Spike Train Decoding without Spike Sorting

Valérie Ventura

Department of Statistics
and
Center for the Neural Basis of Cognition
Carnegie Mellon University

Running Head: **Spike Train Decoding without Spike Sorting**

ABSTRACT

We propose a novel paradigm for spike train decoding, which avoids entirely spike sorting based on waveform measurements. This paradigm directly uses the spike train collected at recording electrodes from thresholding the bandpassed voltage signal. Our approach is a paradigm, not an algorithm, since it can be used with any of the current decoding algorithms, such as population vector or likelihood based algorithms. Based on analytical results and an extensive simulation study, we show that our paradigm is comparable to, and sometimes more efficient than, the traditional approach based on well isolated neurons, and that it remains efficient even when all electrodes are severely corrupted by noise, a situation that would render spike sorting particularly difficult. Our paradigm will also save time and computational effort, both of which are crucially important for successful operation of real-time brain-machine interfaces. Indeed, in place of the lengthy spike sorting task of the traditional approach, it involves an exact expectation EM algorithm that is fast enough that it could also be left to run during decoding to capture potential slow changes in the states of the neurons.

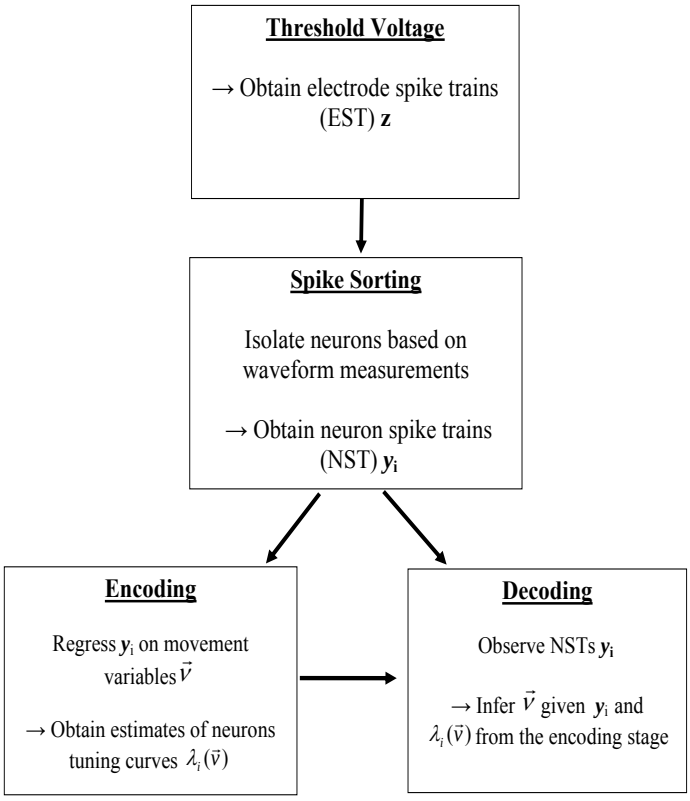


Figure 1: *Traditional NST encoding/decoding paradigm. Both encoding and decoding use as inputs the neurons' spike trains (NSTs)* $\mathbf{y_i}$*, which are extracted from the electrodes' spike trains (ESTs)* $\mathbf{z}$ *via spike sorting.*

1 Introduction

Assume we have the spike trains of motor cortical neurons, each tuned to hand velocity, and that our goal is to predict movement (Georgeopoulos *et al.*, 1988). This "population coding" is of interest partly for its role in the neural basis of action and also for its use in brain-machine interfaces, which would allow direct mental control of external devices (Wessberg *et al.* 2000, Carmena *et al.* 2003, Musallam *et al.* 2004, Schwartz, 2004, Santhanam *et al.* 2006, Hochberg *et al.* 2006, Brockwell *et al.* 2007). Decoding of this population signal has been accomplished successfully with the population vector (PV) algorithm (Georgopoulos *et al.*, 1986, 1988, 1989; Taylor *et al.*, 2002), and linear methods (Salinas and Abbott, 1994, Moran and Schwartz, 1999), which characterize each neuron's activity by preferred direction and firing rate. Maximum likelihood (Brown *et al.*, 1998) and Bayesian (Sanger, 1996) methods make use of the full probabilistic descriptions of each neuron's activity, and are efficient when the model is correct (Kass *et al.*, 2005). More recently, filtering/dynamic Bayesian methods combine a maximum likelihood approach with smoothness constraints on the decoded trajectories (Zhang *et al.*, 1998; Brown *et al.*, 1998; Brockwell *et al.*, 2004; Barbieri *et al.* 2004; Wu *et al.* 2004; Shoham *et al.* 2005; Truccolo *et al.* 2005; see Brockwell *et al.* 2007 for a review and references therein).

These increasingly efficient decoding methods all use as inputs the spike trains of well isolated cortical neurons obtained from spike sorting the electrical signal at electrodes chronically implanted in the cortex.

Fig.1 summarizes the current encoding/decoding paradigm. To keep the development of ideas simple, we assume that we record the voltage of single electrodes, as opposed to tetrodes or arrays. We focus on a representative electrode that records I neurons. First the bandpassed signal at the electrode is thresholded to give the times at which spikes occur. We discretize time in bins small enough so that at most one spike can occur in a bin. Without loss of generality we use one millisecond bins. The discretized electrode spike train (EST) is denoted by $\mathbf{z} = (z_t; t = 1, \ldots, T)$, where $z_t = 1$ means that a spike occurred at t, and otherwise $z_t = 0$. It is the aggregate of I spike trains, $\mathbf{y_i} = (y_{it}, t = 1, \ldots, T)$, each produced by a neuron whose activity is at least partly determined by movement variables $\vec{v}$. To facilitate pictorial representations in this paper, we take $\vec{v} = (v_x, v_y)$ to be the velocity of a hand in a 2D plane, though in real application we would consider 3D intended or actual velocity, position, acceleration, etc. The neurons' spike trains (NSTs) $\mathbf{y_i}$ are not observed but are inferred to some accuracy via spike sorting, the process of assigning spikes to neurons based on discriminating measurements of their characteristic waveforms. Next is encoding, the process of estimating how neurons encode information about $\vec{v}$. The standard approach is to estimate their firing rates $\lambda_i(\vec{v}; \theta_i)$, with unknown parameters θ_i usually estimated by regressing the NST $\mathbf{y_i} = (y_{it}, t = 1, \ldots, T)$ on velocity $\vec{v}_t$ according to, for example,

$$y_{it} = \lambda_i(\vec{v}_t; \theta_i) + \epsilon_t, \quad t = 1, \ldots, T, \tag{1}$$

where ϵ_t are random errors [1]. The relationship between $\lambda_i(\vec{v}; \theta_i)$ and $\vec{v}$ can be visualized by plotting spike counts against $\vec{v}$ (Georgopoulos *et al.*, 1982). This plot, or prior knowledge, helps build a model for $\lambda_i(\vec{v}; \theta_i)$. For example, cosine tuning specifies that the tuning function varies linearly with $\vec{v}$ according to

$$\lambda_i(\vec{v}; \theta_i) = \theta_{i0} + \theta_{i1} v_x + \theta_{i2} v_y, \tag{2}$$

where θ_{i0} measures the baseline firing rate and the vector $(\theta_{i1}, \theta_{i2})$ points in the preferred direction of neuron i, while its magnitude measures its directional sensitivity. Finally, decoding consists of predicting velocity given the NSTs $\mathbf{y_i}$ and the estimated tuning curves $\hat{\lambda}_i$. For example, the population vector (Georgopoulos *et al.* 1986, 1988) predicts velocity at time t by a normalized version of

$$\vec{P}_t = \sum_{i=1}^{N} y_{it} \vec{D}_i, \tag{3}$$

where $\vec{D}_i$ is the unit-magnitude vector whose direction maximizes $\hat{\lambda}_i$. Note that the particular forms of Eqs.1, 2, and 3 were chosen for their simplicity in this introduction. Alternatives which may be more appropriate are discussed later.

Spike sorting is a difficult task, as evidenced by the large body of literature (For reviews see Lewicki, 1998; Brown *et al.* 2004). 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 contribute to the recorded data and determining which neurons produced which spikes. By and large, spike sorting papers focus on two broad problems, feature selection and clustering techniques. Features can be raw waveform measurements or projections on lower dimensional spaces, e.g. projections in PCA subspaces. Clustering techniques are many and range from simple nonparametric nearest-neighbors methods to sophisticated mixture model based clustering (Shoham *et al.* 2003). The former usually yield hard clusters, while the latter yield soft clusters. Some methods (Fee *et al.* 1996, Pouzat *et al.* 2004) include, more or less formally, additional information such as refractory periods and non-stationarity of waveforms.

[1] Spike counts in larger time bins would normally be used in Eq.1. We omitted this step because it is not crucial to the development of ideas in the introduction and to avoid excess notation.

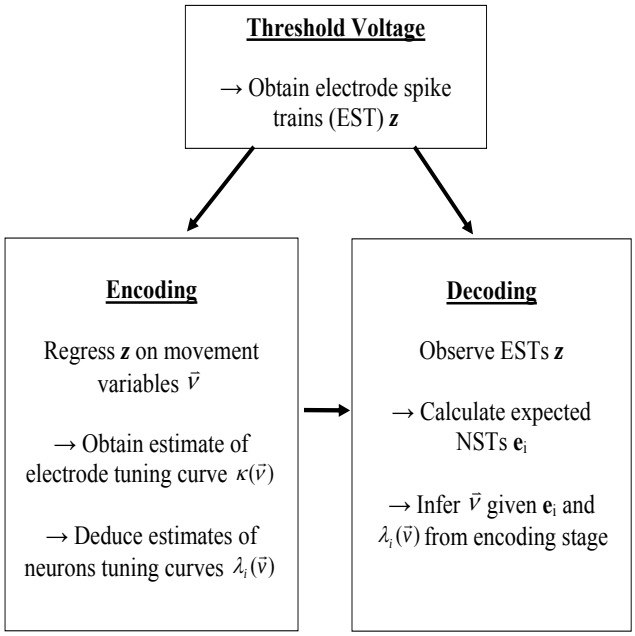


Figure 2: *Proposed EST encoding/decoding paradigm. Both encoding and decoding use directly as inputs the electrodes' spike trains (ESTs)* $\mathbf{z}$*. Spike sorting is avoided completely.*

Despite significant improvements, spike sorting remains a lengthy and imperfect process (Harris *et al.* 2000). For example, it is difficult to classify spikes when waveform measurements clusters overlap and to detect when several neurons spiked together. These and other problems are exacerbated in the low signal to noise ratio (SNR) case, when waveforms and noise have similar amplitudes and noise can deform the recorded waveforms. These problems are severe enough that noisy electrodes are often abandoned even though they might be recording tuned neurons. The computational effort required for good spike sorting is of particular concern in the context of neural prostheses, which we had in mind in developing this work. Indeed for a prosthetic device to be operated by a human in real time, a prohibitively high data rate is required to transmit raw neuronal signals from a chronically implanted device in the brain for the purpose of spike sorting, which is beyond the capability of miniature battery-powered wireless links. Therefore spike sorting may have to be done directly in the brain by a small chip, whose computing power will likely be too limited to allow use of the most accurate spike sorting algorithms. To complicate matters, chronically implanted recording devices cannot be placed strategically to minimize noise, so that low SNR can be expected. Electrodes might also shift over time and thus record spike trains from different neurons, which may in turn require regular adjustment of the spike sorter parameters.

The purpose of this paper is to propose a spike sorting-free encoding/decoding paradigm, which is statistically as efficient as the traditional paradigm, even in the low SNR case. Fig.2 summarizes what we also refer to as the direct method, because it takes directly as inputs the recorded ESTs rather than the NSTs. Avoiding spike sorting begins with the observation that, with all firing rates expressed in spikes per milliseconds, the firing rate κ of an electrode is related to the firing rates λ_i of the I neurons it records:

$$\kappa(\vec{v};\Theta) = 1 - \prod_{i=1}^{I} \left(1 - \lambda_i\left(\vec{v};\theta_i\right)\right), \tag{4}$$

where $\Theta = (\theta_i, i = 1, \ldots, I)$ is the combined vector of tuning curve parameters.

Eq.4 was obtained by writing the probability of detecting one spike as one minus the probability that no neuron spiked, since a spike will be detected at the electrode at time t if and only if at least one neuron spiked at t. Eq.4 implies that the EST $\mathbf{z}$ contains information about each $\lambda_i(\vec{v};\theta_i)$, with which θ_i can be estimated. Indeed, regressing $\vec{v}$ on $\mathbf{z}$ according to

$$z_t = \kappa(\vec{v}_t;\Theta) + \epsilon_t, \quad t = 1,\ldots,T, \tag{5}$$

yields an estimate $\widehat{\Theta} = (\hat{\theta}_i, i = 1,\ldots,I)$, which in turn provides estimates $\hat{\lambda}_i = \lambda_i(\vec{v},\widehat{\theta}_i)$. The NSTs are not needed to obtain estimated tuning curves. Note that unidentifiabilities might arise from estimating $\lambda_i(\vec{v};\theta_i)$ from Eq.5. Also, the regressions in Eqs.1 and 5 usually require either spike counts in larger time bins or use of the binary models given later by Eqs.11 and 12. To keep the introduction simple and to avoid excess notation, we defer the treatment of these issues to the next section.

Eq.4 also provides the link between the observed EST $\mathbf{z}$ and unobserved NSTs $\mathbf{y_i}$ which allows us to bypass spike sorting for decoding. Given that a spike is detected at the electrode at time t ($z_t = 1$), the probability that neuron i produced that spike is $\lambda_i(\vec{v_t})/\kappa(\vec{v_t})$, so that the expectation of $\mathbf{y_i}$ at time t, given z_t, is

$$e_{it} = E(y_{it} \mid z_t) = z_t \, \frac{\lambda_i\left(\vec{v}_t;\theta_i\right)}{\kappa(\vec{v}_t;\Theta)}, \quad t = 0,\ldots,T. \tag{6}$$

Note that an electrode that records just one neuron has $e_{it} = y_{it}$, since e_{it} reduces to z_t in Eq.6, while spike sorting would yield $y_{it} = z_t$. We avoid spike sorting for decoding by using the conditional expected NSTs in Eq.6 in place of the NSTs to obtain velocity predictions. For example, we replace the population vector in Eq.3 by

$$\vec{P}_t = \sum_{i=1}^{N} e_{it}\vec{D}_i. \tag{7}$$

The principles of direct decoding in Fig.2 are conceptually straightforward. In practice however, regressions that arise from mixtures like $\kappa(\vec{v};\Theta)$ in Eq.5 are known to be difficult to fit. Below, we develop an exact expectation EM algorithm (Dempster *et al.* 1977) that is conceptually simple and leads to an easy-to-implement procedure amenable to any statistical package. It is also computationally fast compared to spike sorting. To investigate the use of Eq.6 in place of NSTs, we focus on population vector and maximum likelihood velocity predictions to demonstrate that our proposed paradigm applies across decoding methods. We do not consider dynamic Bayesian decoding algorithms because they would only add unnecessary detail and complexity. Based on analytical results and on an extensive simulation study, we demonstrate that our paradigm is comparable to, and sometimes more efficient than, the traditional approach based on well-isolated neurons, and that it remains efficient even when all electrodes are severely corrupted by noise, a situation that would render spike sorting particularly difficult.

2 Methods

We divided this section into four subsections. Sections 2.1 and 2.2 develop the algorithms for spike sorting-free encoding and decoding respectively, where encoding refers to the estimation of the neurons' tuning curves, while decoding refers to velocity prediction. One important feature of our method is that it separates noise and real spikes. Despite its importance, for clarity we delay the discussion of the low SNR case to Section 2.3. Section 2.4 describes our simulation study.

2.1 Spike sorting free encoding

We focus on one representative electrode and denote by I the number of neurons it records. I is usually obtained as a byproduct of spike sorting. We propose a spike sorting-free alternative later and assume until then that I is known.

Our task is to fit a regression like Eq.5 to obtain an estimate of Θ, which in turn provides estimates of the tuning curves, $\hat{\lambda}_i = \lambda(\vec{v}; \hat{\theta}_i)$. Maximum likelihood (ML) estimators are attractive because they make the most efficient use of the data. See for example Kass, Ventura, and Brown (2005). The ML estimator $\widehat{\Theta}$ is the value of Θ that maximizes the likelihood function

$$L(\Theta) = p(\mathbf{z} = (z_t, t = 1, \ldots, T); \Theta) = \prod_{t=1}^{T} p(z_t; \Theta), \tag{8}$$

where $L(\Theta)$ is defined as the joint distribution of the observed data, here the EST $\mathbf{z}$, and $p(z_t; \Theta)$ is the probability distribution of a spike occurring at t, which is specified below in Eq.12. The reduction of Eq.8 to the product over time bins of the marginal distributions of z_t is practically attractive because it allows us to process each time bin separately. It does not imply that $\mathbf{z}$ follows a Poisson process. Indeed, dependences of spiking probability on the past could be built into the firing rates, for example by letting λ_i depend on the time elapsed since previous spikes to account for refractory periods. See for example Kass and Ventura (2001).

Because $\mathbf{z}$ has for its firing rate the mixture $\kappa(\vec{v}; \Theta)$ in Eq.4, its distribution in Eq.8 also depends on $\kappa(\vec{v}; \Theta)$. Likelihoods that arise from such mixtures are well-known to be difficult to optimize, and a latent variable approach is often preferred. We use as latent variables the identity of every combination of neurons that could have produced a spike at the electrode and use an EM algorithm (Dempster *et al.* 1977) to optimize $L(\Theta)$. Hence, we associate with a spike at t the unobserved I-dimensional binary latent vector $x_t = (y_{1t}, \ldots y_{It})$, where $\mathbf{y_i} = (y_{it}, t = 1, \ldots, T)$ is the NST of neuron i. The NSTs are usually inferred by spike sorting. Here, they remain unknown. 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. In statistical jargon $(\mathbf{z}, \mathbf{x})$ is a latent marked point process with $\mathbf{x}$ as the unobserved marking variable.

Suppose that $\Theta^{(k)}$ is the current parameter value and that we want to update it to $\Theta^{(k+1)}$, with the eventual aim of reaching the ML estimator $\widehat{\Theta}$. The EM algorithm is based on the following inequality;

$$\begin{aligned}
\log L(\Theta) - \log L\left(\Theta^{(k)}\right) &\geq \sum_{\mathbf{x}\in\mathcal{X}} \log p(\mathbf{x}, \mathbf{z}; \Theta) p\left(\mathbf{x} \mid \mathbf{z}; \Theta^{(k)}\right) - \sum_{\mathbf{x}\in\mathcal{X}} \log p(\mathbf{x}, \mathbf{z}; \Theta^{(k)}) p\left(\mathbf{x} \mid \mathbf{z}; \Theta^{(k)}\right) \\
&= E\left(\log p(\mathbf{X}, \mathbf{z}; \Theta) \mid \mathbf{z}; \Theta^{(k)}\right) - E\left(\log p(\mathbf{X}, \mathbf{z}; \Theta^{(k)}) \mid \mathbf{z}; \Theta^{(k)}\right) \\
&\equiv Q\left(\Theta; \Theta^{(k)}\right) - Q\left(\Theta^{(k)}; \Theta^{(k)}\right),
\end{aligned} \tag{9}$$

where $\mathbf{X}$ denotes the latent random vector that takes values $\mathbf{x} \in \mathcal{X}$ with probabilities $p(\mathbf{x} \mid \mathbf{z}; \Theta)$, the conditional distribution of $\mathbf{x}$ given the EST $\mathbf{z}$, and E calculates the expectation with respect to $p(\mathbf{x} \mid \mathbf{z}; \Theta)$ of what is typically called the distribution of the *complete data*, $p(\mathbf{x}, \mathbf{z}; \Theta)$. Some intuition is given below. It can be further shown that

$$\max_{\Theta} Q\left(\Theta; \Theta^{(k)}\right) \geq Q\left(\Theta^{(k)}; \Theta^{(k)}\right), \tag{10}$$

so that if $\Theta^{(k+1)}$ denotes the value that maximizes $Q\left(\Theta;\Theta^{(k)}\right)$, then Eq.10 together with Eq.9 imply $L\left(\Theta^{(k+1)}\right) \geq L\left(\Theta^{(k)}\right)$. The EM algorithm amounts to iteratively maximizing $Q\left(\Theta;\Theta^{(k)}\right)$, which by Eq.10 must increase Q monotonically, until convergence to the maximum likelihood.

For an intuitive interpretation, consider the original aim: we want to maximize the likelihood $L(\Theta) = p(\mathbf{z};\Theta)$, or log-likelihood $\log L(\Theta)$. Had we observed the latent variables $\mathbf{x}$, we would instead maximize the *complete data* log likelihood $\log L_{complete}(\Theta) = \log p(\mathbf{x},\mathbf{z};\Theta)$, since $\mathbf{x}$ and $\mathbf{z}$ together contain at least as much information about Θ than $\mathbf{z}$ does alone. But since $\mathbf{x}$ has not been observed, the EM trick consists of replacing $\mathbf{x}$ by $p\left(\mathbf{x} \mid \mathbf{z};\Theta\right)$ in $\log L_{complete}(\Theta)$, where $p\left(\mathbf{x} \mid \mathbf{z};\Theta\right)$ is the distribution of values that $\mathbf{x}$ could have taken to give rise to the EST $\mathbf{z}$ we observed. This replacement amounts to calculating Q, the expectation of $\log L_{complete}(\Theta)$ with respect to $\mathbf{x}$, given $\mathbf{z}$. A nice geometric interpretation is also provided by Neal and Hinton, 1998.

To calculate Q we need $p(\mathbf{z},\mathbf{x};\Theta)$ and $p(\mathbf{x} \mid \mathbf{z};\Theta)$. We now derive these distributions. Because z_t and y_{it} are binary variables, natural statistical models to describe their variations are Bernoulli distributions with probabilities of a spike $\lambda_i(\vec{v}_t;\theta_i)$ and $\kappa(\vec{v}_t;\Theta)$ respectively, that is

$$p(y_{it};\theta_i) = [\lambda_i(\vec{v}_t;\theta_i)]^{y_{it}}\,[1-\lambda_i(\vec{v}_t;\theta_i)]^{1-y_{it}}, \quad y_{it}=0,1, \tag{11}$$

and

$$p(z_t;\Theta) = [\kappa(\vec{v}_t;\Theta)]^{z_t}\,[1-\kappa(\vec{v}_t;\Theta)]^{1-z_t}, \quad z_t=0,1. \tag{12}$$

Note that Eqs.11 and 12 give complete specifications of the regressions in Eqs.1 and 5. As for the joint distribution of the latent variable x_t, it can be reduced to the product of the marginals

$$p(x_t;\Theta) = \prod_{i=1}^{I} p(y_{it};\theta_i), \tag{13}$$

with $p(y_{it};\theta_i)$ given by Eq.11, provided we assume that the neurons are independent. Approaches for the dependent case are considered in the discussion section. Now, just as with $p(\mathbf{z};\Theta)$ in Eq.8, we can reduce $p(\mathbf{z},\mathbf{x};\Theta)$ and $p(\mathbf{x} \mid \mathbf{z};\Theta)$ to the product over time bins of the marginals,

$$p(\mathbf{z},\mathbf{x};\Theta) = \prod_{t=1}^{T} p(z_t,x_t;\Theta) \quad \text{and} \quad p(\mathbf{x} \mid \mathbf{z};\Theta) = \prod_{t=1}^{T} p(x_t \mid z_t;\Theta).$$

Considering first the complete data distribution, we use basic laws of probabilities to write

$$p(z_t,x_t;\Theta) = p(z_t \mid x_t;\Theta)\;\; p(x_t;\Theta).$$

Because a spike is recorded at the electrode if and only if at least one neuron spiked, $z_t = 1$ if and only if $y_{it} = 1$ for some i, so that $p(z_t \mid x_t;\Theta)$ reduces trivially: if x_t is a vector of zeros, then $z_t = 0$ with probability 1, otherwise $z_t = 1$ with probability 1. Hence, the distribution of the *complete data* is

$$p(z_t,x_t;\Theta) = I_{\{x_t \text{ consistent with } z_t\}} p(x_t;\Theta), \tag{14}$$

where $p(x_t;\Theta)$ is given by Eq.13 and I_A is an indicator variable that takes value one if A is true, and zero otherwise. To derive $p(x_t \mid z_t;\Theta)$ 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, if $z_t = 1$ and x_t is not identically zero,

$$p(x_t \mid z_t = 1;\Theta) = I_{\{x_t \text{ consistent with } z_t = 1\}} \frac{p(x_t, z_t = 1;\Theta)}{p(z_t = 1;\Theta)},$$

with the denominator given by the Bernoulli distribution in Eq.12. Because $z_t = 1$ is implied by x_t not identically zero, dropping $z_t = 1$ preserves the probability in the numerator. Putting results together we have

$$p(x_t \mid z_t = 1; \Theta) \quad = \quad I_{\{x_t \text{ consistent with } z_t = 1\}} \ \frac{p(x_t; \Theta)}{\kappa(v_t; \Theta)} \tag{15}$$

with $p(x_t; \Theta)$ in Eq.13 and $\kappa(v_t; \Theta)$ the firing rate induced by the neurons at the electrode in Eq.4. Although we neither observed x_t, nor inferred it by spike sorting, we were able to derive its distribution given the observed EST z_t.

With $p(\mathbf{z}, \mathbf{x}; \Theta)$ and $p(\mathbf{x} \mid \mathbf{z}; \Theta)$, we can proceed with the EM algorithm. We first use Eqs.13 and 14 to rewrite $Q\left(\Theta, \widehat{\Theta}^{(k)}\right)$ in Eq.9 as

$$\begin{aligned} Q\left(\Theta, \widehat{\Theta}^{(k)}\right) &= E\left(\log\left(\prod_{i=1}^{I} p(\mathbf{Y}_i; \theta_i)\right) \mid \mathbf{z}; \widehat{\Theta}^{(k)}\right) && (16) \\ &= \sum_{i=1}^{I} E\left(\log p(\mathbf{Y}_i; \theta_i) \mid \mathbf{z}; \theta_i^{(k)}\right) && (17) \\ &= \sum_{i=1}^{I} Q_i\left(\theta_i, \hat{\theta}_i^{(k)}\right), && (18) \end{aligned}$$

which shows that maximizing Q with respect to Θ is equivalent to maximizing each Q_i with respect to the respective θ_i. This is easy to do, once we recognize that $p(\mathbf{Y}_i; \theta_i)$ in Eq.17 is the distribution of the NST of neuron i, that is the likelihood we would maximize to estimate θ_i, if $\mathbf{y_i}$ has been made available via spike sorting. In other words, if the NST $\mathbf{y_i}$ were known, the value of θ_i that maximizes Q_i would be the ML estimator $\hat{\theta}_i$ obtained by regressing $\mathbf{y}_i$ on $\vec{v}$ as in Eq.1. Because $\mathbf{y}_i$ is unobserved, the EM algorithm requires that we use instead its expectation, $E\left(\mathbf{Y_i} \mid \mathbf{z}; \widehat{\Theta}^{(k)}\right)$, given the EST $\mathbf{z}$ and current value of the parameter $\Theta^{(k)}$. Given $z_t = 0$, we have trivially

$$E\left(Y_{it} \mid z_t = 0; \widehat{\Theta}^{(k)}\right) = 0. \tag{19}$$

Given $z_t = 1$, Y_{it} is Bernoulli with expectation

$$E\left(Y_{it} \mid z_t = 1; \widehat{\Theta}^{(k)}\right) = P\left(Y_{it} = 1 \mid z_t = 1; \widehat{\Theta}^{(k)}\right) = \sum_{x_t : y_{it} = 1} P\left(X_t = x_t \mid z_t = 1; \widehat{\Theta}^{(k)}\right), \tag{20}$$

with probabilities in the summand given by Eq.15, and summation over the 2^{I-1} values of $x_t = (y_{1t}, \ldots, y_{It})$ that have ith component $y_{it} = 1$.

We can now give a version of the EM algorithm specifically tailored to our goal of fitting the neuron's tuning curves without spike sorting, which we refer to as the EST encoding algorithm. It is an exact expectation EM rather than the most common stochastic EM; it is computationally very fast.

EST encoding algorithm
Input: The EST $\mathbf{z}$
Initialize $\Theta = \Theta^{(k)}; k = 0$

(E-step) Compute the expected spike train for neuron i, $i = 1, \ldots, I$

$$\mathbf{e}_i^{(k)} = E\left(\mathbf{Y_i} \mid \mathbf{z}; \mathbf{\Theta^{(k)}}\right) \tag{21}$$

using Eqs.19 and 20.

(M-step) Regress $\mathbf{e}_i^{(k)}$ on $\vec{v}$ to obtain the ML estimator $\hat{\theta}_i^{(k+1)}$, where θ_i parametrizes the tuning function $\lambda_i(\vec{v}, \theta_i)$ of neuron i.

Let $k \leftarrow k + 1$ and iterate until convergence

2.1.1 Determining the number of neurons recorded by the electrodes

So far we have concentrated on one representative electrode, and have assumed that it records I neurons; I is usually known as a byproduct of spike sorting. Here we propose a spike sorting free alternative that uses classic results of likelihood theory.

For a fixed number of neurons I, the EST encoding algorithm yields the MLE $\widehat{\Theta}$, the value that maximizes $L(\Theta)$. As we increase I, $L(\widehat{\Theta})$ also increases. This is well known: the larger the model is, the better it fits the data. Therefore $L(\widehat{\Theta})$ cannot be used as a criterion for model selection since the largest model would always be selected. This is a common problem to which several solutions exist. The likelihood ratio test (LRT) allows "formal" comparisons of two nested models, by capping at a prespecified $\alpha\%$ the probability of rejecting the small model by mistake. Two models are nested if one is a particular case of the other; for example, the two-neuron model $\Theta = (\theta_1, \theta_2)$ is a special case of the three-neuron model $\Theta = (\theta_1, \theta_2, \theta_3)$ when $\theta_3 = 0$. The Akaike's information criterion (AIC) and the Bayesian inference criterion (BIC) allow comparisons of models that are not necessarily nested. Both consist of assigning a score to each model of the form "goodness of fit" minus "complexity", specifically $AIC(\Theta) = \log L(\widehat{\Theta}) - dim(\Theta)$ and $BIC(\Theta) = \log L(\widehat{\Theta}) - \frac{dim(\Theta)}{2} \log n$. The AIC and BIC do not control the probabilities of making mistakes. Instead, the model with highest AIC minimizes the expected Kullback-Leibler distance between true and chosen models, while the model with highest BIC has highest posterior probability, when a uniform prior on the space of models considered is used. The usual procedure for AIC and BIC is to choose a range of values for I, obtain scores for all models, and retain the model with the highest score. Instead we adopt a greedier procedure that gives the same results while saving time; for each electrode, we proceed sequentially, testing first the "no neuron" versus the one neuron model, the one versus the two neurons model, and so on until the larger model provides no significant improvement over the smaller. The procedure follows.

Determining the number of neurons
Initialize: $I = 0$

- Let Θ_{small} be the model with I neurons, and Θ_{big} the model with $I+1$ neurons, with MLEs $\widehat{\Theta}_{small}$ and $\widehat{\Theta}_{big}$ obtained by the EST encoding algorithm.

- Reject the model with I neurons in favor of the model with $I+1$ neurons if

$$\log L(\widehat{\Theta}_{big}) - \log L(\widehat{\Theta}_{small}) > \text{Critical value}$$

with

$$\text{Critical value} \quad = \quad \begin{cases} \textbf{(LRT)} \ \chi^2_{q,(1-\alpha)}/2 \\ \textbf{(AIC)} \ \text{q} \\ \textbf{(BIC)} \ (\text{q}/2) \log n \end{cases}$$

where $q = dim(\Theta_{big}) - dim(\Theta_{small})$, $\chi^2_{q,(1-\alpha)}$ is the $(1-\alpha)$th quantile of the Chi-square distribution with q degrees of freedom, and n is the size of the data.

Let $I \leftarrow I + 1$ and iterate until the smaller model is retained

Although each procedure has a different theoretical justification, in practice they only differ in the amount of evidence required to favor large over small models. The larger the critical value is, the more evidence is needed to favor the large model, so that AIC yields larger models than BIC.

2.1.2 Identifiability

In simplified terms, the parameters of the tuning curves, θ_i, are unidentifiable if they cannot be estimated uniquely. In our application, identifiability issues can arise for two reasons, *data identifiability* and *model identifiability*.

Model unidentifiabilities happen when neurons have overlapping tuning curves, because it is difficult to untangle neurons based only on what is observed at the electrode. This situation is analogous to overlapping waveform-feature clusters in the spike sorting context. In that case, it may be that several values of Θ maximize the likelihood $L(\Theta)$; one of these values corresponds to the actual neurons recorded by the electrode, if the spike-rate models are not too badly mis-specified, while the others correspond to imaginary neurons whose combined activity is not distinguishable from that of the actual neurons. The particular estimate of Θ we obtain depends on the initial values; the same would happen with any other likelihood maximization procedure. This type of unidentifiability is likely to happen often in practice. However, the simulation study clearly shows that this does not affect the quality of decoding. It makes sense too: if we use imaginary neurons whose combined activity is the same as that of the actual neurons, none of the information collected at the electrode is lost. For example, if all neurons have the same tuning curves, our algorithm will detect only one imaginary neuron, with firing rate the aggregate of the actual neurons' firing rates; this neuron contains all the information about movement variables the electrode provides.

Data unidentifiability has to do with how the *true* tuning curves of the neurons, λ_i^* say, relate to the rate

they induce at the electrode, $\kappa^* = 1 - \prod_i (1 - \lambda_i^*)$. By *true* we refer to the unknown mechanism that generates the spike trains, rather than to a model like Eq.2 we fit to them, which is our best attempt at capturing the variations we observe. If λ_i^* and κ^* have the same functional form, then it is impossible to distinguish the activity of the electrode from the activity of one, or combined activities of two or more neurons. This is the case, for example, if all λ_i^* are constant. But if $\lambda_i^* = \theta_i ||v||$, say, with $||v||$ the velocity magnitude, then κ^* is a polynomial in $||v||$ of order I, so that the activity recorded at the electrode is qualitatively different from the activity of single neurons, and an algorithm like ours can detangle neurons from electrodes. Given the non-linear relationship between κ^* and λ_i^*, it is hard to imagine situations other than the trivial constant firing rate case that would yield data unidentifiabilities[2]. But because λ_i^* is the firing rate per millisecond, it is small enough that κ^* might be well approximated by $\kappa^* \equiv \sum \lambda_i^*$, so that λ_i^* linear in movement variables could yield a κ^* that is close to linear in the same variables; this includes the commonly used cosine tuning. However our simulation study shows that even exactly cosine neurons do not cause our algorithm to degrade.

2.2 Spike sorting free decoding

Now that estimated tuning curves $\hat{\lambda}_i(\vec{v}) = \lambda(\vec{v}; \hat{\theta}_i)$ are available, we turn to velocity predictions from ESTs. We previously focused on one representative electrode. Here we work with a population of J electrodes, from which N neurons are recorded. The EST of electrode j is denoted by $\mathbf{z_j}$ and its estimated firing rate by $\hat{\kappa}_j(\vec{v}) = \kappa_j(\vec{v}; \widehat{\Theta}) = 1 - \prod_{i \in I_j} (1 - \hat{\lambda}_i(\vec{v}))$, where I_j is the set of indices of the neurons recorded by electrode j, while the subscript j_i identifies the electrode that records neuron i.

The population vector (PV) (Georgopoulos *et al.*, 1986) predicts velocity at time t by a normalized version of

$$\vec{P}_t^{PV} = \sum_{i=1}^{N} y_{it} \, \vec{D}_i, \tag{22}$$

a linear combination of the neurons' preferred directions (PD) $\vec{D}_i$ and their spike counts y_{it}, where $\vec{D}_i$ is the unit-magnitude vector whose direction maximizes $\hat{\lambda}_i(\vec{v})$. We propose two estimators related to Eq.22 that do not require that the NSTs $\mathbf{y_i}$ be known. The naive EST prediction,

$$\vec{P}_{Ut}^{PV} = \sum_{j=1}^{J} z_{jt} \, \vec{E}_j \tag{23}$$

is the usual PV prediction obtained by treating the electrodes as if they were neurons with tuning curves $\kappa_j(\vec{v})$. The subscript "U" stands for unsorted and $\vec{E}_j$ is the unit-magnitude vector whose direction maximizes $\hat{\kappa}_j(\vec{v})$. In the rest of the paper we refer to $\vec{P}_U$ simply as the naive prediction. The (non-naive) EST prediction

$$\vec{P}_{Et}^{PV} = \sum_{i=1}^{N} e_{it} \, \vec{D}_i \tag{24}$$

has the same form as Eq.22, but with spike counts y_{it} replaced by their conditional expectations

$$e_{it} = E(y_{it} \mid z_{j_i t}) = z_{j_i t} \, \frac{\hat{\lambda}_i(\vec{v}_t)}{\hat{\kappa}_{j_i}(\vec{v}_t)}, \tag{25}$$

[2]Solutions to this problem may exist within group theory. A rigorous proof seems too difficult to attempt here.

defined earlier in Eq.6. For single-neuron electrodes, $e_{it} = y_{it}$. Otherwise, an electrode that records several neurons has

$$\sum_{i \in I_j} e_{it} \geq z_{jt},$$

which mirrors the inequality we would get from spike sorted data, $\sum_{i \in I_j} y_{it} \geq z_{jt}$. Both inequalities are consistent with $\kappa_j(\vec{v}) \leq \sum_{i \in I_j} \lambda_i(\vec{v})$ and account for neurons spiking together. Note that Eqs.25 and 21 appear related, but are equal only when (i) the EST encoding algorithm has converged so that $\Theta^{(k)} = \widehat{\Theta}$ in Eq.21 and (ii) they are both conditional on the ESTs used to obtain $\widehat{\Theta}$. We show in the next section that $\vec{P}_E^{PV}$ yields the same mean prediction as $\vec{P}^{PV}$, but that its variance is smaller.

Maximum likelihood (ML) methods rely on a statistical model that specifies the probability distribution of the spike trains, and the ML prediction is the value that maximizes their joint distribution, that is the likelihood. For example, if we assume that the spike counts y_{it} have Poisson distributions with means the firing rates $\lambda_i(\vec{v}_t)$, then, assuming that the neurons are independent, the likelihood is

$$L(\vec{v}) = \prod_{i=1}^{N} \frac{[\lambda_i(\vec{v})]^{y_{it}} \; e^{-\lambda_i(\vec{v})}}{y_{it}!},$$

and the ML estimate of velocity at t is

$$\vec{P}_t^{\mathrm{ML}} = \underset{\vec{v}}{\operatorname{argmax}} \, L(\vec{v}). \tag{26}$$

Just as with PV predictions, we define two alternative ML predictions that do not require the spike sorted data. They are the naive prediction

$$\vec{P}_{Ut}^{\mathrm{ML}} = \underset{\vec{v}}{\operatorname{argmax}} \prod_{j=1}^{J} \frac{[\kappa_j(\vec{v})]^{z_{jt}} \; e^{-\kappa_j(\vec{v})}}{z_{jt}!}, \tag{27}$$

which treats electrodes as if they were neurons, and the EST prediction

$$\vec{P}_{Et}^{\mathrm{ML}} = \underset{\vec{v}}{\operatorname{argmax}} \prod_{i=1}^{N} \frac{[\lambda_i(\vec{v})]^{e_{it}} e^{\lambda_i(\vec{v})}}{e_{it}!}, \tag{28}$$

based on the expected NSTs in Eq.25. We wrote the right hand side of Eq.27 as a product over the electrodes because, under the assumption that the neurons are independent and that each neuron is recorded by only one electrode, the ESTs $\mathbf{z_j}$ are also independent.

The predictions in Eqs.24 and 28 are straightforward in principle, but they assume that the conditional expected spike counts in Eq.25 are known. For single-neuron electrodes, we have trivially $e_{it} = z_{j_i t}$, the observed electrode's spike counts. Otherwise e_{it} is a function of the very velocity $\vec{v}_t$ we seek to predict. The obvious solution is to replace $\vec{v}_t$ in Eq.25 with an estimate, which we denote by $\hat{v}_t$ to differentiate it from the velocity predictions we have denoted so far by $\vec{P}$, with various sub- and sup-scripts. Several options are available. We consider

$$\hat{v}_t = k^{-1} \sum_{i=0}^{k-1} \vec{P}_{U(t-i)}, \tag{29}$$

the average of the current and $(k-1)$ previous naive predictions, and

$$\hat{v}_t = (k^{recur})^{-1} \sum_{i=1}^{k^{recur}} \vec{P}_{E(t-i)}, \tag{30}$$

(A) $\vec{P}_E$ (B) $\vec{P}_E^{recur}$

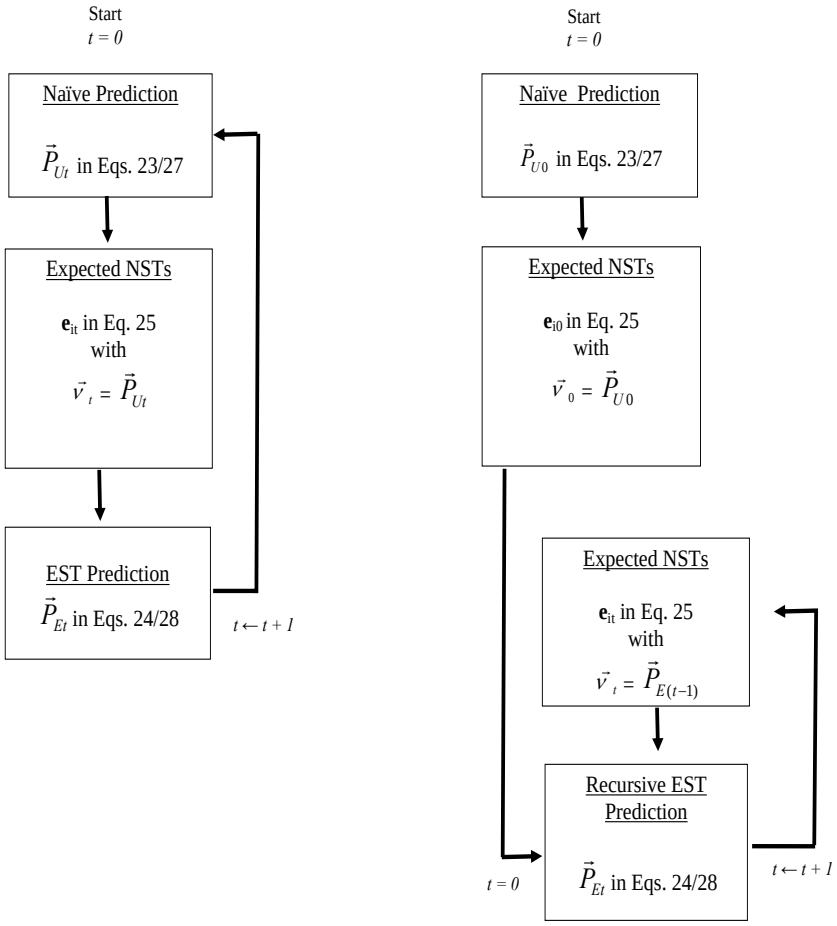


Figure 3: *Proposed EST predictions. (A) $\vec{P}_E$ uses the naive prediction $\vec{P}_U$ to calculate the conditional expected NSTs e_i in Eq.25. (B) Recursive prediction $\vec{P}_E^{recur}$ uses $\vec{P}_U$ for the first decoding time only. For $t > 0$, conditional expected NSTs are calculated based on past predictions $\vec{P}_E^{recur}$.*

the average of the k^{recur} previous EST predictions. Replacing $\vec{v}_t$ by Eqs.29 or 30 makes sense only under the assumption that $\vec{v}_t$ evolves in time with some degree of smoothness, as is the case for real movements. Use of Eq.30 produces a purely recursive algorithm, since past predictions are used to calculate e_{it}, which are in turn used to produce the next prediction. We therefore refer to Eq.28 together with Eq.30 as the recursive EST prediction $\vec{P}_{Et}^{recur}$. Fig.3 gives a flowchart summary of our proposed EST predictions, valid for PV and ML methods.

The price to pay for replacing $\vec{v}_t$ in Eq.25 by an estimate is bias. Eqs.29 and 30 yield increasingly biased estimates of $\vec{v}_t$ with increasing values of k and k^{recur}, which in turn induces bias in e_{it}, so that $e_{it} = E(y_{it} \mid z_{j_i t})$ no longer holds exactly. This puts into question the use of e_{it} in place of y_{it} to make predictions. The only bias free scenario happens with use of Eq.29 with $k = 1$, since $\vec{P}_{Ut}$ (Eqs.23 and 27) is unbiased for $\vec{v}_t$. But as we show later, $\vec{P}_{Ut}$ has large variance, so that the e_{it} have large variances too, which is bound to degrade the efficiency of $\vec{P}_E$. We therefore need to investigate values of the bias parameters k and k^{recur} that strike a good balance between small variability and small bias. This balance is also a function of the proportion of single-neuron electrodes. Indeed, if all electrodes are single-neuron,

then $e_{it} = y_{it} = z_{j_i t}$ for all i, and the choice of estimate of $\vec{v}_t$ in Eq.25 is irrelevant since naive and EST predictions reduce to the usual NST predictions (Eqs.22 and 26). But if too few electrodes record only one neuron, the expected counts might be too variable (if Eq.29 with $k = 1$ is used) or too biased (if larger k or k^{recur} are used) to yield efficient movement predictions.

At this point we have proposed a complete spike sorting free method for encoding and decoding movement. Our next step is to evaluate how good the method is. Efficiency comparisons between proposed and traditional approaches are difficult, except for PV predictions under simplifying assumptions. We report this in the next section. We subsequently describe a simulation study designed to compare the efficiencies of PV and ML predictions under general conditions.

2.2.1 Analytical comparisons of efficiencies of PV predictions

We compare the PV predictions in Eqs.22, 23 and 24, and show that, under the simplifying assumptions specified below, $\vec{P}_U^{PV}$ is less efficient than the traditional NST prediction $\vec{P}^{PV}$, while $\vec{P}_E^{PV}$ is more efficient. This suggests that spike sorting can be avoided without loss of efficiency.

For clarity, we drop the time subscript t and the superscript PV, since we discuss only PV predictions. To make analytical calculations possible, we simplify the relationships between electrodes and neurons' spike counts and firing rates, and replace Eq.4 by

$$\kappa_j(\vec{v}) \quad = \quad \sum_{i \in I_j} \lambda_i(\vec{v}), \tag{31}$$

which effectively assumes that neurons recorded at an electrode do not spike together. This further implies

$$z_{jt} \quad = \quad \sum_{i \in I_j} \; y_{it}. \tag{32}$$

This is not too strong a simplification given that the firing rates are fairly low. We also assume that $\vec{v}_t$ is known in Eq.25. The more realistic setting where $\vec{v}_t$ is replaced by Eq.29 or 30 is treated later by simulation.

It is easy to show that $\vec{P}$ and $\vec{P}_E$ have the same expectation

$$E(\vec{P}) = E(\vec{P}_E) = \sum_{i=1}^{N} \lambda_i(\vec{v}) \vec{D}_i,$$

which means that they yield the same movement predictions on average. Details are in the appendix. The variance-covariance matrices of $\vec{P}$ and $\vec{P}_E$ cannot be compared without imposing any constraint on $\sigma_i^2 = \mathrm{Var}(y_i)$. It is not too restrictive to assume that $\mathrm{Var}(y_i) = \alpha \; \lambda_i(\vec{v}), \alpha \in R^+$, which is to say that the variance of spike counts is proportional to their means. This happens, for example, when spike counts are over or under dispersed Poisson random variables. The case $\alpha = 1$ corresponds to Poisson spike trains. Under this assumption, we show in the appendix that

$$\text{Var-Cov}(\vec{P}_E) \leq \text{Var-Cov}(\vec{P}).$$

That is, traditional NST prediction $\vec{P}$ and proposed EST prediction $\vec{P}_E$ have the same expectation, but $\vec{P}_E$ has variances and covariances always smaller than those of $\vec{P}$. This result makes sense; $\vec{P}$ and $\vec{P}_E$ are

both calculated given the observed electrodes' spike counts z, but $\vec{P}_E$ uses e_i, the expectation of y_i given z_{j_i}, which removes from $\vec{P}_E$ the variability of y_i about its expectation.

Although more intuitive, it is also more difficult to show that $\vec{P}_U$ is inferior to $\vec{P}$, because the two predictions are not equal on average, either in direction or in magnitude. To see this, consider electrode j; its contribution to $\vec{P}_U$ is in the direction of $\vec{E}_j$, while its contribution to $\vec{P}$ or $\vec{P}_E$ is in the direction of $\sum_{i \in I_j} \lambda_i \vec{D}_i$; these two directions are unlikely to be equal. Rather than an analytical comparison, we provide an intuitive argument, and for illustrative purposes assume that neurons are cosine with rates

$$\lambda_i(v) = k_i + m_i \vec{v} \cdot \vec{D}_i,$$

where $k_i \geq 0$, $m_i \geq 0$ and $\vec{D}_i$ is its unit-length PD. From Eq.31, the electrode that records these neurons is also approximately cosine with rate

$$\kappa_j(\vec{v}) = K_j + M_j \, \vec{E}_j . \vec{v},$$

where $K_j = \sum_{i \in I_j} k_i$, $\vec{E}_j = \sum_{i \in I_j} m_i \, \vec{D}_i / M_j$ is the unit-length electrode's PD, and $M_j = \| \sum_{i \in I_j} m_i \, \vec{D}_i \|$. We compare $\vec{P}$ and $\vec{P}_U$ based on the idea that they can be considered predictions from different sets of neurons, and that better neurons yield better predictions. Good neurons are typically well modulated. Taking as a measure of modulation the difference between maximum and minimum firing rates, the modulations of neuron i and electrode j are $2m_i \, \|v\|$ and $2M_j \|v\|$ respectively. Because the PDs are unit vectors, Cauchy-Schwartz inequality yields $M_j = \| \sum_{i \in I_j} m_i \, \vec{D}_i \| \leq \sum_{i \in I_j} m_i$, with equality if and only if all $\vec{D}_i$'s point in the same direction. This shows that the modulation of any electrode is less than the sum of the modulations of the neurons it records, unless they all have the same PD. Although this statement was proved for cosine tuned neurons, it is likely to apply generally. In the appendix, we provide an additional argument, based on a movement prediction perspective, that helps explain why decoding from isolated neurons is better than decoding from electrodes.

We conclude from this section that the proposed EST prediction $\vec{P}_E^{PV}$ is superior to the traditional PV prediction, under minimal assumptions about the variances of the spike counts. On the other hand, intuitive arguments suggest that the naive prediction $\vec{P}_U^{PV}$ is not as efficient. These results were obtained assuming that $\vec{v}_t$ was know in Eq.25. In practice however, e_i will depend on an estimate of $\vec{v}_t$ (Eqs.29 or 30), which will increase either its bias, its variance, or both, so some efficiency will be lost. We investigate under which conditions $\vec{P}_E^{PV}$ remains acceptably efficient in the simulation study.

2.3 The low signal to noise ratio case

Imagine that an electrode records tuned neurons, but that the SNR is low, so that waveforms and noise have comparable amplitudes. Setting the threshold high means that many true spikes will be missed, while setting the threshold lower means that more true spikes, but also more "noise spikes", will be detected. It may then be difficult to separate waveforms from noise, in which case spike sorting might prove too difficult and the electrode discarded. This is a common problem.

Our method handles the low SNR case seamlessly, by treating noise spikes as if they were produced by a "noise" neuron, whose firing rate is a constant with respect to movement variables. All we have to do

is include a flat tuning curve $\lambda_i(\vec{v};\theta_i) = \theta_{i0}$ to be fitted to the electrode in the EST encoding algorithm. Our algorithm is designed to handle joint spiking so that real spikes will be retrieved even if they occur jointly with noise. This is illustrated in the simulation study. Note that fitting a noise neuron can be done either if the ESTs are contaminated by noise or not. Indeed we can test the statistical significance of noise, testing the model with just a noise neuron (one parameter θ_{0i}) versus the model with one neuron (three parameters with use of firing rates in Eq.34), then the models with one neuron versus one neuron and noise, the models with one neuron and noise versus two neurons, etc.

In the decoding stage, noise neurons do not contribute to velocity predictions. To see this, consider for example the ML prediction in Eq.28. Since the firing rates of noise neurons do not depend on $\vec{v}$, the likelihood can be decomposed into the product of noise and other neurons, so that the velocity prediction is

$$\begin{aligned} \vec{P}_{Et}^{\mathrm{ML}} &= \underset{\vec{v}}{\mathrm{argmax}} \left(\prod_{\text{noise neurons}} \frac{\lambda_i^{e_{it}} e^{\lambda_i}}{e_{it}!} \prod_{\text{tuned neurons}} \frac{[\lambda_i(\vec{v})]^{e_{it}} e^{\lambda_i(\vec{v})}}{e_{it}!} \right) \\ &= \underset{\vec{v}}{\mathrm{argmax}} \left(\prod_{\text{tuned neurons}} \frac{[\lambda_i(\vec{v})]^{e_{it}} e^{\lambda_i(\vec{v})}}{e_{it}!} \right). \end{aligned}$$

It does not depend on noise.

2.4 Simulation study

Earlier, we provided some analytical results to compare the efficiencies of PV predictions based on traditional and proposed paradigms. ML predictions do not allow for tractable analytical results, so we compare paradigms based on a simulation study similar to Brockwell *et al.* (2004). We simulated spike trains for N neurons, assuming the modified cosine tuning functions

$$\lambda_i^*(\vec{v}_t) = g(k_i + m_i \vec{v}_t \cdot \vec{D}_i), \tag{33}$$

where $\vec{v}_t$ is the value of the two-dimensional velocity at time t, k_i and m_i are positive constants determining base firing rate and directional sensitivity, and $\vec{D}_i$ is the unit-length PD. The function g was used to sharpen tuning curves more or less. We used $g(x) = x^a$ with $a < 1$ producing tuning curves broader than cosine, and $a > 1$ sharpening them more; $g(x) = \exp(x)$ was also used to produce sharp tuning. Each neuron was assigned a random PD $\vec{D}_i$ and random values of k_i and m_i, so that its minimum firing rate was positive and maximal firing rate between 80 and 100 Hertz. These rates are roughly consistent with M_1 data. Given the velocities, the spike counts were taken to have Poisson distributions with mean in Eq.33. To create ESTs, we randomly assigned the N neurons onto J electrodes so that all electrodes would record at least one neuron. To create the low SNR case we added noise spikes at the rate of 100 Hz to all electrodes. Because the maximum firing rate of all neurons was set between 80 and 100 Hz, many electrodes recorded more noise spikes than real spikes.

In implementing ML decoding, we assumed Poisson spike trains with exponential cosine firing rate

$$\lambda_i(\vec{v};\theta_i) = \exp(\theta_{0i} + \theta_{1i} v_x + \theta_{2i} v_y). \tag{34}$$

Unless $g = \exp$ in Eq.33, the model we fit to the data is different from the generative model, a realistic scenario since in real applications we are unlikely to use the "true" model. Parameters $\theta_i = (\theta_{0i}, \theta_{1i}, \theta_{2i})$

in Eq.34 were estimated by ML based on data from the generative model in Eq.33, obtained by doing four loops of the velocity trajectory specified below. The data used to fit Eq.34 were then discarded, and the fitted firing rates $\hat{\lambda}_i$ used for movement predictions based on fresh ESTs from the generative model.

Naive velocity predictions prescribe that we use $\hat{\kappa}_j(\vec{v})$ for the electrodes' estimated firing rates. This gave substantially biased predictions. To understand why, imagine an electrode that records two neurons with PDs 0 and 180 degrees. Observing a large electrode spike count suggests that $\vec{v}$ is near 0 or 180 degrees. A maximum likelihood approach forces us to choose one of the two values, which does not summarize the information adequately. A better summary would be the conditional distribution of $\vec{v}$ given the observed spike count and firing rates, which is the output of Bayesian decoding algorithms. Dynamic Bayesian decoding is treated in a forthcoming publication. To circumvent this problem here, we estimated the electrodes' firing rates by fitting the model in Eq.34 to the ESTs.

Two velocity trajectories were used. For the decoding study we assumed that the two-dimensional velocity $\vec{v}_t$ traces out the path in Fig.7 over the course of 12 s, with path defined by $x_t = 6\cos(\pi t/6), y_t = 2\sin(\pi t/2)$, for $t \in [0, 12]$, and velocity defined by the respective derivatives. To illustrate the properties of the EST encoding algorithm, we used a simpler circular velocity trajectory over the course of the 12 seconds, with path $x_t = 12\cos(\pi t/12), y_t = 12\sin(\pi t/12)$, for $t \in [0, 12]$. For this path the velocity amplitude remains constant and tuning curves can therefore be displayed as functions of direction on a circular plot. We found circular plots to be clear and visually appealing.

Simulations were based on independent data sets of N neurons, randomly assigned to J electrodes. We mainly used $N = 80$ and $J = 40$ but other values were considered as needed. The 12 s in the experiment were divided into 400 time bins of 30 ms, and velocity predictions obtained for each dataset using all methods. We assess the quality of decoded velocities by the integrated squared error (ISE), defined as the average over all 400 time bins of the squared difference between decoded and actual velocities. The ISE is a combined measure of bias and variance. To compare the efficiencies of predictions from ESTs and NSTs, we calculate, for each dataset, the ratio of the ISEs of EST decoded trajectories over the ISEs of the NST predictions. An ISE ratio below one indicates that the spike sorting free approach is more efficient than the traditional approach. Because ISE ratios vary from dataset to dataset, we summarize their values across many simulated samples using a box plot, which shows the quartiles (25th, 50th and 75th quantiles) as a box, with whiskers extending on each side to the 2.5th and 97.5th quantiles. Box plots are an effective way to visually compare several distributions.

3 Results

We split this section in two parts. The first illustrates how the EST encoding algorithm works. The second presents the results of the decoding simulation study.

3.1 Illustrations of the EST encoding algorithm

Consider an electrode that records $I = 3$ neurons, whose true rates are exponential cosine, $\lambda_i^*(\vec{v}) = \exp(k_i + m_i\vec{v}\cdot\vec{D}_i)$ with preferred directions $\vec{D}_i = 0$, 90 and 180 degrees. They are shown as bold dashed curves in Figs.4BCD. Note that the generative model matches the decoding model of Eq.34, which ensures

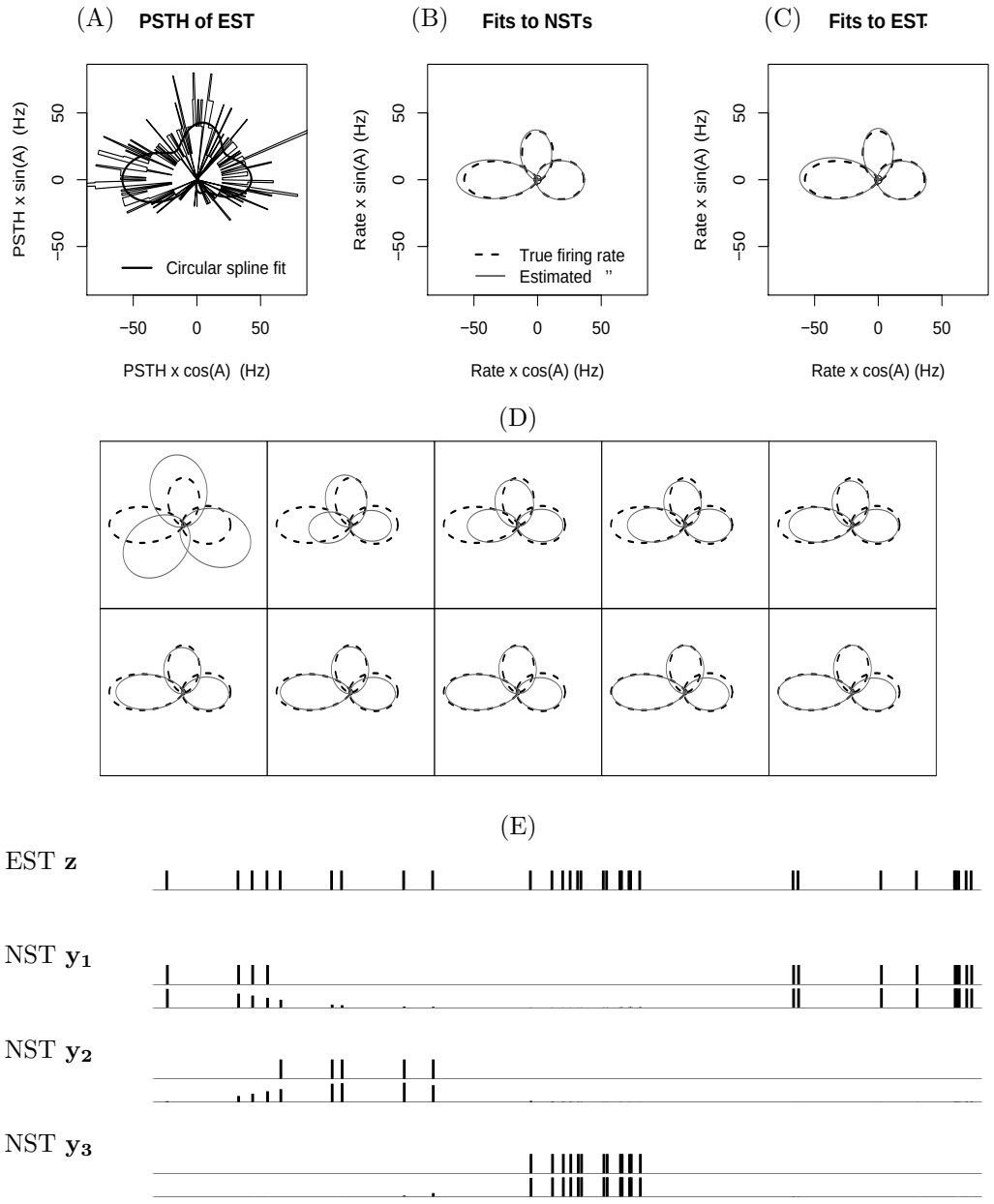


Figure 4: *(A) Circular PSTH (PSTH wrapped around the origin) of the EST, with circular spline fit overlaid; $A = \arctan(v_x/v_y)$ is the direction of the constant magnitude velocity $\vec{v}$. The spline fit reveals 3 bumps that suggest that 3 or more unimodal neurons are recorded by this electrode. The EST is indeed the aggregate of $I = 3$ NSTs simulated from Poisson neurons with rates shown in (B,C,D) as dashed curves. (B) True firing rates λ_i^*, and rates fitted to the NSTs. (C) True rates and rates estimated by the EST encoding algorithm. The fits in (B) and (C) are almost identical. (D) First 10 iterations of the algorithm. The dashed curves are the true rates λ_i^* and the solid curves their current estimates. The first panel shows starting values, which we took to have PDs equally spaced on [0,2π]. All starting values (random shapes, sizes, placements) converged to the same solution. (E) Observed EST $\mathbf{z}$, and unobserved NSTs $\mathbf{y_i}$ of the 3 neurons. The spike trains shown below each NST are the conditional expected NSTs in Eq.21 after the EST encoding algorithm has converged.*

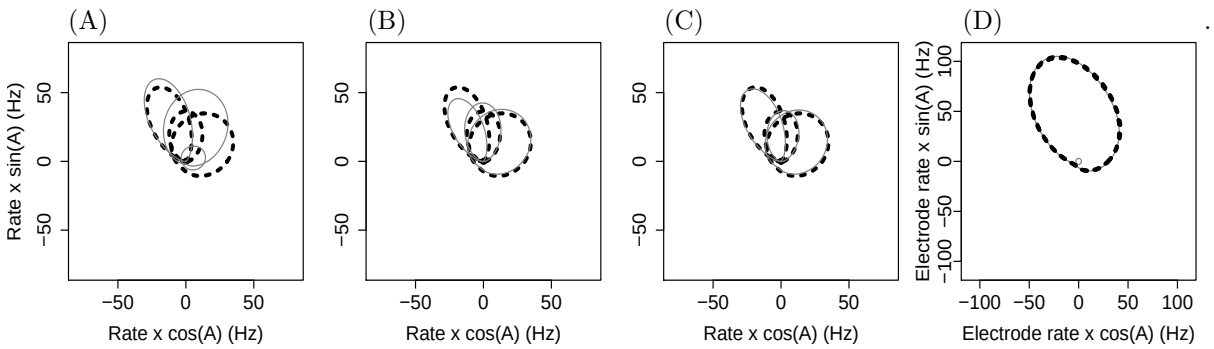


Figure 5: *(A) Fitted rates (solid curves) obtained by the EST encoding algorithm applied to an electrode that records $I = 3$ Poisson neurons with rates shown in bold dashed. Because true rates overlap significantly, true and fitted rates do not match. (B,C) Same as (A), for 2 other random starts of the algorithm. (D) Firing rate of the electrode. The true rate is in dashed bold. The estimated rates from panels (A,B,C) are drawn in solid. All rates overlap and so are not distinguishable by the naked eye.*

that Fig.4 shows the properties of the EST encoding algorithm without corruption from a bad model choice. The simple circular velocity path was used to allow for circular graphical displays. Fig.4A shows a PSTH of the EST, from which we can see three modes, which suggest that the electrode records at least three unimodal neurons that are tuned to velocity. It is that information our algorithm uses to determine the number of neurons and their tuning curves. In practice we will not be able to identify neurons with the naked eye; see, for example, Figs.5 and 6. Fig.4B shows the tuning curves fitted to the NSTs $\mathbf{y_i}$, and Fig.4C the tuning curves obtained by running the EST encoding algorithm on the EST $\mathbf{z}$; the fits are practically identical. Fig.4D shows the initial values and the first nine iterations of the algorithm. Fig.4E shows a portion of the EST $\mathbf{z}$, the corresponding portions of the three unobserved NSTs $\mathbf{y_i}$, as well as the conditional expected NSTs $\mathbf{e_i}$ in Eq.21, calculated after the algorithm converged. We see that $\mathbf{e_i}$ is close to $\mathbf{y_i}$, although $\mathbf{e_i}$ is probabilistic in nature and thus contains full and partial spikes. Fig.4E shows that, although we do not spike sort based on waveform information, the EST encoding algorithm yields spike sorted trains as a byproduct of encoding. This could be used to improve spike sorting, as mentioned in the discussion.

In this example, as in all situations where true tuning curves do not overlap much, the EST encoding algorithm yields fitted tuning curves that are similar to those obtained by the common NST based procedure. The ideal situation breaks down when tuning curves overlap significantly, as shown in Fig.5. We simulated the spike train of an electrode recording $I = 3$ neurons with exponential cosine tuning curves as above, but more clustered PDs at 45, 90 and 112 degrees. The PSTH of the EST (not shown) revealed just one bump, while the neuron number testing procedure described earlier estimated the correct number of neurons. Figs.5ABC show the fitted tuning curves obtained with three random starts of our algorithm. Although they seldom match the true curves, the maximum log likelihood achieved in all cases is $\log L(\widehat{\Theta}) = -1402$, which means that the three solutions shown fit the observed EST equally well. This is further confirmed visually in Fig.5D, which shows the fitted rate at the electrode, $\hat{\kappa} = \kappa(\vec{v}, \widehat{\Theta})$, for the three EM runs, as well as the true κ^*: the four curves are not distinguishable. Fig.5 illustrates what we called earlier model unidentifiability. We show later in the simulation study that model unidentifiabilities do not affect decoding efficiency.

3.1.1 Separating noise from true spikes

Consider an electrode that records one neuron tuned to movement, with tuning curve λ^* shown in Fig.6CD, and whose waveform has maximum amplitude normally distributed with mean 3 and variance 1. Assume that the noise on that electrode is normally distributed with mean 0 and variance 1. 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 159 Hertz[3], also shown in Fig.6C, a much higher rate than that of the neuron; the proportions of real versus noise spikes is 22% versus 78%. Fig.6A shows a circular PSTH of the EST, from which it is hard to tell if the electrode records tuned neurons. Fig.6B shows a histogram of the recorded spikes amplitudes; the real spikes lay below the normal distribution with mean 3 and variance 1 overlaid on the plot. It would be difficult to spike sort this data based on maximum waveform amplitude. Fig.6D shows four solutions of the EST encoding algorithm, corresponding to the four clove-shaped initial values overlaid. The algorithm also converges to the solution in the 2nd panel when we used the true rates as initial values. Unless we start the algorithm with the correct proportion of noise (about 80%; second panel), $\hat{\lambda}$ does not match the true λ^*. Such model unidentifiabilities were expected since the noise rate completely overlaps the rate of the neuron (Fig.6C). However all solutions achieve the same maximum log likelihood, $\log L(\widehat{\Theta}) = -2427.5$, which means that they fit the noise corrupted EST equally well. Moreover, $\hat{\lambda}$ is approximately proportional to λ^* in each case, which means that the algorithm estimates a neuron with the correct qualitative properties, which effect on decoding will be similar to that of the actual neuron.

Rather than illustrate that noise and spikes can be separated perfectly, Fig.6 suggests that adding a noise neuron to be fitted separates the tuned from the untuned portions of the EST, the untuned portion being composed of noise and/or untuned neurons, but potentially also of portions of tuned neurons.

A last comment on Fig.6 concerns joint spiking. The neuron and the noise produced 2205 and 7965 spikes respectively; on 348 occasions they occurred simultaneously. This makes for a total of 10170 spikes, with only 9822 detected at the electrode due to joint occurrences. The proportion of the neuron's spikes corrupted by noise is a non-negligible 16%; they might be difficult to retrieve via spike sorting. On the other hand, our algorithm is designed to handle joint spiking. For the solution in the 2nd panel of Fig.6D, our algorithm retrieved a total of 10159 spikes, close to the actual number (10170), even though its input EST contained only the 9822 recorded spikes.

3.1.2 Determining the number of neurons

For each electrode, we must determine if it records tuned neurons and if so how many. The procedure described earlier consists of comparing the increase in log likelihood to AIC, BIC, or LRT critical values, for increasingly large models, and stop when the increase is no longer significant.

Table 1 gives the maximum log likelihood achieved by fitting I neurons to the data of Fig.4 , for $I = 0, \ldots 5$. We first determine if the electrode records any neuron that is tuned to movement by comparing the models with $I = 0$ and $I = 1$. The former fits a constant firing rate to the EST, so $dim(\Theta_{small}) = 1$, and $dim(\Theta_{big}) = 3$ for the one neuron model in Eq.34; this gives $q = dim(\Theta_{big}) - dim(\Theta_{small}) = 2$. The

[3]The probability of a spike in a 1 msec bin is $\Pr(Z > 1) = 0.159$ where Z is Normal(0,1).

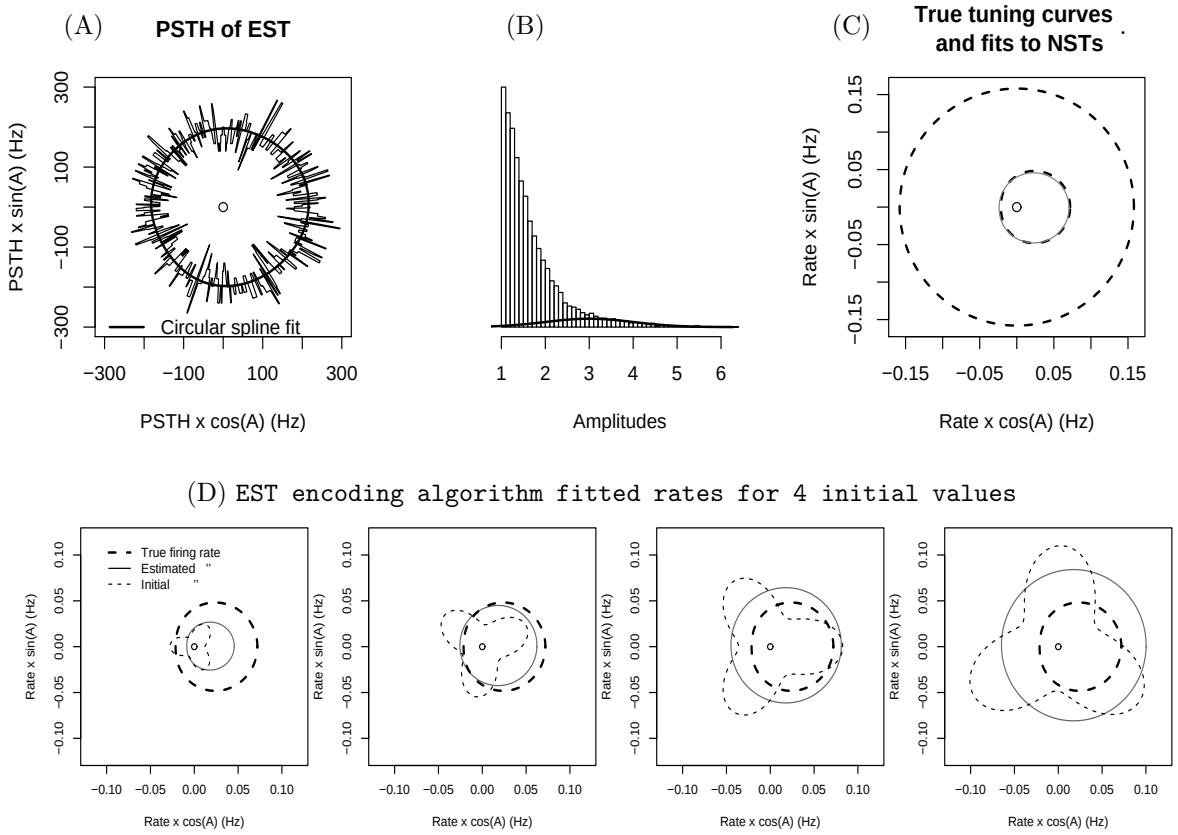


Figure 6: *(A) Circular PSTH of noise contaminated EST, with circular spline fit overlaid: it is difficult to see if the electrode records tuned neurons. (B) Electrode voltage amplitudes exceeding the threshold 1. The distribution of amplitudes of the neuron's waveform is overlaid. 78% of recorded spikes are noise, and it would be hard to spike sort noise from real spikes. (C) True rates for neuron and noise, with fits to the NST overlaid (not distinguishable). (D) Estimated neuron tuning curve obtained by applying the EST encoding algorithm to the noise contaminated EST. Fitted noise rates were not shown for clarity. The 4 panels correspond to the 4 clove leaf initial values overlaid. Fitted rates do not match the true rate, but are approximately proportional so they convey similar information about movement parameters.*

I	0	1	2	3	4	5
$\log L(\widehat{\Theta})$	-1661	-1568	-1442	-1412	-1410	-1409
$\log L(\widehat{\Theta}_{big}) - \log L(\widehat{\Theta}_{small})$	NA	93	126	30	2	1

Table 1: *Maximum log likelihood* $\log L(\widehat{\Theta})$ *achieved by fitting* I *neurons to the EST of the electrode used for Fig.4. The true number of neurons on this electrode is* $I = 3$.

difference in log likelihood is $\log L(\widehat{\Theta}_{big}) - \log L(\widehat{\Theta}_{small}) = (-1568) - (-1661) = 93$, which we compare to $\chi^2_{q,(1-\alpha)}/2 = 3$ for the LRT with significance level $\alpha = 5\%$, $q = 2$ for AIC, and $(q/2)\log n = 5.991$ for BIC with $n = 400$, the number of time bins we used to fit the models. The increase in log likelihood exceeds all critical values by a large amount, leaving no doubt that the electrode is recording at least one tuned neuron. To determine the number of neurons, the same procedure is applied albeit with $q = 3$ since the dimension of θ increases by three each time an additional neuron is included in the model. The corresponding AIC, BIC, and LRT critical values are 3, 8.99, and 3.91 respectively. The maximum log likelihood increases significantly up to $I = 3$ neurons, but the increase is not significant when a fourth neuron is added. We conclude that the electrode records $I = 3$ neurons, the correct number in this instance. For the electrode in Fig.5, $\log L(\widehat{\Theta})$ = -2137, -1464, -1411, -1402, -1402, and -1401 for $I = 0$ to $I = 5$ respectively, from which we conclude that the electrode records $I = 3$ tuned neurons, the correct number, despite substantial overlapping of the tuning curves.

3.1.3 Implementation issues

So far we have not discussed implementation issues because they are not central to the ideas in this paper. However, as with all numerical algorithms, it is important to consider them. The procedure we have adopted in this paper is as follows.

For clarity and consistency in Section 2, we developed the methodology based on binary spike trains. However, the theory extends to spike trains more coarsely binned, which is computationally more efficient. In the implementation of all results we used bins of 30 msecs. To fit I neurons, we took starting values to be I tuning curves with PDs as spread out as possible, and we declared that the algorithm had converged when the increase in log likelihood remained smaller than $\epsilon = 0.1$ for eight consecutive iterations. With these choices, and using an Intel(R) Pentium(R) 4 with 3.80GHz CPU and 4 Gigs of RAM, the EST encoding algorithm took an average of 60 or 20 seconds per electrode, depending on whether or not we fitted a noise neuron to the electrode, and almost half these times with use of $\epsilon = 0.2$. These timings are based on many simulations, and turned out to be independent of the total number of neurons and electrodes. Data driven initial values and better strategies to determine the number of neurons would accelerate the algorithm further. Decoding can also start before convergence, since estimated tuning curves are available at all times, and the algorithm can be left to run after convergence to track possible changes in tuning.

3.2 Decoding a trajectory

So far we have demonstrated that we can assess how many tuned neurons an electrode records, estimate their tuning curves, and separate noise from true spikes. However we also showed that the estimated

tuning curves are not necessarily those of the actual neurons, but can be those of imaginary neurons whose combined activity is not distinguishable from that of the actual neurons, and that separating noise from real spikes is more akin to decomposing the EST into tuned and untuned components. These effects are due to model unidentifiabilities. In this section we show that neither unidentifiabilities nor a high proportion of noise spikes have much effect on decoding accuracy. To save space, we report efficiency results for ML decoding only. PV decoding was overall less efficient but gave qualitatively similar results throughout.

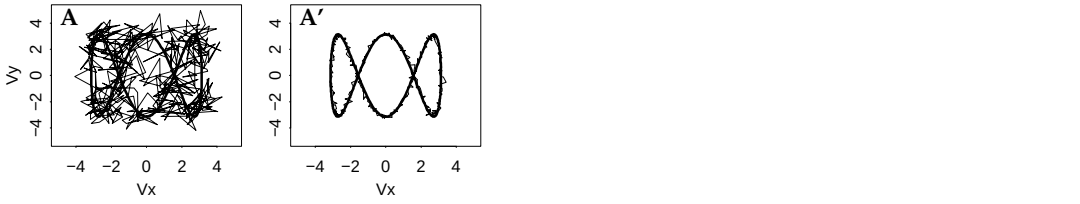
Fig.7 shows ML decoded trajectories based on simulated data sets of $N = 80$ cosine neurons randomly assigned to $J = 40$ electrodes. Panels with a prime letter such as Fig.7A' show the average prediction over 100 datasets. Non-primed panels show the prediction for a particular dataset. High and low SNR use the same datasets, but noise neurons spiking at 100 Hz were added to all electrodes in the low SNR case. Fig.7A shows the traditional NST prediction summarized in Fig.1. Exponential cosine tuning curves (Eq.34) were fitted to the NSTs and ML velocity predictions obtained from NSTs (Eq.26). High and low SNR cases are equivalent if we assume that we were able to spike sort the ESTs perfectly, so we left the right side of the plot empty. Fig.7B shows the naive prediction (Eq.27) based on treating electrodes as if they were neurons. Figs.7CD shows a hybrid between traditional and proposed decoding paradigms. Specifically, tuning curves were fitted to the NSTs, as in Fig.7A, but predictions were obtained from ESTs, as summarized in Fig.3. Fig.7C is for $\vec{P}_E$ and Fig.7D for the recursive prediction $\vec{P}_E^{recur}$. Finally, Figs.7EF shows the complete spike sorting free method. Comparing Figs.7EF to 7CD shows the effect of using imaginary rather than actual neurons.

We first verify from the prime panels that NST and EST predictions estimate the correct trajectory on average over data sets. The slight deviations are due to the difference between generative (Eq.33) and fitted models (Eq.34) and to the bias of $\vec{P}_E$ and $\vec{P}_E^{recur}$. The effect is more pronounced in the low SNR case, because then there are no single-neuron electrodes, so that expected spike counts must be calculated for all neurons. Focusing now on the non-primed panels, we see that the naive prediction $\vec{P}_U$ is poor and degrades in the high noise case, as expected from the analytical results in Section 2. On the other hand, $\vec{P}_E$ and $\vec{P}_E^{recur}$ compare quite well with the traditional NST prediction, and they are robust to contamination from noise. Finally, comparison of Figs.7CD and 7EF suggests that decoded trajectories are similar when tuning curves are estimated from NSTs or ESTs. That is, potential unidentifiability of tuning curves (or neurons) does not appear to affect the quality of decoding.

Fig.7 provides visual confirmation that our spike sorting free paradigm compares well with the traditional paradigm, including in the low SNR case when spike sorting would be difficult. To provide a more quantitative assessment, Fig.8 shows box plots of ISE ratios (RISEs) from 100 simulated datasets of $N = 80$ neurons randomly assigned to $J = 40$ electrodes. The successive panels of Fig.8 correspond to neurons that are increasingly more sharply tuned than cosine neurons, with tuning curve $\lambda_i^*(\vec{v}) = (k_i + m_i \vec{v} \cdot \vec{D}_i)^a$, $a = 0.75$, 1, 1.5 and 3 respectively (Eq.33). The same decoding model $\lambda_i(\vec{v}) = \exp(\theta_{0i} + \theta_{1i} v_x + \theta_{2i} v_y)$ (Eq.34) was always used. The number of neurons per electrode was determined using BIC and AIC. The latter was found to give somewhat better efficiencies, so we used AIC for all results shown here. In estimating the tuning curves via the EST encoding algorithm, we considered the addition or omission of noise neurons to be fitted to all electrodes. Hence efficiencies for $\vec{P}_E$ and $\vec{P}_E^{recur}$ are summarized by two boxplots (tagged E1–E2, F1–F2) corresponding to these options. Comparing boxes E1 to E2 and F1 to F2 suggests that fitting noise neurons to all electrodes improves decoding efficiency, even when spike trains contain no noise spikes (high SNR case). However the improvement is minimal when neurons have sharp tuning curves like $\lambda_i^*(\vec{v}) = (k_i + m_i \vec{v} \cdot \vec{D}_i)^3$. We discuss this further at the end of this section. Henceforth

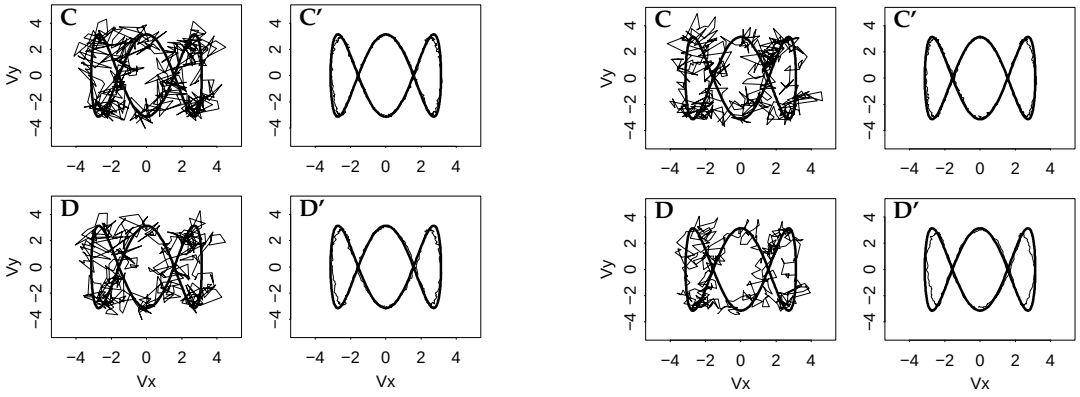
High SNR **Low SNR**

- **Traditional NST paradigm (Fig.1)**



- **Naive method**

- **Hybrid method (NST encoding, EST decoding)**



- **Proposed EST paradigm (Fig.2)**

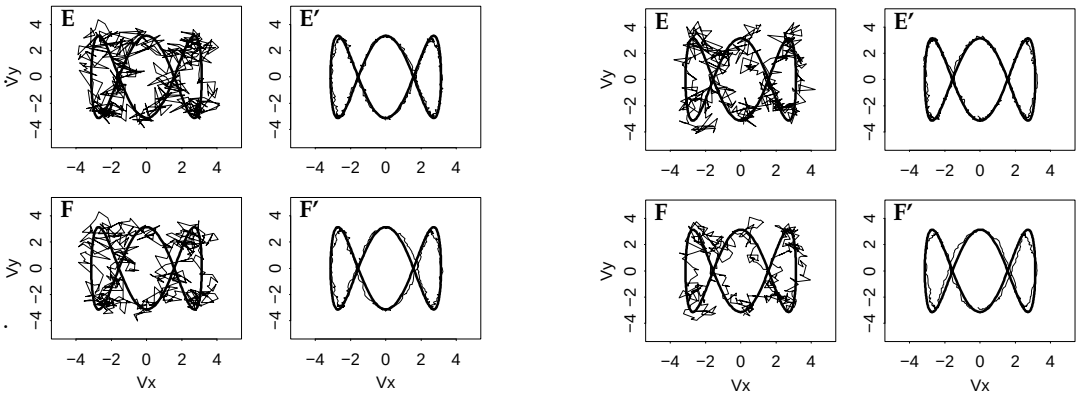
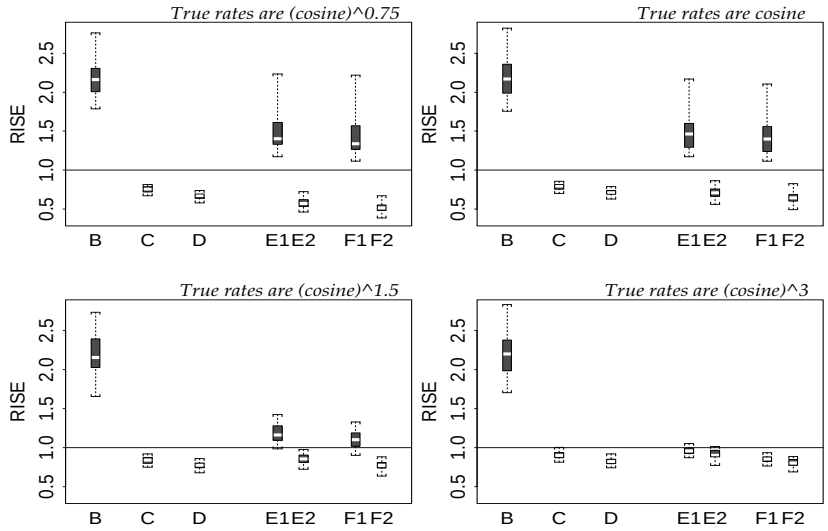
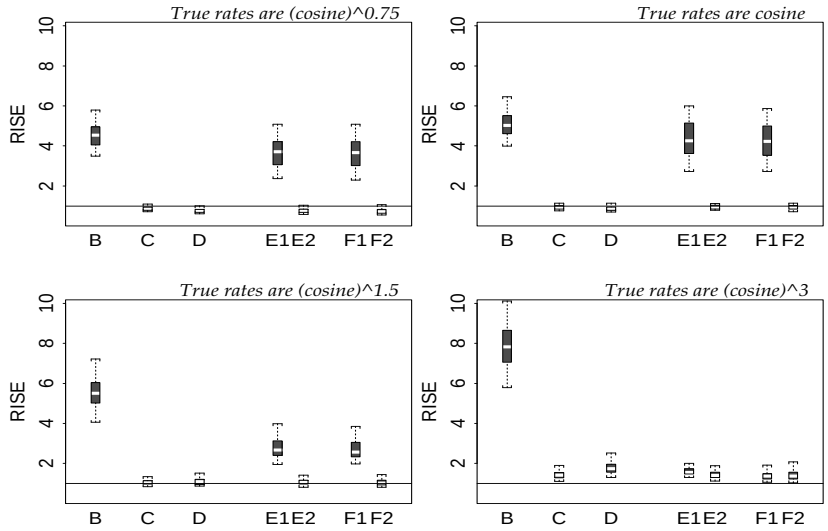


Figure 7: *Trajectories decoded by maximum likelihood using traditional (Fig.1) and proposed (Fig.2) methods. The true trajectory is the smooth bold line. The right hand side uses the same datasets as the left hand side, but noise neurons spiking at 100 Hz were added to each electrode. The primed panels show decoded trajectories averaged over 100 datasets, the non-primed panels the decoded trajectories for one particular dataset. Datasets consist of $N = 80$ exactly cosine neurons randomly assigned to $J = 40$ electrodes. (A) Traditional approach; firing rates and velocity predictions are obtained from NSTs. (B) Naive prediction; the traditional approach is applied to electrodes, as if they were neurons. (C,D) Hybrid approach; tuning curves are fitted to the NSTs as in (A), but velocity predictions calculated from ESTs. (C) is for $\vec{P}_E$ and (D) for the fully recursive $\vec{P}_E^{recur}$, as per Fig.3. (E,F) Proposed approach; tuning curves and predictions are obtained from ESTs. (E) is for $\vec{P}_E$ and (F) for $\vec{P}_E^{recur}$.*

High SNR



Low SNR



legend	prediction	Fit noise neurons to electrodes
B	$\vec{P}_U$	. . . . . N/A . . . . .
C	$\vec{P}_E$ (hybrid)	. . . . . N/A . . . . .
D	$\vec{P}_E^{recur}$ (hybrid)	. . . . . N/A . . . . .
E1	$\vec{P}_E$	no
E2	"	yes
F1	$\vec{P}_E^{recur}$	no
F2	"	yes

Figure 8: *Boxplots of ISE ratios (RISEs) for EST compared to NST velocity predictions. Each boxplot summarizes the distribution of 100 RISEs obtained from 100 datasets of $N = 80$ neurons randomly assigned onto $J = 40$ electrodes. True rates are (cosine)^a with $a = 0.75, 1, 1.5$ and 3. Fitted rates are exp(cosine). In the low SNR case, noise neurons spiking at 100 Hz were added to all electrodes. We use the same nomenclature as Fig.7. (Boxes B) RISE of naive prediction $\vec{P}_U$ over traditional NST prediction $\vec{P}$. (Boxes C, D) RISE of hybrid EST predictions $\vec{P}_E$ and $\vec{P}_E^{recur}$ over NST prediction $\vec{P}$. The hybrid method uses ESTs for decoding but NSTs for encoding; hence boxplots A, B, and C all use the same tuning curves, which correspond to actual neurons. (Boxes E1-2, F1-2) RISE of EST predictions $\vec{P}_E$ and $\vec{P}_E^{recur}$ over NST prediction $\vec{P}$, with noise neurons omitted or included in the EST encoding algorithm.*

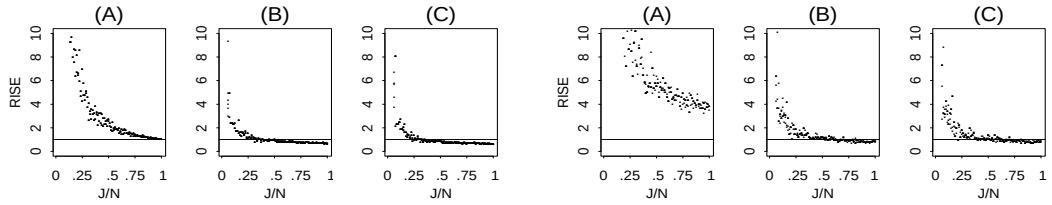


Figure 9: *RISEs of naive prediction $\vec{P}_U$ (A) and EST predictions $\vec{P}_E$ (B) and $\vec{P}_E^{recur}$ (C), as functions of the electrode to neuron ratio J/N, for 10 datasets of N neurons with true tuning curves (cosine)$^{1.5}$, randomly assigned to J electrodes. We used $N = 80$ and 40 and let J take values from 5 to N.*

we refer to boxplots E2 and F2 when discussing the efficiencies of $\vec{P}_E$ and $\vec{P}_E^{recur}$, which corresponds to fitting noise neurons to all electrodes and determining numbers of neurons via AIC in the EST encoding algorithm.

Efficiencies also depend on the bias parameters k and k^{recur} in the calculation of $\vec{P}_E$ and $\vec{P}_E^{recur}$. We reproduced Fig.8 for varying values of k, k^{recur}, N and J (not shown), from which we determined that optimal choices were $k \doteq 8 - 10$ and $k^{recur} = 1$. Figs.7 and 8 use these values. It makes sense that the optimal k should be larger than the optimal k^{recur} since $\vec{P}_U$, on which $\vec{P}_E$ is based, is more variable than $\vec{P}_E^{recur}$.

Now that good parameters have been established for our algorithm, we compare the various prediction methods. The RISEs of naive predictions $\vec{P}_U$ (boxes A) are well above one, which confirms the earlier analytical result that failing to sort the spike trains degrades decoding efficiency. The proposed EST predictions fare much better in the high SNR case, with average RISEs below one (boxes C, D, E2 and F2), which suggests that efficiency can in fact be gained from avoiding spike sorting. This suggestion does not extend to the low SNR case, since average RISEs are around one. However this result is unfair to our method, since we ignored the noise altogether when decoding from NSTs, when it would likely be very difficult to spike sort. Finally, comparing C to E2 and D to F2 suggests that unidentifiabilities of neurons has no effect on efficiency, as we had already observed in Fig.7.

Our results so far have relied on datasets of $N = 80$ neurons randomly assigned to $J = 40$ electrodes. As discussed earlier, the efficiencies of the proposed predictions depend on the proportion of single-neuron electrodes. Fig.9 shows the RISEs of decoded trajectories for 10 datasets of N neurons with (cosine)$^{1.5}$ tuning, randomly assigned to J electrodes. We used $N = 80$ and $N = 40$ and let J take values from 5 to N. Other tuning curves and values of N produced similar results. As expected from earlier analytical results, $\vec{P}_U$ is always inferior to $\vec{P}$, except when $N = J$ in the high SNR situation, in which case the two are equivalent. On the other hand, $\vec{P}_E$ and $\vec{P}_E^{recur}$ are superior to $\vec{P}$ approximately when $J/N > 25\%$, which corresponds approximately to 10% or more single-neuron electrodes. This is likely to be achieve in practice. These conclusions also apply when the ESTs are heavily contaminated by noise, although the efficiency gain is lesser. But as mentioned earlier, spike sorting would be hard in the low SNR case, so the RISEs reported in Fig.9 and other figures unfairly penalize the proposed approach.

We wrap up this section with thoughts on model building. Even though our analytical calculations suggested that decoding efficiency could be gained from our paradigm, we were still surprised by the good results in Fig.8. To investigate this further, consider a selected electrode from the simulation study.

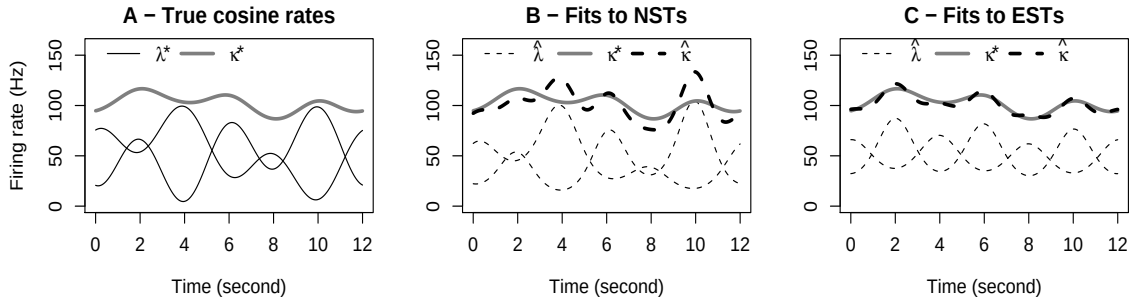


Figure 10: *True and estimated firing rates of neurons recorded by an electrode in the simulation study. (A) True cosine rates λ_i^* and rate induced at the electrode $\kappa^* = 1 - \prod(1 - \lambda_i^*)$, plotted as functions of the 12 second velocity trajectory shown in Fig.7. (B) Firing rates $\hat{\lambda}_i$ fitted to the NSTs, the corresponding rate at the electrode, $\hat{\kappa} = 1 - \prod(1 - \hat{\lambda}_i)$, and the true electrode rate, κ^*. (C) Fitted rates obtained by the EST encoding algorithm, the corresponding fitted electrode rate, and the true electrode rate. The proposed method provides a better fit for the electrode firing rate, given the same model for neurons firing rates.*

It records two exactly cosine neurons, with true firing rates λ_i^*, shown in Fig.10A as functions of time along the 12 second trajectory path. The firing rate $\kappa^* = 1 - \prod(1 - \lambda_i^*)$ induced at the electrode is overlaid in bold. Although neurons are cosine, we fitted exponential cosine tuning curves throughout the simulation. Fig.10C shows the fits obtained by the EST encoding algorithm with noise neurons included, the implied $\hat{\kappa}$, and the true κ^*. The agreement between $\hat{\kappa}$ and κ^* is good. Fig.10B shows the corresponding rates fitted to the NSTs and the implied electrode rate. This time the disagreement between $\hat{\kappa}$ and κ^* is substantial. Fig.10B illustrates the lack of fit that can be expected from fitting an incorrect model. Lack of fit typically causes loss of decoding efficiency. What is remarkable is that the EST algorithm is able to alleviate discrepancies between the unknown generative model and the model we choose to fit to the neurons, and thus improve decoding. This happens in part because the noise neurons we fit to all electrodes make for more flexible models. This is important in real application, since we never use the "correct" model. In the fourth panels of Fig.8, the exponential cosine model provided a better fit to neurons with sharper tuning curves $\lambda^*(\vec{v}) = (k_i + m_i \vec{v} \cdot \vec{D}_i)^3$, so that the traditional NST approach was (almost) fully efficient. The better match between generative and decoding models also meant that noise neurons were not crucially needed to provide flexibility in the EST encoding algorithm. In that case the average RISEs were around one, meaning that traditional and proposed decoding paradigms were equally efficient on average.

4 Discussion

We have proposed a novel paradigm for spike train decoding, which avoids entirely spike sorting based on waveform information. Our approach is a paradigm, not an algorithm, since it can be used with any of the current decoding algorithms. In this paper we focused on population vector and maximum likelihood decoding. A forthcoming paper deals with dynamic Bayesian decoding.

Based on extensive simulations we showed that, provided that at least 10% of the electrodes are tuned to

relevant movement variables, our paradigm is at least as efficient as traditional decoding based on well-isolated neurons. Our approach is particularly attractive for two reasons. First, in place of the lengthy spike sorting task of the traditional approach, it involves an exact expectation EM that is fast enough that it could also run during decoding to capture potential slow changes in the states of the neurons. This is particularly relevant for neural prostheses, for which speed and computing power are limiting bottlenecks. Second, our paradigm remains efficient even when all electrodes are severely corrupted by noise, a situation that is common with chronically implanted electrodes, and that renders traditional spike sorting particularly difficult. In addition, our approach appears to alleviate some model mis-specifications, which is of interest because the statistical models we use are only approximations to the true models that generate the data.

While it is true that our paradigm avoids spike sorting based on waveforms, it does implicitly extract neurons' identities based on information about neuron tuning, as was revealed in Fig.4D. When used in conjunction with spike waveforms information, the EST encoding algorithm improves spike sorting and encoding simultaneously. These methods and results are presented in a forthcoming paper, with applications more general than decoding. We did not consider waveforms in this paper because our objective was to propose a full encoding/decoding paradigm that does not require them. All that is required are the spike trains collected at recording electrodes from thresholding the bandpassed voltage signal. Moreover, including waveform information to our current paradigm will not improve decoding, since the efficiencies reported in Figs.7 and 8 are similar either if we use actual neurons or not.

Our encoding algorithm includes assumptions that would be hard to dispute. We assume that movements are smooth, so that current values of movement variables contain information about the same variables in the immediate future, and that motor neurons are broadly tuned to these variables, relying on a large body of work starting with Georgopoulos *et al.* (1982). We avoid issues of model selection altogether because they are not specific to our approach. The process of deciding how neurons encode for movement variables would be similar with the current and proposed paradigms, and future research providing better tuning curve models can be incorporated in our approach, as with the traditional approach. To keep the development of ideas simple, we further assume that neurons are independent and Poisson, although this does not affect the quality of our results. The last assumption could be dropped by using a firing rate model that depends on the past.One option is the inhomogeneous Markov model of Kass and Ventura (2001), which can accommodate effects such as refractory periods. The independence assumption could also be dropped by building dependencies between neurons in the firing rates, as suggested for example by Martignon *et al.* (2000), Okatan *et al.* (2005), and Kulkarni and Paninski (2007). Finally, we assume that a neuron could be recorded by only one electrode. To handle electrode arrays, for which the same neuron can be recorded by several electrodes, an option would be to include in the model a binary include/exclude variable for each neuron on all electrodes, and let these be estimated by the EM algorithm.

Acknowledgments

The author was supported by NIH grants 2RO1MH064537 and 1R01EB005847.

References

Barbieri, R, Frank, LM, Nguyen, DP, Quirk, MC, Solo, V, Wilson, MA, and Brown, EN. (2004). Dynamic analyses of information encoding by neural ensembles. *Neural Computation*, 16(2): 277–307.

Brockwell, AE, Rojas, A, and Kass, RE. (2004). Recursive Bayesian decoding of motor cortical signals by particle filtering. *Journal of Neurophysiology*, 91:1899–1907.

Brockwell, AE, Kass, RE, and Schwartz, AB. (2004). Statistical Signal Processing and the Motor Cortex. *to appear*.

Brown, EN, Kass, RE, and Mitra, PP. (2004). Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nature Neuroscience*, 7:456-461.

Brown, EN, Frank, LM, Tang, D, Quirk, MC, Wilson, MA (1998). A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *Journal of Neuroscience*, 18:7411-7425.

Carmena, JM, Lebedev, RE, Crist, JE, O'Doherty, JE, Santucci, DM, Dimitrov, DF, Patil, PG, Henriquez, CS, and Nicolelis, MAL. (2003). Learning to control a brain-machine interface for reaching and grasping by primates. *PLoS Biol,* 1:193–208.

Dempster, AP, Laird, NM, and Rubin, DB. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *J. of the Royal Statistical Society B*, 39(1):138.

Fee, MS, Mitra, PP, and Kleinfeld, D. (1996). Variability of extracellular spike waveforms of cortical neurons. *J. Neurophysiol*, 76:3823–3833.

Georgopoulos, AP, Kalaska, JF, Caminiti, R, and Massey, JT. (1982). On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *J. Neurosci.*, 2:1527–1537.

Georgopoulos, AP, Kettner, RE, and Schwartz, AB. (1988). Primate motor cortex and free arm movements to visual targets in three-dimensional space. II. coding of the direction of movement by a neuronal population. *Neuroscience*, 8:2928–2937.

Georgopoulos, AP, Schwartz, AB, and Kettner, RE. (1986). Neuronal population coding of movement direction. *Science*, 233:1416–1419.

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

Hochberg, LR, Serruya, MD, Friehs, GH, Mukand, JA, Saleh, M, Caplan, AH, Branner, A, Chen, D, Penn, RD, and Donoghue, JP. (2006). Neuronal ensemble control of prosthetic deices by a human with tetraplegia. *Nature*, 442:164–171.

Kass, RE and Ventura, V. (2001). A Spike-train probability model. *Neural Comput.* 13: 1713–1720.

Kass, RE, Ventura, V, and Brown, EN. (2005). Statistical issues in the analysis of neuronal data. *J. Neurophysiology*, 94:8–25.

Kemere, C, Shenoy, KV, and Meng, TH. (2003). Model-based neural decoding of reaching movements: A maximum likelihood approach. *IEEE Trans. on Biomedical Engineering*, 51:925–932.

Kulkarni, JE and Paninski, L. (2007). Common-input models for multiple neural spike-train data. *Network: computation in neural systems*, to appear.

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

Martignon, L, Deco, G, Laskey, K, Diamond, M, Freiwald, W, and Vaadia, E. (2000) Neural Coding: Higher-Order Temporal Patterns in the Neurostatistics of Cell Assemblies. *Neural Comput.* 12: 2621-265.

McCullagh, P and Nelder, JA. (1999). *Generalized Linear Models*. Chapman and Hall, second edition.

Moran, DW and Schwartz, AB. (1999). Motor Cortical Representation of Speed and Direction During Reaching. *J Neurophysiol*, 82: 2676-2692.

Musallam, S, Corneil, BD, Greger, B, Scherberger, H, and Andersen, RA. (2004). Cognitive control signals for neural prosthetics. *Science*, 305:258–262.

Neal, RM, and Hinton, GE. (1999). A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants. *Learning in Graphical Models*. Michael Jordan, editor. Kluwer Academic.

Okatan, M, Wilson, MA, and Brown, EN. (2005). Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. *Neural Computation*, 17:1927–1961.

Pouzat, C, Delescluse, M, Voit, P, and 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.

Salinas, E, and Abbott, LF. (1994). Vector reconstruction from firing rates. *Journal of Computational Neuroscience*, 1:89–107.

Sanger, TD. (1996). Probability density estimation for the interpretation of neural population codes. *J. Neurophysiology*, 76(4):2790–2793.

Santhanam, G, Ryu, SI, Yu, BM, Afshar, A, and Shenoy, KV. (2006). A high-performance brain-computer interface. *Nature*, 442:195–198.

Schwartz, AB. (2004). Cortical neural prosthetics. *Annual Review of Neuroscience*, 27:487–507.

Shoham, S, Fellows, MP, and Normann, RA. (2003). Robust, automatic spike sorting using mixtures of multivariate distributions. *J. Neurosci. Methods*, 127:111-122.

Shoham, S, Paninski, LM, Fellows, MR, Hatsopoulos, NG, Donoghue, JP, and Normann, RA. (2005). Statistical encoding model for a primary motor cortical brain-machine interface. *IEEE Trans. on Biomedical Engineering*, 52(7):1312–1322.

Taylor, DM, Helms Tillery, SI, and Schwartz, AB. (2002). Direct cortical control of 3D neuroprosthetic devices. *Science*, 296:1829–1832.

Truccolo, W, Eden, UT, Fellos, MR, Donoghue, JP, and Brown, EN. (2005). A point process framework for relating neural spiking activity to spiking history, neural ensemble and extrinsic covariate effects. *J. Neurophysiology*, 91:1074–1089.

Wessberg, J, Stambaugh, CR, Kralik, JD, Beck, PD, Lauback, M, Chaplin, JK, Kim, J, Biggs, SJ, Srinivasan, MA, and Nicolelis, MAL. (2000). Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. *Nature*, 408(6810):361–365.

Wu, W., Shaikhouni, A, Donoghue, JP, and Black, MJ. (2004). Closed-loop neural control of cursor motion using a kalman filter. *Proc. IEEE Engineering in Medicine and Biology Society*, 4126–4129.

Zhang, K, Ginzburg, I, McNaughton, BL, and Sejnowski, TJ. (1998). Interpretating neuronal population activity by reconstruction: Unified framework with applications to hippocampal place cells. *Journal of Neurophysiology*, 79:1017–1044.

Appendices

4.1 Efficiencies of proposed and traditional PV predictions

$$
\begin{aligned}
E(\vec{P}_E) &= E\left(\sum_{i=1}^{N} e_i\ \vec{D}_i\right) = \sum_{i=1}^{N} E(e_i)\ \vec{D}_i && E \text{ is a linear operator} \\
&= \sum_{i=1}^{N} E(z_{j_i}) \frac{\lambda_i(\vec{v})}{\kappa_{j_i}(\vec{v})}\ \vec{D}_i && \text{by definition of } e_i \text{ in Eq.25} \\
&= \sum_{i=1}^{N} \left[\sum_{j\in I_{j_i}} E(y_j)\right] \frac{\lambda_i(\vec{v})}{\kappa_{j_i}(\vec{v})}\ \vec{D}_i && \text{by definition of } z_{j_i} \text{ in Eq.32} \\
&= \sum_{i=1}^{N} \left[\sum_{j\in I_{j_i}} \lambda_j(\vec{v})\right] \frac{\lambda_i(\vec{v})}{\kappa_{j_i}(\vec{v})}\ \vec{D}_i && y_j \text{ has mean } \lambda_j(\vec{v}) \text{ by definition} \\
&= \sum_{i=1}^{N} \lambda_i(\vec{v})\ \vec{D}_i && \text{by definition of } \kappa_{j_i}(\vec{v}) \text{ in Eq.31.}
\end{aligned}
$$

Similarly, $E(\vec{P}) = E\left(\sum_{i=1}^{N}\ y_i\ \vec{D}_i\right) = \sum_{i=1}^{N}\ E(y_i)\ \vec{D}_i = \sum_{i=1}^{N}\ \lambda_i(\vec{v})\ \vec{D}_i$.

Predictions $\vec{P}$ and $\vec{P}_E$ are two-dimensional, so we calculate their variance covariance matrices to assess their variabilities. Considering the x-coordinate of $\vec{P}$ we have

$$
\begin{aligned}
\mathrm{Var}(P_x) &= \mathrm{Var}\left(\sum_{i=1}^{N} y_i\ D_{ix}\right) \\
&= \sum_{i=1}^{N} \mathrm{Var}(y_i\ D_{ix}) \qquad (\text{since } y_i \perp y_j) \\
&= \sum_{i=1}^{N} \mathrm{Var}(y_i)\ D_{ix}^2 = \sum_{i=1}^{N} \sigma_i^2\ D_{ix}^2,
\end{aligned}
$$

where σ_i^2 is the variance of spike count y_i. If we assumed y_i to be Poisson, then we would have $\sigma_i^2 = \lambda_i(\vec{v})$. Similarly for the y-coordinate we have

$$
\mathrm{Var}(P_y) = \sum_{i=1}^{N} \sigma_i^2\ D_{iy}^2.
$$

Finally the covariance between P_x and P_y is

$$\begin{aligned}
\mathrm{Cov}(P_x, P_y) &= \mathrm{Cov}\left(\sum_{i=1}^{N} y_i\, D_{ix}, \sum_{j=1}^{N} y_j\, D_{jy}\right) \\
&= \sum_{i=1}^{N} \mathrm{Cov}(y_i\, D_{ix}, (y_i\, D_{iy}) \qquad (\text{since } y_i \perp y_j) \\
&= \sum_{i=1}^{N} \sigma_i^2\, D_{ix} D_{iy}.
\end{aligned}$$

Note that if the PDs are uniformly distributed and if σ_i is approximately the same for all neurons, $\mathrm{Cov}(P_x, P_y) \doteq 0$.

To deal with Var $(\vec{P}_E)$, we first rewrite $\vec{P}_E$ as

$$\vec{P}_E = \sum_{i=1}^{N} e_i\, \vec{D}_i = \sum_{i=1}^{N} z_{j_i}\, \frac{\lambda_i(\vec{v})}{\kappa_{j_i}(\vec{v})}\, \vec{D}_i = \sum_{j=1}^{J} z_j \sum_{i \in I_j} \frac{\lambda_i(\vec{v})}{\kappa_{j_i}(\vec{v})}\, \vec{D}_i,$$

by switching the summation over the neurons by the summation over the electrodes. Because the z_i's are independent (the e_i's were not), Var $(\vec{P}_E)$ is now more easily tractable. Specifically, for the x-coordinate

$$\begin{aligned}
\mathrm{Var}(P_{Ex}) &= \sum_{j=1}^{J} \mathrm{Var}(z_j) \left(\sum_{i \in I_j} \frac{\lambda_i}{\kappa_{j_i}}\, D_{ix}\right)^2 \\
&= \sum_{j=1}^{J} \left[\sum_{i \in I_j} \mathrm{Var}(y_i)\right] \left(\sum_{i \in I_j} \frac{\lambda_i}{\kappa_{j_i}}\, D_{ix}\right)^2 \\
&= \sum_{j=1}^{J} \left[\sum_{i \in I_j} \sigma_i^2\right] \left(\sum_{i \in I_j} \frac{\lambda_i}{\kappa_{j_i}}\, D_{ix}\right)^2,
\end{aligned}$$

and similarly

$$\mathrm{Var}(P_{Ey}) = \sum_{j=1}^{J} \left[\sum_{i \in I_j} \sigma_i^2\right] \left(\sum_{i \in I_j} \frac{\lambda_i}{\kappa_{j_i}}\, D_{iy}\right)^2,$$

while

$$\mathrm{Cov}(P_{Ex}, P_{Ey}) = \sum_{j=1}^{J} \left[\sum_{i \in I_j} \sigma_i^2\right] \left(\sum_{i \in I_j} \frac{\lambda_i}{\kappa_{j_i}}\, D_{ix}\right) \left(\sum_{i \in I_j} \frac{\lambda_i}{\kappa_{j_i}}\, D_{iy}\right).$$

The variance-covariance matrices of $\vec{P}$ and $\vec{P}_E$ cannot be compared without imposing any constraint on $\sigma_i^2 = \mathrm{Var}(y_i)$. It is not too restrictive to assume that Var $(y_i) = \alpha\, \lambda_i(\vec{v}), \alpha \in R^+$, which is to say that the variance of spike counts is proportional to their means. This happens, for example, when spike counts are over or under dispersed Poisson random variables; the case $\alpha = 1$ corresponds to Poisson spike trains. Under this assumption,

$$\mathrm{Var}(P_{Ex}) = \alpha \sum_{j=1}^{J} \left[\frac{\left(\sum_{i \in I_j} \lambda_i\, D_{ix}\right)^2}{\sum_{i \in I_j} \lambda_i}\right]$$

and

$$\mathrm{Var}(P_x) = \alpha \sum_{i=1}^{N} \lambda_i \, D_{ix}^2 = \alpha \sum_{j=1}^{J} \left(\sum_{i \in I_j} \lambda_i \, D_{ix}^2 \right).$$

Both $\mathrm{Var}(P_x)$ and $\mathrm{Var}(P_{Ex})$ are summations of positive quantities over the electrodes. Applying Cauchy-Schwartz inequality yields

$$\left(\sum_{i \in I_j} \lambda_i D_{ix} \right)^2 \leq \left(\sum_{i \in I_j} \lambda_i \right) \left(\sum_{i \in I_j} \lambda_i D_{ix}^2 \right),$$

for $j = 1, \ldots, J$, which in turn implies

$$\text{Var-Cov}(\vec{P}_E) \leq \text{Var-Cov}(\vec{P}),$$

where the inequality applies to all elements of the matrices.

4.2 Decoding efficiency degrades if we use electrodes in place of well isolated neurons

We offer a movement prediction perspective to explain why decoding from isolated neurons might be better than decoding from electrodes. Assume that an electrode records the two neurons with firing rates in Fig.11A; Fig.11B shows the rate $\kappa(\vec{v})$ induced by these neurons at the electrode. We plotted the rates in spike counts per 30 msec bins, and assumed that $\vec{v}$ had constant magnitude, with direction varying in $[0, 2\pi]$. Spike counts vary about their expectations, which is represented by dotted lines $2\sigma_i$ on each side of $\lambda_i(\vec{v})$. Without loss of generality we used $\sigma_i = \sqrt{\lambda_i(\vec{v})}$, consistent with Poisson counts. Assume now that $z = 12$ spikes are detected at the electrode in a time bin, depicted by the horizontal line in Fig.11B. The relationship between spike counts and tuning curves implies that velocities that could have produced such a count are approximately between 0.44 and 5.17 radians. On the other hand, if we know that neuron 1 spiked $y_1 = 10$ times and neuron 2 $y_2 = 2$ times, we obtain from Fig.11A that the velocity is between 0.37 and 3.4 for neuron 1, and between 0 and 2.46 or between 5.02 and 2π based on neuron 2. Combining the two sources of information places the velocity between 2.46 and 3.4 approximately, a more accurate prediction than that based on the electrode.

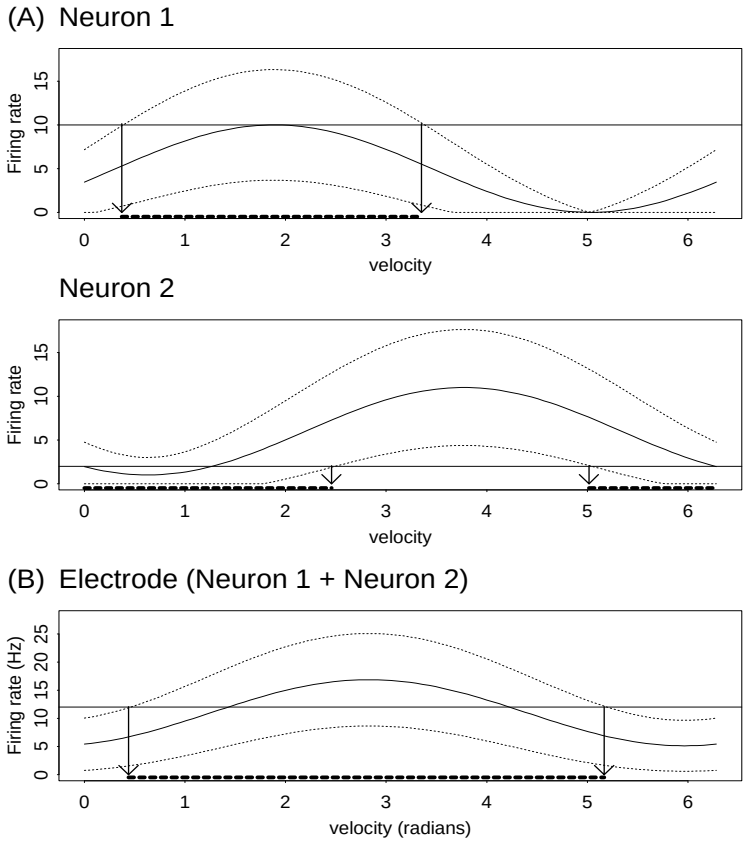


Figure 11: *(A) Tuning curves $\lambda_i(\vec{v})$ of two neurons. (B) Tuning curve of the electrode $\kappa(\vec{v}) = 1 - (1 - \lambda_1(\vec{v}))(1 - \lambda_2(\vec{v}))$. Spike counts vary about their expectations, represented by dotted lines $2\sigma_i = 2\sqrt{\lambda_i(\vec{v})}$ on each side of $\lambda_i(\vec{v})$. (B) Assume that at time t, $z = 12$ spikes are recorded from the electrode, depicted by the horizontal line in (B); the relationship between spike counts and tuning curves implies that the velocity is between 0.44 and 5.17 radians, the interval marked by a thick line at the a-axis in (B). If we knew that neuron 1 spiked $y_1 = 10$ times and neuron 2 spiked $y_2 = 2$ times, we would infer that the velocity was between 0.37 and 3.4 based on neuron 1, and between 0 and 2.46 or between 5.02 and 2π based on neuron 2; these intervals are marked by thick lines in the x-axes in (A). Combining the two sources of information would place the velocity at the intersection of these intervals, which is between 2.46 and 3.4 radians, a more accurate prediction than that based on the electrode spike count in (B).*