xt} = 0$ otherwise. That is, both spike-sorting rules correctly classify all spikes. The linked EM algorithm also provides estimates of the tuning functions, which are the same estimates we would obtain with the traditional approach. Indeed, at convergence, the responsibilities of z_t to each of the I neurons are $e_{it} = 1$ if the spike at t was produced either by neuron i alone or jointly with other neurons and $e_{it} = 0$ otherwise. Hence the M-step 2b consists of the regressions of the true neurons' spike trains on c.

2.4.2 Spike Sorting Based on Only Tuning Information. To spike-sort the data using only tuning information, all we need is the ML estimate of Θ, which in turn gives an estimate of the spike classification rule in equation 2.8. To do that, we use the linked EM algorithm, but we set the initial values $\psi_x^{(0)}$ to the same constant for all $x \in \mathcal{X}$, we let $\Psi^{(k)} = \Psi^{(0)}$ for all iterations k, and we run only step 2 of the algorithm. The linked EM algorithm effectively reduces to the algorithm of Ventura (2008). Figures 3 and 11B provide illustrations.

2.5 A Clutter Cluster Serves to Collect Noise and Outlying Spikes. For automatic spike sorting, WFs a_t are typically assumed to originate from one of several components, which implicitly correspond to different neurons. To explicitly allow for neurons spiking together, we instead assumed that a_t originates from one of $\text{card}(\mathcal{X}) = 2^I - 1$ components, each corresponding to a combination $x \in \mathcal{X}$ of neurons spiking approximately together to produce a spike. This choice is important to identify not only joint spikes but also individual neurons' spikes in the low-SNR case, when spikes are contaminated by noise, as we illustrate in section 3.3. But this choice also means that the numbers of probabilities π_x^* in equation 2.1 and distributions $f_x^*(a)$ in equations 2.1 and 2.6 increase exponentially with I. Some of these quantities cannot be estimated accurately with a finite amount of data. For example, suppose that an electrode records $I = 4$ neurons, all firing with a homogeneous Poisson rate of 100 Hz, and that a single, substantially different, waveform is recorded whenever two spike peaks are separated by less than $\gamma = 2$ msec. Then 1 second of data would typically contain 51 clean spikes from each of the $I = 4$ neurons, 13 overlaps from each of the $\binom{I}{2} = 6$ pairs of neurons, 3 overlaps from each of the $\binom{I}{3} = 4$ triplets of neurons, and no occurrence of the 4 neurons spiking approximately together. While this provides enough data to estimate the π_x^* and $f_x^*(a)$ that correspond to single neuron spikes, the estimates of these components for $x \in \mathcal{X}$ such that $\sum_i x_i \geq 2$ would be at best too variable to be useful.

A standard solution in classic spike sorting is to use a noise collection cluster that captures outlier waveforms produced by noise or joint spikes (Sahani, 1999; Shoham et al., 2003). Hence in practice, equation 2.1 is typically estimated by the mixture

$$f(a; \Pi_1, \Psi_1) = \sum_{x \in \mathcal{X}_1} \pi_x f_x(a; \psi_x) + \left(1 - \sum_{x \in \mathcal{X}_1} \pi_x\right) f_{clutter}(a; \psi_0), \tag{2.15}$$

where $\mathcal{X}_1$ contains the neuron combinations x that produce enough spikes to allow estimation of the corresponding f_x, $f_{clutter}$ is the garbage collector distribution used to estimate f_x^* for all $x \in \mathcal{X} \setminus \mathcal{X}_1$, and $\Pi_1 = (\pi_x, x \in \mathcal{X}_1)$ and $\Psi_1 = (\psi_0, \psi_x, x \in \mathcal{X}_1)$ are the reduced sets of parameters to be estimated. The set $\mathcal{X}_1$ is not known a priori but is estimated jointly with equation 2.1. The simplest method to do that is penalized likelihood, which has as special cases the Akaike's information criterion (AIC; Akaike, 1974) and the Bayesian information criterion (BIC; Schwartz, 1978): competing models are fitted by maximum likelihood, a score of the form "goodness of fit" minus "model complexity" is assigned to each model, and the model with the highest score is selected. The competing models considered here are equation 2.15 for different sets $\mathcal{X}_1$, or rather for different sizes of $\mathcal{X}_1$. Indeed, card$(\mathcal{X}_1) = 3$ means that three "arbitrary" components $\pi_x f_x$ are fitted to the observed waveforms, where x presumably, although not certainly, correspond to single neurons rather than to combinations of neurons. Hence the chosen model typically has $\mathcal{X}_1 = (x \in \mathcal{X} : \sum_i x_i = 1)$, while $f_{clutter}$ covers the waveforms from joint spikes and noise.

We adopt the same approach with the proposed spike sorter. In practice, we estimate equation 2.6 with the mixture

$$f(a; c, \Psi_2, \Theta) = \sum_{x \in \mathcal{X}_2} \pi_x(c; \Theta) f_x(a; \psi_x) + \left(1 - \sum_{x \in \mathcal{X}_2} \pi_x(c; \Theta)\right) f_{clutter}(a; \psi_0), \tag{2.16}$$

where $\Psi_2 = (\psi_0, \psi_x, x \in \mathcal{X}_2)$ is the reduced set of parameters for WF distributions, while Θ remains the full set of tuning function parameters. Again, $\mathcal{X}_2$ is determined by penalized likelihood. Because $\pi_x(c; \Theta)$ depends on tuning functions, we must specify the exact forms of the sets $\mathcal{X}_2$ we wish to consider rather than just their sizes, as was the case for $\mathcal{X}_1$. Indeed, while equation 2.15 with card$(\mathcal{X}_1) = 3$ corresponds to a unique model, equation 2.16 with card$(\mathcal{X}_2) = 3$ could be either the model with clusters $\pi_x f_x$, $x = (1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$ or one of the $\binom{3}{2} = 3$ models with any two of the previous clusters and their "cross" cluster, for example, $x = (1, 0, 0), (0, 1, 0)$, and $x = (1, 1, 0)$. While in principle we could also consider models composed only of "higher-order" clusters, for example, $x = (1, 1, 0)$, $(0, 1, 1)$, and $x = (1, 1, 1)$, such models imply that the largest WF clusters arise from joint rather than from single neuron spikes, and thus are unlikely to compare well with more sensible models. Therefore, in searching for the best model, we restrict our attention to models with components $\pi_x f_x$ such that $\sum_i x_i \leq 2$.

Because equation 2.16 explicitly models through $\pi_x(c; \Theta)$ the probabilities of several neurons spiking together, joint spikes' clusters should be more easily identified. Hence, it seems likely that the chosen model will be such that $\mathcal{X}_1 \subset \mathcal{X}_2$, and that the components in $\mathcal{X}_2 \setminus \mathcal{X}_1$ will correspond to clusters of joint spikes. This is illustrated in section 3.3.

2.6 Implementation Details. As with all EM algorithms, the issues of choosing initial values, determining the number of neurons, and avoiding convergence to local likelihood maxima must be addressed. These issues have been the subject of extensive research in the statistics community. (For recent reviews, see Sahani, 1999; McLachlan & Peel, 2000; Figueiredo & Jain, 2002.) Although practically important, these issues are not specific to the EM algorithm developed here, so we kept their implementation to a minimum to avoid cluttering the main ideas. Basically, we chose reasonable initial values and determined the number of neurons using a penalized likelihood approach, as described below. We did not implement any fancy parameter space search to avoid local likelihood maxima. However, we give evidence in section 3.6 that including tuning information for spike sorting helps determine the number of neurons and facilitates the convergence of the algorithm to the global maximum.

Throughout this letter, we assume without loss of generality that the waveforms' first PC distributions $f_x(a; \psi_x)$, $x \in \mathcal{X}$, are univariate normal distributions with means and variances (μ_x, σ_x^2). To fit I neurons to an electrode, we choose initial values $\mu_x^{(0)}$ and $\sigma_x^{(0)}$ as follows. For the I combinations $x \in \mathcal{X}$ corresponding to single neurons ($\sum_i x_i = 1$), we simulate $\mu_x^{(0)}$ at random between the $\frac{100(i-1)+10}{I}$th and the $\frac{100i-10}{I}$th sample quantiles of a_t. For example, when $I = 2$, we simulate $\mu_{01}^{(0)}$ at random between the 5th and 45th quantiles of a_t and $\mu_{10}^{(0)}$ between the 55th and 95th quantiles. This guarantees that the $\mu_x^{(0)}$ are random but well spread out. We simulate $\sigma_x^{(0)}$ at random between $S/(I+2)$ and S/I, where S is the sample standard deviation of a_t, so that the initial PC distributions do not overlap much. For $x \in \mathcal{X}$ that correspond to joint spikes, we let $\mu_x^{(0)}$ be the sample mean of a_t and simulate $\sigma_x^{(0)}$ at random between $90S$ and $100S$. This is akin to using very spread uniform distributions and reflects our lack of knowledge about where joint spike clusters might be. We then simulate the $\pi_x^{(0)}$ that correspond to single neurons at random between 30% and 70%, and take the $\pi_x^{(0)}$ that correspond to joint spikes to be the product of the corresponding single neuron probabilities. Finally we standardize the $\pi_x^{(0)}$ so they sum to one. As for initial values for λ_i to run the linked EM algorithm, we take them to be equal to the constants $\pi_x^{(0)}$ for the I values of x such that $\sum_i x_i = 1$.

3 Results

Traditional automatic spike sorting is based on describing waveform features (WFs) with a mixture of distributions,

$$f(a; \Pi, \Psi) = \sum_{x \in \mathcal{X}} \pi_x f_x(a; \psi_x), \tag{3.1}$$

where π_x is the proportion of spikes generated by neuron combination x and f_x is the probability distribution of their WFs, indexed by a parameter

ψ_x. This description ignores the fact that the probability of observing a spike depends on the covariates c that modulate neurons' firing rates. The proposed automatic spike sorter rectifies this by letting the distribution of WFs depend on c,

$$f(a \mid c; \Theta, \Psi) = \sum_{x \in \mathcal{X}} \pi_x(c; \Theta) f_x(a; \psi_x), \tag{3.2}$$

where f_x is as in equation 3.1, but $\pi_x(c; \Theta)$ is now the covariate varying proportion of spikes generated by neuron combination x when the covariate takes value c. It is a function of the neurons' tuning functions $\lambda_i(c; \theta_i)$, as specified by equation 2.4. Equation 3.2 yields a different WF distribution for different values of c, whereas equation 3.1 describes the WF distribution when the values of c are "lost" or unavailable. The automatic spike sorter based on equation 3.2 is necessarily superior to the traditional spike sorter because it uses more information. Additionally, equation 3.2 explicitly models through $\pi_x(c; \Theta)$ the probabilities of several neurons spiking together as functions of the neurons' firing rates, which should facilitate the identification of joint spikes.

But the new spike sorter seems to put the cart before the horse: How can tuning functions be used before the data are spike-sorted? The answer lies with the linked EM algorithm (see section 2.4) used to estimate equation 3.2: given the spike times and waveform measurements, the algorithm is designed to provide estimates of the neurons' tuning functions, just as the traditional EM algorithm (see section 2.3) is designed to provide estimates of the waveform distributions f_x in equation 3.1. Figures 2, 8, and 12 illustrate this and also illustrate that the proposed method reduces spike misclassification rates, helps identify joint spikes, and improves the convergence of the EM algorithm.

To avoid the inherent uncertainty in determining the true spike identities in extracellular recordings, we use simulated data, and to produce easily readable graphics, we use electrodes that record only two neurons. Although in real neural data, there is typically significant information for spike sorting in up to three or four of the PCs, we use only one PC to facilitate the visual comparison between spike sorters. This choice implies no loss of generality. Indeed, both spike sorters use the same information from waveforms, so it is fair to compare their performances. In practice more PCs are used to reduce WF overlap. Section 3.5 illustrates how the spike misclassification rates vary with cluster overlap. Finally, we assume that waveforms' PCs are normally distributed and that neurons spike according to inhomogeneous Poisson processes.

3.1 Motor Cortex Example. This example is inspired by movement decoding experiments in primary motor cortex (M1). Say that $I = 2$ simulated M1 neurons are recorded by the same electrode while a monkey

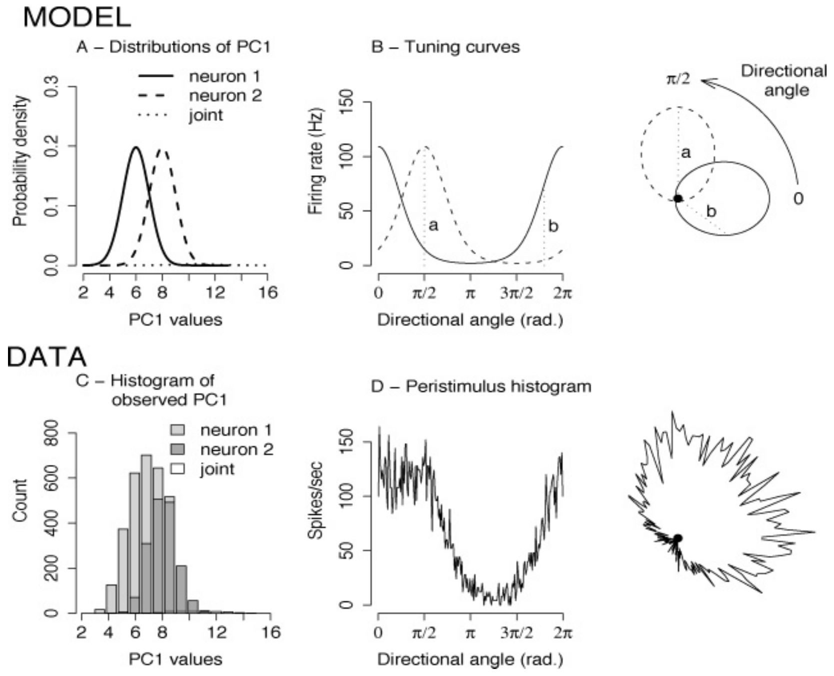


Figure 1: Motor cortex example. (A, B) Simulation model and (C, D) data from the model. (A) True distributions of spike waveforms first PC, $f^*_x(a), x \in \mathcal{X}$, weighted by the probabilities of their occurrences, π^*_x. (B) Neurons' true tuning curves $\lambda^*_i(d) = \exp(2.7 + 2\cos(d - d_i)), i = 1, 2$, as functions of directional tuning d, in Cartesian and circular coordinates. The neurons' preferred directions are $d_1 = 0$ and $d_2 = \pi/2$. (C) Color-coded histogram of spikes first PC collected at the electrode. The shades of gray indicate the true identities of the spikes. (D) Peristimulus histogram of the recorded electrode spike train, in Cartesian and circular coordinates. Traditional spike sorting uses the information in C. The proposed method combines the information in C and D.

traces a 2D constant-speed circular trajectory with his hand. The trajectory can be described by a scalar parameter $d = \arctan(y/x) \in [0, 2\pi]$, which measures the angle of the hand position (x, y). The true tuning curves $\lambda^*_i(d)$ are shown in Figure 1B. By *true*, we refer to the unknown mechanism that generates the spike trains rather than to the model we chose to fit to them. To produce easily readable graphics, we used unrealistically highly modulated tuning curves, with minimum and maximum firing rates approximately 0 and 110 Hz, respectively. Our other examples are realistic. We found that the tuning curves and their estimates appeared visually less cluttered when plotted in a circular coordinates, as displayed in Figure 1B.

With $I = 2$ neurons, $\mathrm{card}(\mathcal{X}) = 2^I - 1 = 3$ types of spikes can be recorded at the electrode, which are the spikes generated by neurons 1 and 2 alone, and their joint spikes. They are coded by $x = (1, 0) \equiv 10$, $x = (0, 1) \equiv 01$, and $x = (1, 1) \equiv 11$, respectively. We assume that the signal-to-noise ratio is large enough so that spikes crossing threshold are real spikes rather than noise. The noise case is treated in section 3.3. Without loss of generality, we use the waveforms' first PCs (PC1) for spike sorting and assume that their true distributions f_x^* are normal with means and variances (6, 1), (8, 1), and (10.5, 3) for $x = 10, 01$, and 11, respectively. These parameters are arbitrary but could be made to match actual data by shifting and rescaling the normal distributions without qualitatively changing the results. Given the tuning curves specified above, the true probabilities of neurons spiking alone and together are $\pi_{10}^* = \pi_{01}^* = 49.5\%$ and $\pi_{11}^* = 1\%$. Figure 1A displays $\pi_x^* f_x^*$ for $x \in \mathcal{X}$. Because π_{11}^* is close to zero, $\pi_{11}^* f_{11}^*$ is hard to distinguish from the x-axis. The waveform clusters $\pi_x^* f_x^*$ overlap, so spikes cannot be sorted with perfect confidence.

Using the model in Figures 1A and 1B, we simulated the electrode spike train during 50 loops of the circular trajectory, letting the neurons spike according to Poisson processes with rates $\lambda_i^*(d)$. A color-coded histogram of the spikes' PCs is in Figure 1C. In practice, the spike identities are unknown, and it is the purpose of spike sorting to determine them. The traditional procedure consists of clustering the data that make the histogram in Figure 1C. The proposed spike sorter also uses tuning information, which we represented by the peristimulus histogram of the spikes in Figure 1D. This plot shows that spiking happens primarily when directional tuning is approximately between $-\pi/4$ and $\pi/2 + \pi/4$, although, just as with the histogram of PC1 in Figure 1C, it is hard to distinguish individual neurons with the naked eye.

Before we proceed with spike sorting, we must select models to fit to the data. To ensure that the results of this illustrative example are not corrupted by possible effects of model inadequacies, we use the correct family of models: $f_x(a; \psi_x), x \in \mathcal{X}$, are taken to be normal distributions with means and variances (μ_x, σ_x^2) and the tuning curves to be of the form $\lambda_i(d; \theta_i) = \exp(\theta_{0i} + \theta_{1i} \cos(d) + \theta_{2i} \sin(d)), i = 1, \ldots, I$. The parameters $\Psi = (\pi_x, \psi_x, x \in \mathcal{X})$ and $\Theta = (\theta_i, i = 1, \ldots, I)$ are estimated using the EM algorithms in sections 2.3 and 2.4, with initial values generated according to section 2.6. The number of neurons recorded by the electrode was determined by penalized likelihood: both AIC and BIC found $I = 2$ neurons, the correct number. Plots from left to right in Figure 2A show $\pi_x^{(k)} f_x(a; \psi_x^{(k)}), k = 0, \ldots, 5$, the initial values and first five iterations of the EM algorithm of section 2.3, and the last panel shows $\hat{\pi}_x f_x(a; \hat{\psi}_x)$, the solution after convergence, for $x = 01$ and $x = 10$. We omitted estimates of $\pi_{11} f_{11}(a; \psi_{11})$ because they cannot be distinguished from the horizontal axis. Figure 2B shows a run of the linked EM algorithm. The same initial values were used in Figures 2A and 2B. The successive plots show

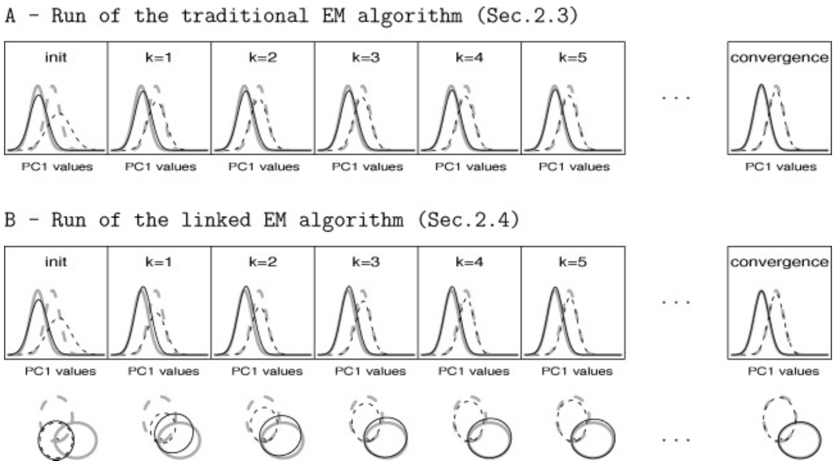


Figure 2: (A) Estimation of $\pi_x f_x(a;\psi_x), x \in \mathcal{X}$ in equation 3.1 by the traditional EM algorithm of section 2.3. The plots from from left to right show the initial values $\pi_x^{(0)} f_x(a;\psi_x^{(0)})$, the first five iterations $\pi_x^{(k)} f_x(a;\psi_x^{(k)})$, and the solutions $\hat{\pi}_x f_x(a;\hat{\psi}_x)$ after convergence. The thick gray curves are the true $\pi_x^* f_x^*(a)$. (B) Joint estimation of $f_x(a;\psi_x), x \in \mathcal{X}$ and $\lambda_i(c;\theta_i), i = 1,\ldots,I$ in equation 3.2 by the linked EM algorithms in section 2.4. The same initial values $f_x(a;\psi_x^{(0)})$ were used in A and B. The initial values $\lambda_i(c;\theta_i^{(0)})$ in B are constant rates equal to the starting values $\pi_x^{(0)}$ used in A. The algorithm in B provides estimates of tuning functions before the data are spike-sorted.

$[\int \pi_x(c;\Theta^{(k)})dc] f_x(a;\psi_x^{(k)})$ for $x = 01$ and $x = 10$, and $\lambda(d;\theta_i^{(k)}), i = 1, 2$, for $k = 0,\ldots,5$, as well as the solution after convergence.[3] This shows clearly that the linked EM algorithm can indeed estimate the tuning curves before the data are spike-sorted.

With Π, Ψ, and Θ estimated, we apply the classification rules in equations 2.2, 2.7, and 2.8 to sort the electrode spike train. Figure 3 shows the outcome for one trial of the simulated data. From inward to outward, the plot shows the neurons' true tuning curves $\lambda_i^*(d)$; the observed electrode spike train for the selected trial, $\mathbf{z}$; the true spike train of neuron 1 we aim to retrieve by spike sorting, $\mathbf{y}_1$; and the spike-sorting classification obtained by using spike waveform information only (see equation 2.2), tuning information only (see equation 2.8), and both sources of information (see equation 2.7). We see that for waveform-based spike sorting, misclassification errors are more numerous in the neurons' preferred directions. This happens because when d is close to the preferred direction of neuron 2 (1), almost

[3]Recall that $\int \pi_x(c;\Theta^{(k)})dc = \pi_x$, the overall proportion of spikes produced by neuron combination x, so that Figures 2A and 2B can be compared.

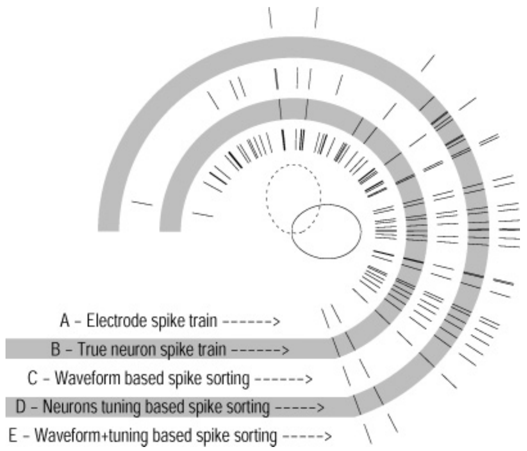


Figure 3: (A) Recorded electrode spike train and (B) true spike train of neuron 1 for one trial of the data of Figure 1. The electrode records $I = 2$ neurons with tuning curves plotted in circular coordinates at the center of the plot. The tuning curve of neuron 1 is the solid curve. (C–E) Estimated spike train of neuron 1 obtained by spike-sorting the electrode spike train using (C) spike waveform information only, (D) timing information only, and (E) both. The latter has only one misclassified spike. Overall, spike sorting in C produced 18% misclassified spikes versus 9% for the proposed method (E).

all spikes recorded at the electrode belong to neuron 2 (1), yet spike sorting classifies them according to waveform information only. On the other hand, tuning-based spike sorting assigns spikes to neurons only when their firing rates are comparatively larger than the firing rates of the other neurons. Although all spike-sorting rules produce errors, combining all sources of information clearly does best. The resulting spike misclassification rates for traditional and proposed approaches are 18% and 9%, respectively, on average across many repeat simulations. Figure 4 shows the same spike identity color-coded histogram as Figure 1C, with misclassified spikes indicated in black.

The proposed spike sorter also helped identify joint spikes: traditional spike sorting (see equation 2.2) retrieved 37% of the actual joint spikes, while the proposed method retrieved 45%. Also for both methods, 90% of all identified joint spikes were actual joint spikes. The noisy channel example of section 3.3 further illustrates the benefit of modeling the probabilities of joint spiking as functions of the firing rates.

3.2 Model Selection for Tuning Curves. The previous example was designed to illustrate the properties of the new spike sorter with easily interpretable graphics, but that example was unrealistic in several ways.

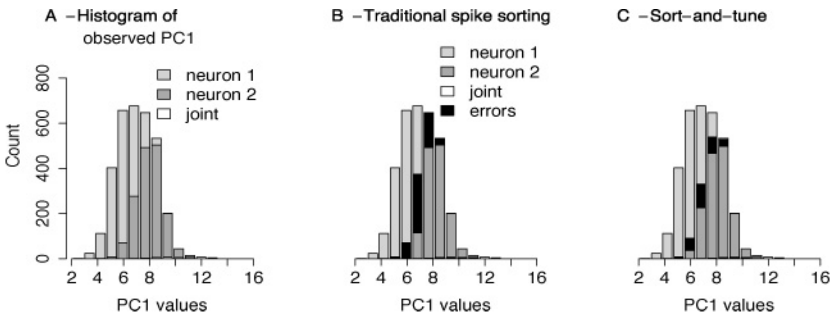


Figure 4: Color-coded histograms of the spike waveforms' first PC. (A) True identities. (B, C) Identities determined by spike sorting using (B) the traditional rule, equation 2.2, and (C) the proposed rule, equation 2.7. Spike classification errors are indicated in black. The spike misclassification rate is 18% in B and 9% in C.

First, M1 neurons are rarely so strongly modulated or exactly cosine tuned, but worse was to assume parametric models for their tuning curves to run the linked EM algorithm. Use of parametric models for WFs is justifiable: normal or t-distributions have long been used successfully, their relative advantages and shortcomings are well documented, and WFs' properties are similar in all experiments. On the other hand, tuning properties are experiment specific and often of primary interest. This example shows how to build nonparametric tuning curve models to run the linked EM algorithm.

Again we consider $I = 2$ simulated M1 neurons, recorded by the same electrode, while a monkey traces a 2D constant-speed circular trajectory. We use the same WF distributions as in the previous example, but we now assume that the true tuning curves $\lambda_i^*(d)$ are double-peaked. They are shown as thick gray curves in Figures 5 and 6. Given these choices, the probabilities that neurons spike alone and together are $\pi_{10}^* = 32\%$, $\pi_{01}^* = 66\%$, and $\pi_{11}^* = 2\%$. We simulated the electrode spike train from this model during 50 loops of the circular trajectory, letting the neurons spike according to Poisson processes with rates $\lambda_i^*(d)$.

To run the linked EM algorithm, we use normal WF distributions and gaussian filters G instead of cosine tuning curves,

$$\begin{aligned}\lambda_i(c = (d, t); \theta_i) = \exp(&\theta_{0i} + \mathrm{G}(\cos(d); \mathrm{bw}_i) + \mathrm{G}(\sin(d); \mathrm{bw}_i)\\ &+ \mathrm{G}(t; \mathrm{bw}_{i0})),\end{aligned} \tag{3.3}$$

where bw are bandwidths that control the amount of smoothing. All equation 3.3 assumes is that the logarithm of the tuning curves is additive in smooth functions of the covariates. We let the data determine which covariates to retain in the model and the degree of smoothness of the filters.

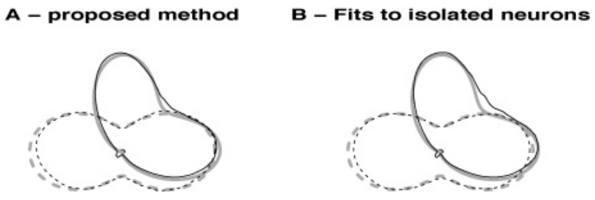


Figure 5: True tuning curves (thick gray curves) and estimates (black curves) obtained by (A) the proposed method and, for comparison, (B) fitting tuning functions to the true neurons' spike trains (no spike sorting needed). The step-by-step run of the linked EM algorithm is shown in the appendix in Figure 12. True $\lambda_i^*(d)$ are of the form $\exp(a_{i1} + b_{i1}\cos(d - d_{i1})) + \exp(a_{i2} + b_{i2}\cos(d - d_{i2})), i = 1, 2$, with $d_{i1} \neq d_{i2}$. The same nonparametric tuning curve model was used in A and B, namely, gaussian filters (see equation 3.3), whose smoothness parameters were determined by cross-validation. B represents the ideal situation since the true neurons' spike trains were used. The estimates in A are as close to the true tuning curves as those in B. This remained true across simulated data (not shown).

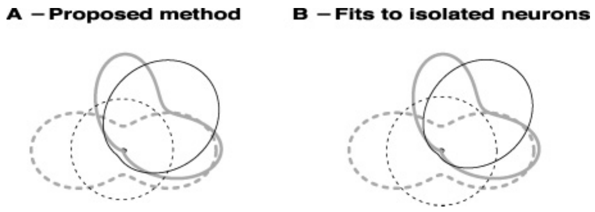


Figure 6: True (thick gray curves) and estimated (black curves) tuning functions obtained by (A) the proposed method and (B) fitting tuning functions to the true neurons' spike trains (no spike sorting needed). The same parametric tuning curve model was used in A and B, namely, an exponential cosine function $\lambda_i(d, \theta_i) = \theta_{0i} + \theta_{1i}\cos(d) + \theta_{2i}\sin(d)$. This model is inappropriate and thus yields biased tuning curve estimates when used within the proposed spike-sorting method. However, tuning curve estimates would be just as biased even if the true neurons' spike trains were available, as shown in B.

In practice, neurons may or may not be tuned to particular covariates, or different neurons might be tuned to different covariates. Here we considered the covariates $\cos(d)$ and $\sin(d)$, which makes sense given the application, and also experimental time t, to capture potential temporal drift effects. To determine which covariates modulate the firing rates, we run the linked EM algorithm and test the statistical significance of each covariate using likelihood ratio (LR) tests (Kass et al., 2005) or model selection criteria like AIC or BIC. That is, we test which components of Θ are significantly different from zero. An LR test thus determined that experimental time did not significantly modulate neurons' spiking (p-value $\ll 0.001$), so we removed t from the model. Although this conclusion is obvious here since we know how the data were generated, the process illustrates how to select appropriate covariates in real applications.

The degrees of smoothness of the gaussian filters were determined from data by cross-validation, a model choice criterion asymptotically equivalent to AIC. Within the family of nonparametric models specified by equation 3.3, the optimal model for neuron 1 (solid curve in Figure 5) uses 12 DFs—1 for the intercept and 5.5 for each of $\cos(d)$ and $\sin(d)$. The optimal model for neuron 2 (dashed curve in Figure 5) uses 1 DF for the intercept and 4.75 for each of $\cos(d)$ and $\sin(d)$. The DFs for $\cos(d)$ and $\sin(d)$ are much larger than one, which strongly suggests that the neurons are not cosine tuned.[4] An LR test comparing cosine and optimal models further confirmed this, as did the K-S goodness-of-fit test of Brown, Barbieri, Ventura, Kass, and Frank (2002) (not shown). In contrast, equation 3.3 fitted to the cosine neurons of section 3.1 yielded DFs close to one, while LR and K-S tests did not reject cosine tuning curves as viable models for those neurons. This shows that common likelihood-based model selection techniques can be applied within the proposed approach to build appropriate tuning curve models.

Note that adding tuning information reduced the spike misclassification rate for neuron 1 from 30.5% to 16.8% and increased that for neuron 2 from 8.4% to 9% on average across simulated samples. Accounting for the two neurons producing 32% and 66% of the spikes, respectively, the overall spike misclassification rate was reduced from 15.5% to 11.5%

Finally, to illustrate the danger of using an inappropriate tuning model, we also fitted exponential cosine curves to the data of Figure 5. The misclassification rate for neuron 1 was reduced from 30.5% to 14%, that for neuron 2 increased from 8.4% to 13.6%, and the overall rate was still reduced from 15.5% to 13.8%. However, the improved rate does not mitigate the severity of the bias in the tuning curve estimates (see Figure 6A). To be fair to our method, we also fitted exponential cosine tuning curves to the true neurons' spike trains. The resulting estimates in Figure 6B are also grossly biased. This shows that assuming an inappropriate tuning curve model has similar consequences regardless of the estimation method.

3.3 Noisy Channel Example. In classic single-electrode electrophysiology, waveforms are often well isolated by hand-positioning electrodes. In

[4]The family of models defined by equation 3.3 includes cosine tuning as a special case, when the bandwidths correspond to one nonparametric degree of freedom (DF) for each of $\cos(d)$ and $\sin(d)$. The DFs of a nonparametric model are defined as the trace of the projection matrix for the regression, to mirror the parametric case, for which that trace equals the number of covariates included in the model. Hence, a nonparametric model that uses p DFs can be thought of as a polynomial of degree p in a covariate or as a model with p different covariates.

that case, classic and proposed spike sorting are equivalent. However, the traditional practice of optimizing the signal-to-noise ratio by manipulating the electrode placement is no longer always possible or practical, for example, when electrodes or arrays are chronically implanted. In that case, many voltage measurements exceeding the threshold will be noise or artifacts, and a significant proportion of spikes will be corrupted by noise. Our method can handle such situations. All we have to do is assume that one of the I neurons recorded by the electrode is a "noise neuron" whose firing rate is constant, and to which we assign the noise spikes. Additionally, as argued in section 2.5, equation 3.2 explicitly models through $\pi_x^*(c)$ the probabilities of several neurons' spiking together as functions of the neurons' firing rates, which will facilitate the identification of noise-corrupted spikes.

Consider the simulated example inspired by Olson, Gettner, Ventura, Carta, and Kass (2000). The experiment that produced the data analyzed there examined the temporal evolution of neuron firing in a part of the brain affected by attention, the supplementary eye field (SEF). A simplified version of the experiment is as follows:

1. A central target appears on a screen at time $t = 0$. A monkey must maintain fixation on the target.
2. A cue appears at a peripheral target and gets turned off.
3. At $t = 200$, the central target is turned off, which is the signal to move the eyes to the peripheral target.

One objective of the study was to draw inferences about the temporal evolution of neurons' firing rates. In that case, the covariate that modulates the firing rates is $c_t = t$, the experimental time. Figure 7B shows the firing rate of a neuron typical of those we estimated for these data. We use this rate as the true firing rate $\lambda_1^*(t)$ of a neuron in the simulation below. Suppose that the noise on that electrode is normally distributed with mean 0 and variance 2. We set the threshold at 1, so that a spike is recorded each time the electrode voltage exceeds 1. A noise signal with the stated characteristics, sampled every millisecond and thresholded at 1, corresponds to a constant noise spiking rate of $\lambda_2^*(t) = 308$ Hertz.[5] Once again, we reduce recorded waveforms to their first PCs and take their true distributions f_{01}^* to be normal with mean 8 and standard deviation 1. We also assume that the first PCs of noise measurements are normally distributed, with mean 0 and standard deviation 5. Given the firing rate of the neuron in Figure 7B and the noise specification, the probability that the voltage-crossing threshold corresponds to noise is $\pi_{10}^* = 87.1\%$. The remaining crossings correspond to actual spikes in proportion $\pi_{01}^* = 8.9\%$, or to spikes corrupted by noise in proportion $\pi_{11}^* = 4\%$. That is, about 30% of the neuron's spikes are corrupted by noise. The PCs of the noise-corrupted spikes are unlikely to have

[5]The probability of a spike in a 1 msec bin is $\Pr(Z > 1) = 0.308$ where Z is Normal(0,1).

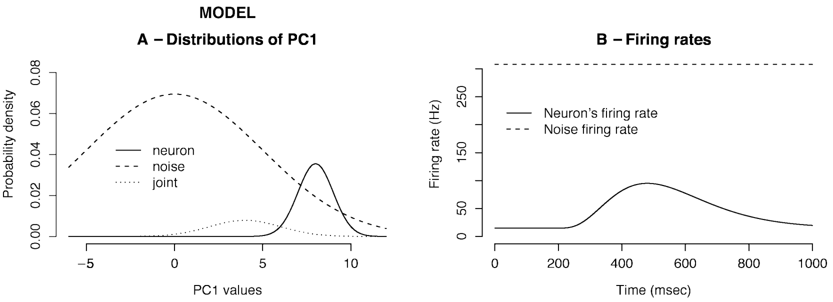


Figure 7: Noisy channel example. (B) An electrode records one neuron, whose true firing rate $\lambda_1^*(c_t)$ varies with $c_t = t$, the experimental time. The electrode's voltage is very noisy and crosses threshold at the rate of 308 Hertz. Out of all "spikes" recorded at the electrode, $\pi_{10}^* = 87.1\%$ of them are noise, $\pi_{01}^* = 8.9\%$ are real spikes, and $\pi_{11}^* = 4\%$ are noise-corrupted spikes. For spike sorting, we reduce waveform and noise measurements to their first PCs. Their true distributions $f_x^*(a)$, $x \in \mathcal{X}$ are shown in *A*, weighted by the probabilities of their occurrences, π_x^*. The noise cluster overlaps the spike clusters almost entirely, so misclassification errors will be numerous.

distribution f_{01}^*, so we assumed that f_{11}^* was normal with mean 4 and standard deviation 2. Figure 7A displays $\pi_x^* f_x^*$ for $x \in \mathcal{X}$.

Using the model in Figure 1, we simulated the noisy electrode spike train for 20 repeats of the experiment, letting the neuron and the noise spike according to Poisson processes with rates $\lambda_i^*(d)$, $i = 1, 2$. For spike sorting, we assumed normal distributions for the PCs, we assumed that the noise was produced by a "noise neuron" with unknown constant spiking rate, and, with no obvious choice of firing rate model, we used regression splines in t with six knots placed at equally spaced quantiles of the data, as in Ventura, Carta, Kass, Gettner, and Olson (2002). Another reasonable nonparametric model that could be fitted to these data is a PSTH, as is done in Ventura (2009).

Figure 8 shows step-by-step runs of the traditional and linked EM algorithms to estimate equation 2.2 and equation 2.7, respectively. The same initial values were used in Figures 8A and 8B. Once again, Figure 8B shows clearly that the linked EM algorithm can estimate firing rates before the data are spike-sorted. Figure 8 also illustrates the benefit of modeling the probability of joint spikes $\pi_{11}(c; \Theta)$ as a function of the firing rates, as per equation 2.4. Indeed, given the same initial values, the traditional EM algorithm fails to detect the cluster of noise-corrupted spikes, while the linked EM algorithm identifies the three clusters well. For traditional spike sorting, the estimate of the proportion π_{11}^* of noise-corrupted spikes is less than 10% of its true value. In contrast, the proposed procedure has $\pi_{11}(c; \widehat{\Theta}) \doteq \pi_{11}^*(c)$ for all c, and $\int \pi_{11}(c; \widehat{\Theta})\, dc \doteq \int \pi_{11}^*(c)\, dc$, so it accounted for noise-corrupted

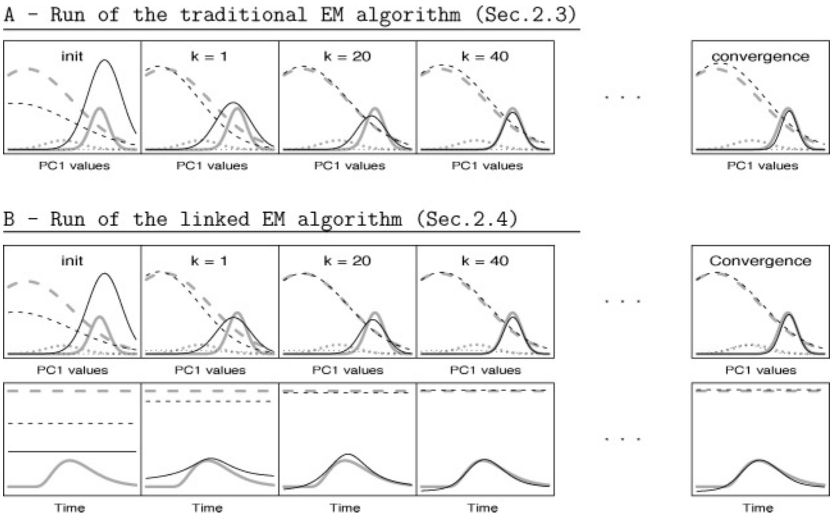


Figure 8: (A) Estimation of $\pi_x f_x(a; \psi_x)$, $x \in \mathcal{X}$ in equation 3.1 by the traditional EM algorithm of section 2.3. The plots from left to right show the initial values $\pi_x^{(0)} f_x(a; \psi_x^{(0)})$; the first, 20th, and 40th iterations $\pi_x^{(k)} f_x(a; \psi_x^{(k)})$; and the solution $\hat{\pi}_x f_x(a; \hat{\psi}_x)$ after convergence. The thick gray curves are the true $\pi_x^* f_x^*(a)$. (B) Joint estimation of $f_x(a; \psi_x)$, $x \in \mathcal{X}$ and $\lambda_i(c; \theta_i)$, $i = 1, \ldots, I$ in equation 3.2 by the linked EM algorithms in section 2.4 The same initial values $f_x(a; \psi_x^{(0)})$ were used in A and B. The initial values $\lambda_i(c; \theta_i^{(0)})$ in B are constant rates equal to the starting values $\pi_x^{(0)}$ used in A. The algorithm in B provides estimates of tuning functions before the data are spike-sorted. In addition, it yields estimates $f_x(a; \hat{\psi}_x)$ that are closer to the true distributions $f_x^*(a)$ than does the algorithm in A.

spikes in the correct proportion. This facilitated the estimation of $f_{11}(t; \psi_{11})$ and, in turn, the estimation of the other WF distributions. We discuss convergence properties of the EM algorithms further in section 3.6.

With Π, Ψ, and Θ estimated, we apply traditional and proposed spike sorters to the electrode spike train. The noise cluster overlaps almost completely the spike WF clusters, which results in a very high spike misclassification rate of 51%. The proposed approach reduces that rate by only 1.5%. Note, however, that a modest reduction in misclassification rates can have substantial effects on tuning curve estimates, which in turn might have an impact on scientific conclusions, as we illustrate in the next section and return to in section 4.

3.4 Designed Experiment Example. Consider a common type of experiment, whose aim is to compare the firing rates of neurons under two or more experimental conditions. For example, Olson et al. (2000) compared

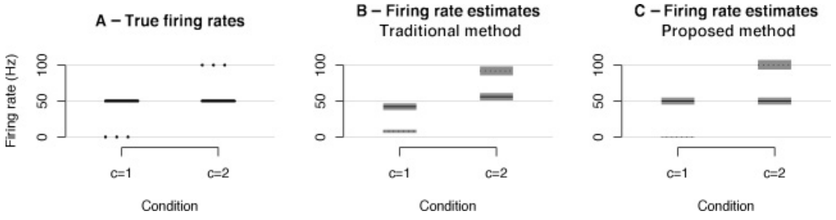


Figure 9: Designed experiment. (A) True firing rates $\lambda_i^*(c)$ of $I = 2$ neurons recorded by the same electrode, under two different experimental conditions $c = 1$ and $c = 2$. (C) Estimated firing rates $\lambda_i(c; \hat{\theta}_i)$ obtained by the linked EM algorithm, with 95% confidence intervals. True and estimated rates are equal within error. (B) Estimated rates calculated from data spike-sorted with the traditional waveform-based approach, with 95% confidence intervals. True and estimated rates do not match within error.

the responses of supplementary eye field (SEF) neurons to eye saccades made in response to endogenous and exogenous cues. Once again we consider an electrode that records $I = 2$ simulated neurons, whose waveform first PCs are as in Figure 1A. We consider two experimental conditions $c = 1$ and $c = 2$ and assume that the neurons' true firing rates are constant within each experimental condition; neuron 1 spikes at $\lambda_1^*(c) = 50$ Hz regardless of the condition, while neuron 2 does not spike when $c = 1$ and spikes at 100 Hz when $c = 2$. Neuron 1 is not tuned to c, which we should be able to verify from data.

We simulated the electrode spike train for 10 seconds in each condition and ran traditional and proposed spike sorters, assuming normal models for the spikes' first PC distributions, and a within-condition constant rate model for $\lambda_i(c; \theta_i)$. Although firing rates are constant within conditions, tuning information is still available for spike sorting because the rates are not all constant between conditions. Indeed, using tuning information reduced the spike misclassification rate from 18% to 11%.

Figure 9 shows true and estimated firing rates, obtained by regression from spike-sorted data in the traditional approach, and as the output of the linked EM algorithm in the proposed approach. The latter estimates are equal to the true rates within error. We also performed LR tests[6] to compare firing rates across conditions and found that neuron 2 had significantly different rates (p-value $\ll 0.0001$), whereas neuron 1 did not (p-value $=$ 0.84%). This is consistent with the truth. This example illustrates once more that the new spike sorter not only can estimate the neurons tuning functions but also can identify the covariates that modulate these functions.

[6]In this particular application, the LR tests are equivalent to the well-known two sample t-tests.

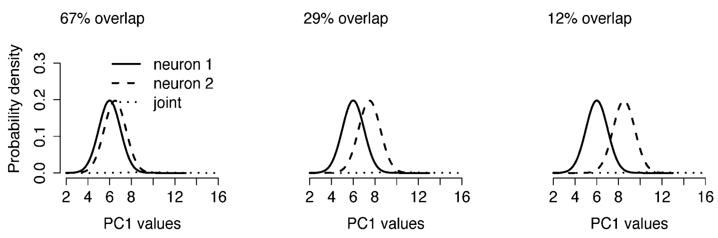
What is troubling is that the traditional procedure yields firing rate estimates that are not equal to the true rates within error (see Figure 9B). Additionally, t-tests determined that both neurons had significantly different rates in the two conditions (p-values $P \ll 0.0001$). This example is discussed further in section 4.

3.5 Spike Misclassification Rates Depend on the Amount of Waveform and Tuning Information. The proposed spike sorter yields a lower spike misclassification rate because it uses more information. In practice, the reduction in rate will be more or less substantial depending on how much information tuning carries about spikes' identities. The only situation the proposed method cannot improve in theory is when neurons are not tuned. Then $\pi_x^*(c) = \pi_x^*$, and new and traditional spike sorters are equivalent. If all neurons have the same tuning curves, tuning does not provide information to identify single-neuron spikes, but joint spikes should still be more easily identified.

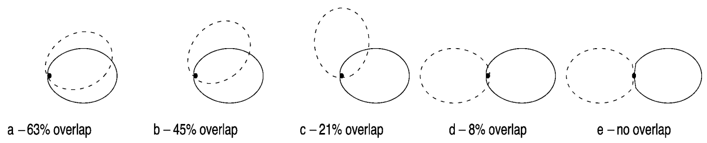
We revisit the motor cortex example of section 3.1. The WF distributions of the two neurons overlapped by 19%, so spikes' classification errors were unavoidable. Their tuning curves overlapped by 21% and thus carried substantial information about spike identities, which reduced the spike misclassification rate from 18% to 9%. We now repeat the same spike-sorting exercise, letting the overlaps of the tuning curves and the WF clusters vary. The precise definition of overlap is given in the appendix. Although overlap percentage is only a crude measure of curve similarity, the idea is that the more two curves overlap, the less information they provide to identify neurons. An overlap of 0% means that two curves (i.e., two WF distributions or two tuning curves) share no overlap. An overlap of 100% means that two curves are identical so they provide no information to separate neurons.

Figure 10C plots the spike misclassification rates for traditional and proposed spike sorters as functions of the overlap percentage between the WF distributions f_{01}^* and f_{10}^*. The solid bold curve corresponds to traditional spike sorting and the lettered curves to the proposed approach for the five tuning curve overlaps shown in Figure 10B. (Note that circular coordinates do not visually represent overlap percentages well.) The misclassification rates reported for Figure 4 can be read from the bold and c-curves in Figure 10C, at value 19% on the x-axis. The values of the lettered curves when the WF clusters overlap is 100% give the spike misclassification rate when only tuning information is used for spike sorting (see equation 2.8). If the tuning curves share no overlap (curve e), spikes can be classified perfectly whatever the information in waveforms. It is clear from Figure 10C that incorporating tuning information always helps classify spikes—if, that is, the tuning curve model is not too badly misspecified (see section 3.2). Given that traditional and proposed approaches share the same set of assumptions, this help comes at the very modest computational cost of running the linked EM algorithm in place of the traditional EM algorithm.

A - Overlap of waveform clusters



B - Overlap of tuning curves



C - Spike misclassification rate

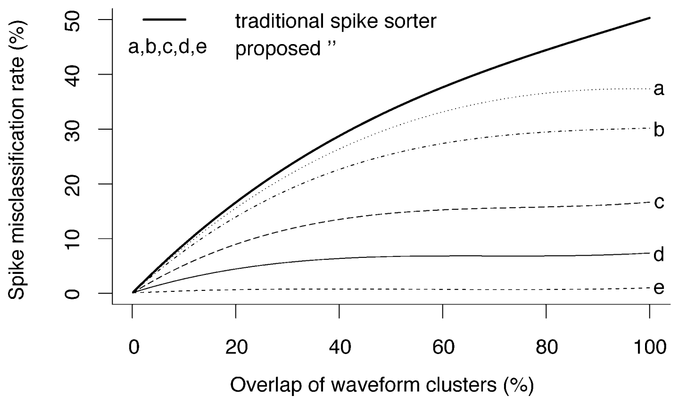


Figure 10: (C) Spike misclassification rates of traditional and proposed spike sorters as functions of waveform clusters overlap for the different tuning functions overlaps shown in B. (A) Sample of WF clusters overlap. The spike misclassification rate increases the more WF clusters and tuning functions overlap. However, as expected from theory, including tuning information reduces the spike misclassification rate. The right end value on each curve in C is the misclassification rate obtained by spike sorting based on tuning information only (see equation 2.8).

Note that the misclassification rates in Figure 10C did not account for the classification of the joint spikes, because it was not easy to find a meaningful definition of the overlap between the three distributions f_x^* we could use for the x-axis. However the proposed spike sorter helped retrieve about 20% more joint spikes than the traditional sorter for all configurations of tuning curves shown in Figure 10B. This remains true when the two neurons have equal tuning curves (not shown).

We also assumed that the electrode recorded $I = 2$ neurons rather than estimate this number by penalized likelihood. In practice, I needs to be determined from the data, with outcome matching the truth or not depending on the amount of waveform and tuning information. This is discussed in the next section.

3.6 Combining Waveform and Tuning Information Helps the EM Algorithm to Converge. With all EM algorithms, the issues of choosing initial values, determining the number of neurons, and avoiding convergence to local likelihood maxima must be addressed. Although practically important, these issues are not specific to the EM algorithms used here, so we kept their development to a minimum to avoid cluttering the main ideas. However, we next provide evidence that including tuning information for spike sorting helps determine the number of neurons and facilitates the convergence of the algorithm to the global maximum likelihood.

Consider again the simulation in Figure 10, for which we had assumed $I = 2$. We now determine I by penalized likelihood. Consider first the traditional approach, which uses WF information only (solid bold curve in Figure 10). When the WF distributions f_{01}^* and f_{10}^* do not overlap, penalized likelihood consistently finds $I = 2$ neurons in every data set simulated from the model. When f_{01}^* and f_{10}^* overlap completely, that number drops to $I = 1$ in every simulated data set. And as expected, the chance of finding two neurons decreases as the overlap increases. Adding tuning information makes it more likely to determine the correct number of neurons. For example, if neurons have tuning curves as in Figure 10Bcde, penalized likelihood correctly finds $I = 2$ neurons in every simulated data set, regardless of the amount of WF information. This includes the case where WF distributions overlap completely. Tuning curves such as those in Figure 10Bab also increase the chance of finding the correct number of neurons, although the chance of finding only one neuron is still substantial when WF distributions overlap substantially.

Next, Figure 11 illustrates that combining waveform and tuning information helps the EM algorithm converge to the global maximum likelihood. Figure 11A shows $\pi_x^* f_x^*(a)$ and their estimates obtained by the traditional EM algorithm applied to a data set simulated from the model in Figure 1. True and estimated curves do not match perfectly, which is due partly to data variability (estimates match the truth only up to random error), but could also be due to the EM algorithm converging to a local maximum.

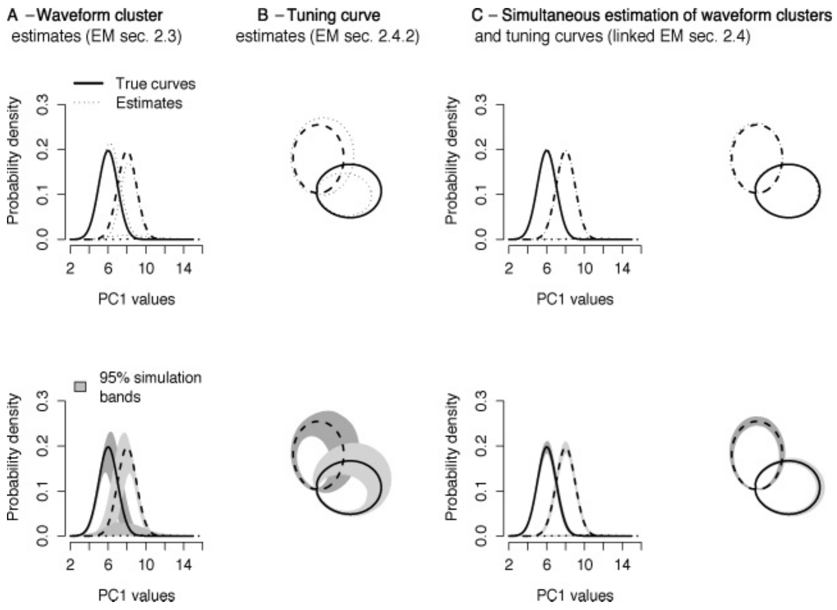


Figure 11: Convergence of EM algorithms. Motor cortex example. (A) Traditional EM algorithm to estimate waveform clusters (see section 2.3). The top plot shows true $\pi_x^* f_x^*(a)$, $x \in \mathcal{X}$ in equation 2.1 and their estimates for one simulated data set. The bottom plot shows the bands in which 95% of estimates from 100 simulated data sets fall. (C) Proposed linked EM algorithm to estimate jointly waveform clusters and tuning curves (see section 2.4). The top plot shows, side by side, true $\pi_x^* f_x^*(a)$, $x \in \mathcal{X}$ and $\lambda_i^*(d)$, $i = 1, \ldots, I$ in equation 2.6, and their estimates for one simulated data set. The bottom plot shows 95% simulation bands. (B) Linked EM algorithm used to estimate tuning curves, ignoring waveform information (see section 2.4.2). The top plot shows true $\lambda_i^*(d)$, $i = 1, \ldots, I$, and their estimates, for one simulated data set. The bottom plot shows 95% simulation bands. This plot illustrates that combining waveform and tuning informations not only helps spike-sort the data, but also helps the EM algorithms converge to the likelihood global maximum.

Figure 11C shows the outcome of the linked EM algorithm applied to the same data set and initialized with the same values. True and estimated WF distributions now match more closely, which confirms that the algorithm in Figure 11A did converge to a local maximum. For completeness, we also applied the second part of the linked EM algorithm to the same data set, with the aim of spike-sorting the data based on tuning information only. The resulting tuning curve estimates are in Figure 11B. The large discrepancy between true and estimated curves in comparison to Figure 11C suggests that the algorithm converged to a local rather to the global maximum.

The lower panels of Figure 11 show the true curves and simulation bands in which 95% of estimates from 100 simulated data sets fall. The bands in Figure 11C remain unchanged if we let the WF distributions f_{01}^* and f_{10}^* spread apart so they no longer overlap (not shown), which suggests that the variability we see in Figure 11C is due to the randomness of the data rather than to convergence difficulties of the linked EM algorithm. By comparison, the fatter bands in Figures 11A and 11B must therefore be partly due to convergence problems.

Figure 11 suggests that the effect of adding tuning information to the waveform information, or vice versa, facilitates the joint estimation of WF distributions and tuning functions, by which we mean that Ψ and Θ are more likely to be estimated by their maximum likelihood estimates rather than by some values at a local maximum of the likelihood. Note that improved estimation is a benefit that is separate from the spike-sorting problem. To clarify, combining both sources of information yields a spike sorter that is superior to the traditional sorter. Because spike-sorting rules depend on unknown parameters, they must be estimated from data. The estimates of WF distributions and tuning curves happen to be superior when they are obtained jointly by the linked EM algorithm rather than separately.

4 Discussion

Current spike-sorting methods focus on clustering neurons' characteristic spike waveforms. The resulting spike-sorted data are then typically used to estimate how covariates modulate the firing rates of neurons. However, when covariates do modulate the firing rates, they too provide information about spikes' identities, which thus far has been ignored for the purpose of spike sorting. We proposed a new automatic spike sorter that combines spike waveform and tuning information. In theory, it yields the lowest spike misclassification rates since it efficiently uses more information than current spike sorters do. It also facilitates the identification of joint spikes and noise-corrupted spikes, because it models explicitly the probabilities of several neurons spiking together as functions of their firing rates. But as with all other statistical methods, its performance in practice will depend on the quality of the assumed models for waveform features and tuning functions.

The proposed spike sorter essentially consists of performing spike-sorting and tuning function estimation simultaneously rather than sequentially, as is currently done. No more assumptions are needed, although the tuning model must now be specified before the data are spike-sorted. While parametric models such as normal or t-distributions are often assumed for waveform features, tuning functions should be modeled nonparametrically unless specific knowledge suggests otherwise. Indeed, waveform features are qualitatively invariant to the experiments that produce them, and models for their distributions have long been evaluated. In contrast, neurons' tuning properties are not only experiment specific, they are typically of primary interest. We showed that nonparametric firing rate models such as gaussian filters and splines could be used within our framework and that likelihood-based model selection techniques could be applied. For example, we tested that gaussian filters provided significantly better fits than cosine functions to the tuning curves of simulated motor cortex neurons and determined the smoothness of these filters by cross-validation. We also showed that standard statistical tests of hypotheses could determine which covariates modulate neurons spiking.

So what could go wrong? We illustrated that assuming an inadequate tuning model could severely bias tuning curve estimates, and hence bias spike classifications. But to be fair to the proposed approach, tuning curve estimates were just as biased when the same model was fitted to the true neurons′ spike trains. This shows that model selection for tuning functions is equally important whatever the estimation procedure may be. An inadequate model for waveform features could also bias tuning curve estimates, since both waveform and tuning models are estimated simultaneously. But tuning curves fitted to data first spike-sorted based on waveforms would also be biased, since a bad waveform model would bias spike classifications.

Unexpectedly, it is the traditional procedure that produced a puzzling result. In the example in section 3.4, the neurons′ firing rates estimated by the proposed spike sorter matched the true rates within error, whereas the rates estimated from data first spike-sorted based on waveforms did not. Note that this effect was not seen in Figures 5 and 6, because the rates estimated by the proposed spike sorter were compared to estimates obtained from the true neurons′ spike trains. The result of section 3.4 is not a fluke but a general flaw of the traditional sequence of first spike sorting based on waveforms, then estimating tuning functions. Ventura (2009) shows that tuning functions estimated by the traditional approach are systematically biased and inconsistent, whereas the proposed spike sorter yields consistent estimates.

Our last comments concern the assumptions we made and possible algorithmic improvements. We estimated the proposed spike sorter via an EM maximum likelihood algorithm. We presented the general form of this algorithm and provided a detailed implementable version under the assumption that neurons are independent and spike according to Poisson processes. We assumed that spike waveform features were normally distributed, but distributions deemed more appropriate could readily be used instead, for example, t-distributions, as suggested by Harris et al. (2000) and Shoham et al. (2003), provided an EM algorithm existed or could be developed to estimate their parameters. Some spike-sorting methods also include, with varying degrees of formality, additional information such as refractory periods and nonstationarity of waveforms, for example spike amplitude decays after short interspike intervals (Fee et al., 1996; Pouzat, Delescluse, Voit, & Diebolt, 2004). We plan to incorporate these effects in our framework by letting the neurons' firing rates depend on the past. We kept to a minimum the issues of determining the number of neurons, choosing initial values, and avoiding convergence to local likelihood maxima. However, we provided evidence that including tuning information alleviated these issues. More sophisticated parameter space searches could also be implemented, for example, reversible Markov chain Monte Carlo methods (Richardson & Green, 1997), or the method of Figueiredo and Jain (2002).

Appendix

A.1 Step-by-Step Run of the Linked Algorithm for Figure 5. See Figure 12.

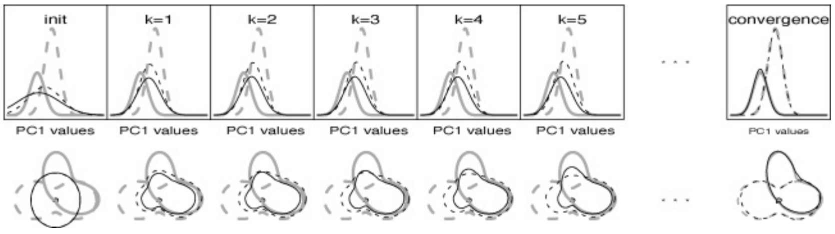


Figure 12: Joint estimation of $\pi_x f_x(a; \psi_x)$, $x \in \mathcal{X}$, and $\lambda_i(c; \theta_i)$, $i = 1, \ldots, I$, by the linked EM algorithm in section 2.4. The plots from from left to right show the initial values, the first five iterations, and the solutions after convergence. The thick gray curves are the true $\pi_x^* f_x^*(a)$ and $\lambda_i^*(c)$.

A.2 Percentage Overlap Between Two Curves. The percentage overlap between two curves, for example, two tuning curves or two WF distributions (see Figure 13) is the intersection area under the curves (dark gray) divided by the total area under the two curves (light gray plus dark gray). A value of 1 indicates complete overlap and a value 0 perfect separation.

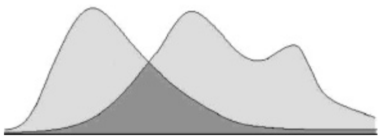


Figure 13.

Acknowledgments

I was supported by NIH grants 2RO1MH064537 and 1R01EB005847 and thank two reviewers for their thoughtful comments.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19* (6), 716–723.

Brown, B. N., Barbieri, R., Ventura, V., Kass, R. E., & Frank, L. M. (2002). The time-rescaling theorem and its applications to neural spike train data analysis. *Neural Computation*, *14*, 325–346.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *J. Royal Statistical Society B*, *39*(1), 138.

Fee, M. S., Mitra, P. P., & Kleinfeld, D. (1996). Variability of extracellular spike waveforms of cortical neurons. *J. Neurophysiol*, *76*, 3823–3833.

Figueiredo, M. A. F., & Jain, A. K. (2002). Unsupervised learning of finite mixture models: Pattern Analysis and Machine Intelligence. *IEEE Transactions*, *24*, 381–396.

Harris, K. D., Henze, D. A., Csicsvari, J., Hirase, H., & Buzsaki, G. (2000). Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *J. Neurophysiol.*, *84*, 401–414.

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

Hathaway, R. J. (1986). Another interpretation of the EM algorithm for mixture distributions. *Stat. Prob. Lett.*, *4*, 53–56.

Kass, R. E., Ventura, V., & Brown, E. N. (2005). Statistical issues in the analysis of neuronal data. *J. Neurophysiology*, *94*, 8–25.

Lewicki, M. S. (1994). Bayesian modeling and classification of neural signals. *Neural Computation*, *6*(5), 1005–1030.

Lewicki, M. S. (1998). A review of methods for spike sorting: The detection and classification of neural action potential. *Network: Comput. Neural Syst.*, *9*, R53–R78.

McLachlan, G. J., & Peel, D. (2000). *Finite mixture models*. New York: Wiley.

Neal, R., & Hinton, G. E. (1998). *A view of the EM algorithm that justifies incremental, sparse, and other variants: Learning in graphical models*. Norwell, MA: Kluwer.

Ohberg, F., Johansson, H., Bergenheim, M., Pedersen, J., & Djupsjobacka, M. (1996). A neural network approach to real-time spike discrimination during simultaneous recording from several multi-unit nerve filaments. *J. Neurosci. Methods*, *64*(2), 181–187.

Olson, C. R., Gettner, S. N., Ventura, V., Carta, R., & Kass, R. E. (2000). Neuronal activity in macaque supplementary eye field during planning of saccades in response to pattern and spatial cues. *J. Neurophysiol.*, *84*, 1369–1384.

Pouzat, C., Delescluse, M., Voit, P., & Diebolt, J. (2004). Improved spike-sorting by modeling firing statistics and burst-dependent spike amplitude attenuation: A Markov chain Monte Carlo approach. *J. Neurophysiol.*, *91*(6), 2910–2928.

Richardson, S., & Green, P. (1997). On Bayesian analysis of mixtures with unknown number of components. *J. Royal Statistical Society B 59*, 731–792.

Sahani, M. (1999). *Latent variable models for neural data analysis*. Unpublished doctoral dissertation, California Institute of Technology.

Sahani, M., Pezaris, J. S., & Andersen, R. A. (1997). On the separation of signals from neighboring cells in tetrode recordings. In M. S. Kearns, S. Solla, & D. Cohn (Eds.), *Advances in neural information processing systems, 11*. Cambridge, MA: MIT Press.

Salganicoff, M., Sarna, M., Sax, L., & Gerstein, G. L. (1988). Unsupervised waveform classification for multi-neuron recordings: A real-time, software-based system I: Algorithms and implementation. *J. Neurosci. Methods*, *25*(3), 181–187.

Schwartz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, *6*(2), 461–464.

Shoham, S., Fellows, M. R., & Normann, R. A. (2003). Robust, automatic spike sorting using mixtures of multivariate t-distributions. *J. Neurosci. Methods*, *127*, 111–122.

Ventura, V. (2008). Spike train decoding without spike sorting. *Neural Computation*, *20*, 923–963.

Ventura, V. (2009). Traditional waveform-based spike sorting yields biased rate code estimates. *PNAS*, *106*, 6921–6926.

Ventura, V., Carta, R., Kass, R. E., Gettner, S. N., & Olson, C. R. (2002). Statistical analysis of temporal evolution in single-neuron firing rates. *Biostatistics*, *1*, 1–20.

Wasserman, L. (2004). *All of statistics*. New York: Springer.

Received December 14, 2007; accepted January 5, 2009.