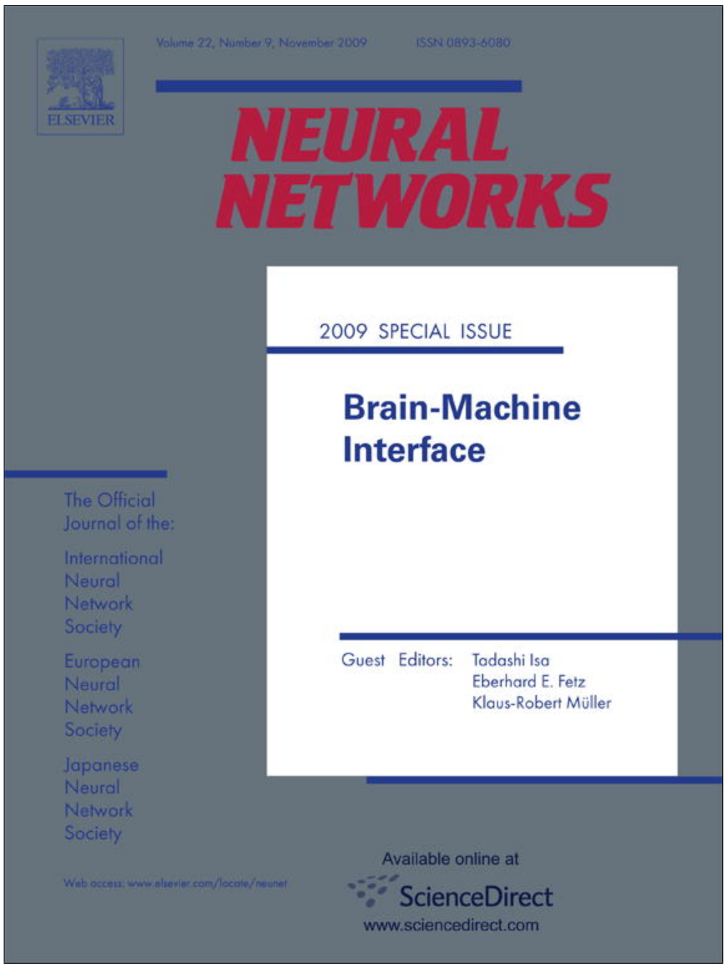

Volume 22, Number 9, November 2009
ISSN 0893-6080
ELSEVIER
NEURAL NETWORKS
2009 SPECIAL ISSUE
Brain-Machine Interface
Guest Editors: Tadashi Isa
Eberhard E. Fetz
Klaus-Robert Müller
The Official Journal of the:
International Neural Network Society
European Neural Network Society
Japanese Neural Network Society
Web access: www.elsevier.com/locate/neunet
Available online at
ScienceDirect
www.sciencedirect.com

Neural Networks 22 (2009) 1235–1246

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2009 Special Issue

Methods for estimating neural firing rates, and their application to brain–machine interfaces

John P. Cunningham[a], Vikash Gilja[b], Stephen I. Ryu[c,1], Krishna V. Shenoy[a,d,*]

[a] *Department of Electrical Engineering, Stanford University, Stanford, CA, USA*
[b] *Department of Computer Science, Stanford University, Stanford, CA, USA*
[c] *Department of Neurosurgery, Stanford University, Stanford, CA, USA*
[d] *Neurosciences Program, Stanford University, Stanford, CA, USA*

ARTICLE INFO

Article history:
Received 20 October 2008
Received in revised form 20 February 2009
Accepted 24 February 2009

Keywords:
Neural firing rate
Brain–machine interface
Neural prosthetic system

ABSTRACT

Neural spike trains present analytical challenges due to their noisy, spiking nature. Many studies of neuroscientific and neural prosthetic importance rely on a smoothed, denoised estimate of a spike train's underlying firing rate. Numerous methods for estimating neural firing rates have been developed in recent years, but to date no systematic comparison has been made between them. In this study, we review both classic and current firing rate estimation techniques. We compare the advantages and drawbacks of these methods. Then, in an effort to understand their relevance to the field of neural prostheses, we also apply these estimators to experimentally gathered neural data from a prosthetic arm-reaching paradigm. Using these estimates of firing rate, we apply standard prosthetic decoding algorithms to compare the performance of the different firing rate estimators, and, perhaps surprisingly, we find minimal differences. This study serves as a review of available spike train smoothers and a first quantitative comparison of their performance for brain–machine interfaces.

1. Introduction

Neuronal activity is highly variable. Even when experimental conditions are repeated closely, the same neuron may produce quite different spike trains from trial to trial. This variability may be due to both randomness in the spiking process and to differences in cognitive processing on different experimental trials. One common view is that a spike train is generated from a smooth underlying function of time (the firing rate) and that this function carries a significant portion of the neural information (vs. the precise timing of individual spikes). If this is the case, questions of neuroscientific and neural prosthetic importance may require an accurate estimate of the firing rate. Unfortunately, these estimates are complicated by the fact that spike data gives only a sparse observation of its underlying rate. Typically, researchers average across many trials to find a smooth estimate (averaging out spiking noise). However, averaging across many roughly similar trials can obscure important temporal features (Nawrot, Aertsen, & Rotter, 1999; Yu et al., 2005, 2009). Trial averaging can be especially problematic in a brain–machine interface (BMI) setting, where physical behavior is not under strict experimental control, and so motor movements and their associated neural activity can vary considerably across trials. Thus, estimating the underlying rate from only one spike train is an important but challenging problem.

To address this problem, researchers have developed a number of methods for estimating continuous, time-varying firing rates from neural spike trains. The goal of any firing rate estimator is twofold: first, the method seeks to return a smooth, continuous-time firing rate that is more amenable to analytical efforts than the spiking neural signal. Second, as is the goal of any statistical signal processing algorithm, the firing rate estimator seeks to denoise the signal (separate the meaningful fluctuations in underlying firing rate from the noise introduced by the spiking process). This firing rate estimation step is shown in Fig. 1. Panel (a) shows a single spike train (one experimental trial) for each of N neural units. The spike train is shown as a train of black rasters, where each raster (vertical tick) represents the occurrence of a spike at that time in the trial. The firing rate estimator seeks to process each of these noisy spike trains into smooth, continuous-time firing rates that are denoised and simpler to analyze, as shown in panel (b). Finally, in a BMI setting (our case of interest here), these firing rates may then be used by a prosthetic decoding algorithm to estimate a motor movement, as shown in Fig. 1, panel (c).

* Corresponding address: Department of Electrical Engineering, Stanford University, 330 Serra Mall, CISX 319, Stanford, CA 94305-4075, USA.
E-mail address: shenoy@stanford.edu (K.V. Shenoy).

1 Present address: Department of Neurosurgery, Palo Alto Medical Foundation, Palo Alto, CA, USA.

0893-6080/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2009.02.004

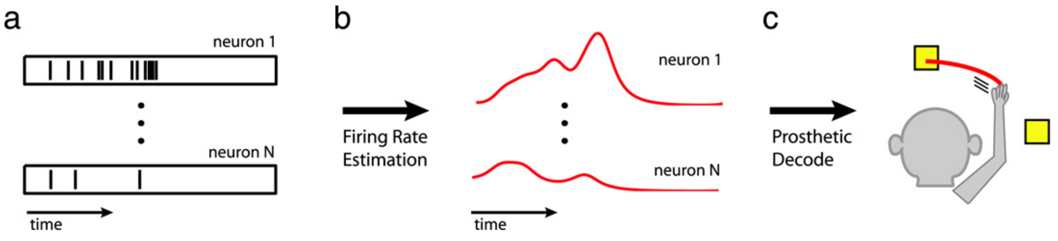


Fig. 1. Context for firing rate estimation and neural prosthetic decode. (a) *N* single spike trains are gathered from *N* neurons on one experimental trial. (b) Those spike trains are denoised and smoothed using a firing rate estimation method. (c) Those firing rates are used by a decoding algorithm to estimate, for example, a reaching arm trajectory.

In this study, we review the methods that have been developed both classically and more recently, from the fields of statistics, machine learning, and computational neuroscience (see Section 2). We point to the relevant publications and give high level overviews of each method, noting a few potential strengths and weaknesses with respect to the problem of estimating firing rates from single spike trains.

Having reviewed several estimation methods, we then turn to the question of performance. To date, no comparison between these methods exists; such comparisons may assist researchers in determining what firing rate estimator is appropriate for what application. In this study, we choose the BMI application of neural prosthetic decoding in an arm-reaching setting. We train a monkey to make point-to-point reaches in a 2D workspace. Using a multi-electrode array implanted in pre-motor/motor cortex, we record spike trains from 10–15 neural units (we consider only high quality single units) during this reaching task. There are many prosthetic decoding algorithms that can decode the arm movement from the recorded neural activity (some papers include: Brockwell, Rojas, and Kass (2004), Brown, Frank, Tang, Quirk, and Wilson (1998), Carmena et al. (2003), Carmena, Lebedev, Henriquez, and Nicolelis (2005), Chestek et al. (2007), Georgopoulos, Schwartz, and Kettner (1986), Hochberg et al. (2006), Kemere, Shenoy, and Meng (2004), Serruya, Hatsopoulos, Paninski, Fellows, and Donoghue (2002), Srinivasan, Eden, Mitter, and Brown (2007), Taylor, Tillery, and Schwartz (2002), Wu et al. (2004), Wu, Gao, Bienenstock, Donoghue, and Black (2006), Velliste, Perel, Spalding, Whitford, and Schwartz (2008) and Yu et al. (2007)). Some of these algorithms use smooth estimates of firing rates as input. Here we investigate how the performance of these decoders changes, depending on what firing rate estimation method is used. In particular, we choose the widespread linear decoder (as recently used in Carmena et al. (2005) and Chestek et al. (2007)) and the Kalman filter (as recently used in Wu et al. (2002, 2004, 2006)). We individually smooth thousands of spike trains (from many trials and many neural units) with each firing rate estimation method, and we decode arm trajectories from these firing rate estimates with the same decoding algorithms.

The purpose of this paper then is both to review available firing rate estimators and to get some understanding of their relevance to BMI applications. This study does not attempt to address the many other important avenues for investigation in BMI or spike train signal processing. For BMI performance, these avenues include at least: prosthetic decode algorithms (Brockwell et al., 2004; Brown et al., 1998; Georgopoulos et al., 1986; Srinivasan et al., 2007; Wu et al., 2004, 2006; Yu et al., 2007), recording technology (Wise, Anderson, Hetke, Kipke, & Najafi, 2004), the design of prosthetic end effectors and interfaces, be that a robotic arm or computer screen (Cunningham, Yu, Gilja, Ryu, & Shenoy, 2008; Schwartz, 2004; Velliste et al., 2008), and multiple signal modalities (*e.g.*, EEG, ECoG, LFP, and spiking activity) (Mehring et al., 2003). Two reviews in particular give a thorough overview of these and other important areas of BMI investigation (Lebedev & Nicolelis, 2006; Schwartz, 2004). For spike train signal processing, there are also many avenues of research not addressed in this study, including at least: spike-sorting (Lewicki, 1998), information-theoretic studies (Borst & Theunissen, 1999; Nirenberg, Carcieri, Jacobs, & Latham, 2001), neural correlations (Pillow et al., 2008; Shlens et al., 2006), methods for multiple simultaneously recorded neurons (Chapin, 2004; Churchland, Yu, Sahani, & Shenoy, 2007; Yu et al., 2009), and more accurate spiking models (Barbieri, Quirk, Frank, Wilson, & Brown, 2001; Johnson, 1996; Kass & Ventura, 2003; Koyama & Kass, 2008; Truccolo, Eden, Fellows, Donoghue, & Brown, 2004; Ventura, Carta, Kass, Gettner, & Olson, 2002). Two reviews in particular discuss these and other issues in spike train processing (Brown, Kass, & Mitra, 2004; Kass, Ventura, & Brown, 2005).

Linking methodological developments to observable physical behavior (such as neural prosthetic decode performance) is critical for increasing the adoption and usefulness of these methods. This study takes an important first step in that direction for the problem of firing rate estimation.

2. Firing rate methods

This section reviews several popular and current firing rate estimation methods. We introduce each method at a high level, point to relevant publications, and suggest potential advantages and disadvantages of each. We then summarize the reviewed methods and discuss related methods and other possibilities that are not yet included in literature.

2.1. Kernel smoothing (KS)

The most common historical approach to the problem of estimating firing rates has been to collect spikes from multiple trials in a time-binned histogram known as a *peri-stimulus-time histogram* (PSTH), which produces a piecewise constant estimate. To achieve a smooth, continuous firing rate estimate, as is often of interest in single trial settings (such as neural prostheses), researchers instead typically use kernel smoothing (KS); that is, they convolve the spike train with a kernel of a particular shape (*e.g.*, Nawrot et al. (1999)). This convolution produces an estimate where the firing rate at any time is a weighted average of the nearby spikes (the weights being determined by the kernel). A Gaussian shaped kernel is most often used (see, *e.g.*, Kass et al. (2005)), and this kernel serves to smooth the spike data to a firing rate that is higher in regions of spikes, lower otherwise. However, the kernel shape and timescale (*e.g.*, the standard deviation of the Gaussian) are frequently chosen in an *ad hoc* way, which greatly alters the frequency content of the resulting estimate (in other words, how quickly firing rate can change, and how susceptible the estimate is to noise).

The most obvious advantage of kernel smoothing is its simplicity. KS methods are extremely fast and simple to implement, which has led to wide adoption. In this study, we implement three Gaussian kernel smoothers of various bandwidths (which determine smoothness): 50 ms standard deviation (KS50), 100 ms (KS100), and 150 ms (KS150). These are common choices for single trial studies, and they produce significantly different estimates of firing rate. This *ad hoc* choice of smoothness is typically considered a major disadvantage of KS methods.

2.2. Adaptive kernel smoothing (KSA)

In Richmond, Optican, and Spitzer (1990), the authors address two concerns with standard KS: first, the *ad hoc* smoothness choice as noted above, and second, the fact that the kernel width can not adapt at different regions of smoothness in the firing rate. We call this fixed frequency behavior *stationarity*. KSA incorporates a nonstationary kernel to allow the spike train to determine the extent of firing rate smoothness at various points throughout the trial. It does so by forming first a stationary firing rate estimate (called a pilot estimate), and from that pilot, it forms a set of local kernel widths at the spike events. These local kernels are then used to produce a smoothed firing rate that changes more rapidly in regions of high firing, and less in regions of less firing. This trend is sensible, as regions of little spiking give fewer observations into the firing rate process underlying the data.

KSA benefits from the simplicity of KS methods, and the added complexity of the local kernel widths increases the computational effort only very slightly. Further, this approach lifts the strict stationarity requirement of many methods. A possible shortcoming is that, even though it adapts the kernel width, KSA still requires an *ad hoc* choice of kernel width for the pilot estimate.

2.3. Kernel bandwidth optimization (KBO)

In KS methods, as latter sections in this paper will show, the *ad hoc* choice of smoothness can have a significant impact on the firing rate estimate. KBO seeks to remove this shortcoming of kernel smoothing by establishing a principled approach to choosing the kernel bandwidth. In Shimazaki and Shinomoto (2007b), a method is developed for automatically choosing the bin width of a PSTH. By assuming that neural spike trains are generated from an inhomogeneous Poisson process (*i.e.*, a Poisson process with time-varying firing rate), the authors show that the mean squared error (MSE) between the PSTH and the true underlying firing rate can be minimized using only the mean rate (rate averaged across time), *without* knowledge of the true underlying firing rate.

In Shimazaki and Shinomoto (2007a), this PSTH method is adapted to similarly optimize the bandwidth of a smoothing kernel. The authors of that report provide a simple algorithm for the popular Gaussian kernel, which we implemented for the purposes of this study. Once the optimal kernel bandwidth is chosen with the algorithm of Shimazaki and Shinomoto (2007a), we then perform standard kernel smoothing (as defined in KS above) with the optimized kernel bandwidth. We refer to this method as KBO.

We also note here a method quite similar in spirit to KBO. In Nawrot et al. (1999), a heuristic method is developed to find the optimal bandwidth of a kernel smoother. We also implemented this method and found that, with the particular motor cortical data of interest for this BMI study, the method of Nawrot et al. (1999) produced very often a flat, uninformative firing rate function (*i.e.*, a very large kernel bandwidth). Accordingly, we chose the newer, principled method of Shimazaki and Shinomoto (2007a, 2007b) (which produces a range of different kernel bandwidths, depending on the spike data) to demonstrate the performance of kernel bandwidth optimization methods.

KBO has the advantage of simple implementation and correspondingly very fast run time (only slightly longer than a regular kernel smoother, due to the overhead required to calculate the optimal bandwidth). Shortcomings of this approach may include the Poisson spiking assumption (required for this method), as much research has shown that neural spiking often deviates significantly from Poisson spiking statistics (see, *e.g.*, Barbieri et al. (2001), Miura, Tsubo, Okada, and Fukai (2007) and Paninski, Pillow, and Simoncelli (2004)).

2.4. Gaussian process firing rates (GPFR)

All kernel smoothing methods, including KS, KSA, and KBO as above, act as low pass filters to produce a smooth, time-varying firing rate. Alternatively, several methods take a probabilistic approach. If one assumes a prior probability distribution for firing rate functions (*e.g.*, some class of smooth functions), and a probability model describing how spikes are generated, given the underlying firing rate (*e.g.*, an inhomogeneous Poisson process (Daley & Vere-Jones, 2002)), one can then use Bayes rule (Papoulis & Pillai, 2002) to infer the most likely (or expected) underlying firing rate function, given an observation of one or multiple spike trains. The methods GPFR, BARS, and BB are variations on this general approach.

In Cunningham, Yu, Shenoy, and Sahani (2008), firing rates are assumed *a priori* to be draws from a Gaussian process. Gaussian processes place a probability distribution on firing rate functions which allows all functions to be possible, but strongly favors smooth functions (Rasmussen & Williams, 2006). This study then assumes that, given the firing rate function, spike trains are generated according to an inhomogeneous Gamma interval process, which is a generalization of the familiar Poisson process to allow spike history effects such as neuronal refractory periods. Bayesian model selection and Bayes' rule are then used to infer the most likely underlying firing rate function, given an observation of one or multiple spike trains. Owing to this probabilistic model, the computational overhead of such a firing rate estimator can be significant, so the authors developed numerical methods to alleviate these challenges (Cunningham, Sahani, & Shenoy, 2008).

GPFR has the advantage of using a probabilistic model, which allows automatic smoothness detection (in contrast to the *ad hoc* smoothness choices made in, for example, KS), and which naturally produces error bars on its predictions (which may be useful for data analysis purposes). GPFR also has the benefit of being able to readily incorporate different *a priori* assumptions about firing rate (such as known, stimulus-driven nonstationarities in the firing rate, which can be controlled through the Gaussian process prior). Even with the significant computational improvements developed in Cunningham, Sahani et al. (2008), GPFR still requires seconds of computational resource (for spike trains roughly one second in length), which may be a disadvantage compared to kernel smoothers (which work in tens to hundreds of milliseconds).

2.5. Bayesian adaptive regression splines (BARS)

Instead of a Gaussian process prior on smooth firing rate functions, BARS, as introduced and used in Behseta and Kass (2005), Dimatteo, Genovese, and Kass (2001), Kass et al. (2005) and Kaufman, Ventura, and Kass (2005), models underlying firing rate with a spline basis. Splines generally are piecewise polynomial functions that are connected at time points called "knots". In Dimatteo et al. (2001), the authors choose a prior distribution on the number of knots, the position of the knots, and other parameters of the spline function. Conditioned on firing rate, BARS then assumes that spikes are generated according to a Poisson spiking process.

This model choice allows Bayesian inference to be carried out. Owing to the forms of the probability distributions chosen, approximate inference methods must be used (an analytical solution is intractable). BARS uses the well established techniques of reversible-jump Markov chain Monte Carlo and Bayesian information criteria to estimate the underlying firing rate (which in this case is the mean of the approximate posterior distribution, given the observed data). BARS is fully described in Dimatteo et al. (2001), and further applications and explanations can be found in Behseta and Kass (2005), Kass et al. (2005), Kaufman et al. (2005) and Olson, Gettner, Ventura, Carta, and Kass (2000). This study uses the MATLAB implementation of BARS available at the time of publication at http://lib.stat.cmu.edu/~kass/bars/bars.html.

One major advantage of BARS is that the spline basis allows different regions of firing rate to change more or less smoothly, which allows high frequency changes in rate while still removing high frequency noise (this is not possible in traditional kernel smoothers). Further, like other probabilistic methods, BARS produces an approximate posterior distribution on firing rates, so valuable features like error bars are available. BARS, like GPFR, suffers from technical complexity that translates into meaningful computational effort and run-time, compared to more basic kernel smoothers.

2.6. *Bayesian binning (BB)*

Instead of assuming a continuous, time-varying firing rate as in many of the above approaches, the authors of Endres, Oram, Schindelin, and Foldiak (2008) assume neural firing rates can be modelled *a priori* by piecewise constant regions of varying width (in contrast to a fixed-width binning scheme like the classic PSTH). This BB approach, like BARS and GPFR, constructs a probabilistic model for spiking, where both the firing rates in piecewise constant regions and the boundaries between the regions themselves have associated probability distributions (together, the boundaries and the firing rates at each interval fully specify a firing rate function). BB then assumes an inhomogeneous Bernoulli process for spiking (*i.e.*, each time point contains 0 or 1 spikes), given the underlying firing rate.

With these assumptions made, Bayes rule is then used to infer the underlying firing rate from the above model. Importantly, because the boundaries and height of the firing rate bins are probabilistic, the result of this firing rate inference is a smooth, time-varying firing rate, and BB is thus comparable to the other methods highlighted in this study. The BB method is fully described in Endres et al. (2008), and we implemented the algorithm using the authors' source code, which is available at the time of this report at http://mloss.org/software/view/67/.

Like GPFR and BARS, BB has the advantage of being a fully probabilistic model, which allows automatic smoothness detection (in contrast to the *ad hoc* smoothness choices made in, for example, KS), and which produces error bars on its predictions. Also, like BARS and KSA (and unlike GPFR, KBO, and KS), BB is a nonstationary smoothing model, so it can adapt its smoothness to regions of faster or slower firing rate changes. However, as BB constructs a thorough probabilistic model for spiking and solves it exactly, the method requires significant computational resource (generally an order of magnitude more than BARS and GPFR, the other computationally expensive methods), which may limit the use of BB in some applications.

2.7. *Summary of reviewed methods*

These methods were chosen in that they all can be used as single trial, single neuron firing rate estimators (as is relevant for neural prosthetic applications). In Fig. 2, we show four examples of firing rates inferred by all eight methods reviewed above. Each panel represents a different spike train, which is denoted above the firing rates as a train of black rasters (as in Fig. 1). These four panels show a range of spiking patterns, including: (a) high firing, (b) sharply increasing activity, (c) sharply decreasing activity, and (d) low firing. Though there are an infinite number of possible firing rate patterns, these four example spike trains illustrate the wide range of firing rate profiles that can be estimated from the same neural activity, depending on the estimation method used.

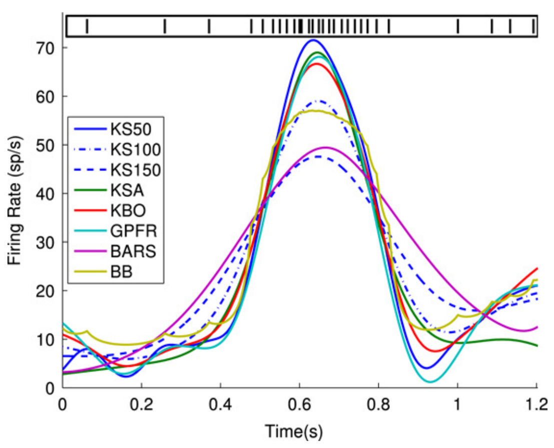
The methods above also demonstrate a range of approaches and features that one might consider in designing a firing rate estimator. Table 1 compares the above methods in terms of five important features, where we indicate generally desirable features in green and undesirable features in red. The first row notes which methods offer principled, automatic determination of the firing rate smoothness (vs. choosing a kernel bandwidth in an *ad hoc* way). The second row indicates whether the method is a proper probabilistic model, which carries advantages previously discussed. The drawback of probabilistic models lies in their computational complexity (and, as a result, run time); the third row of Table 1 details ballpark run-time requirements for estimating one firing rate function from one single spike train. The fourth row details which methods are nonstationary; that is, which methods can adapt the smoothness of the estimate at different points in the spike train. Finally, we also noted above that spike trains are known to depart significantly from Poisson statistics (*e.g.*, refractory periods); the fifth row illustrates which methods are Poisson based and which are not.

It is important to note that all of these methods can also be used for multiple-trial firing rate analyses. Some methods, including BARS and BB, were introduced more with a multi-trial motivation than a single-trial motivation. This study makes no claim on the effectiveness of any of these methods at larger numbers of trials, as such a circumstance is not germane to BMI applications. Thus, the forthcoming results should not be viewed as a statement about the quality of a particular firing rate estimator *in general*, but rather for the single-trial analyses that are relevant in BMI studies.

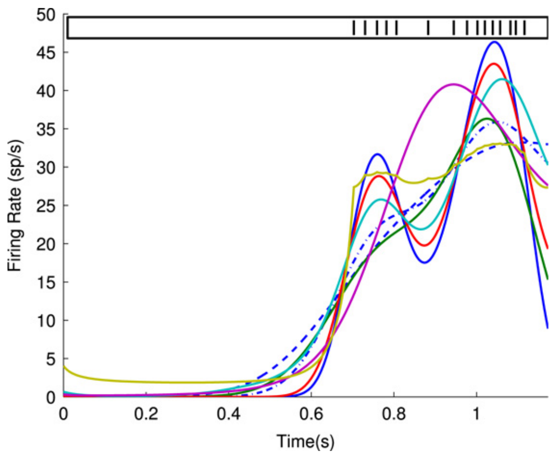
2.8. *Other related methods*

Despite the range of methods already discussed, the above list of recent and classical firing rate estimators is by no means exhaustive. We here discuss a few other possibilities and avenues of investigation not covered by the above methods.

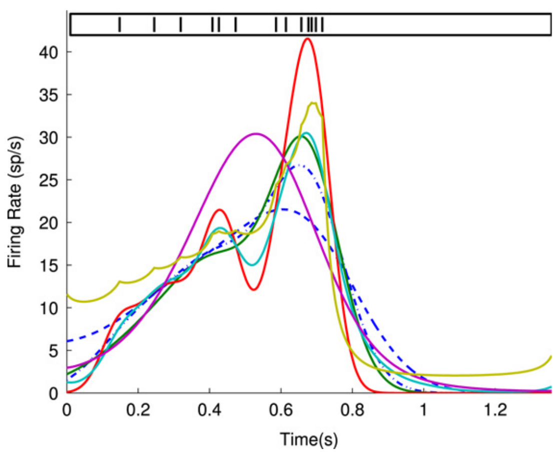
First, we note that none of the above methods are implemented as cross-validation schemes (Bishop, 2006). The probabilistic models (GPFR, BARS, BB) all do Bayesian model selection to adapt their smoothness. KBO uses an MSE criterion and KSA uses a criterion based on the amount of local spiking to adapt their smoothness, whereas KS uses only a user-defined kernel width choice. Another possibility is to cross-validate, where other trials of data are used to inform the parameter (*e.g.*, smoothness) choices when estimating firing rate on a novel spike train (Bowman (1984) reports on the related topic of probability density estimation). For example, one might believe that all firing rates in a particular BMI application evolve with roughly equal smoothness. Even though the firing rates may be quite different trial to trial, one could cross-validate with some criterion (such as decode performance) to choose the smoothness for the firing rate estimation on the new spike train in question. This report does not review that possibility, as we wish to focus on methods that produce firing rates from spike trains based on *only* those spike trains (not a validation set). Further, many, if not all, of the above methods could incorporate a cross-validation scheme: for example, GPFR, BARS, and BB could choose their parameters via cross-validation instead of Bayesian model selection. Thus, cross-validation is a feature of model selection more than it is of the firing rate method used, and we chose to focus on the methods as previously published.



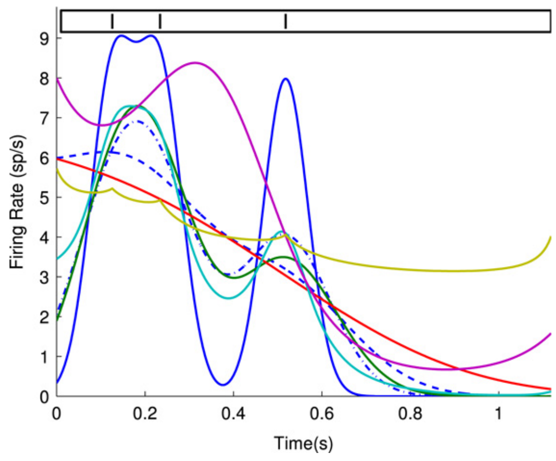
(a) A high firing rate (data from L2006B.196.243).



(b) A sharply increasing firing rate (L2006B.301.60).



(c) A sharply decreasing firing rate (L2006B.170.92).



(d) A low firing rate (L2006B.170.216).

Fig. 2. Example of various firing rate methods applied to data from different neurons and different trials. Each method (see legend) produces a smooth estimate of underlying firing rate from each of the four separate spike trains. The spike trains are represented as a train of black rasters above each panel. Note that KBO obscurs KS50 in panel (c).

Table 1
Summary of firing rate methods reviewed in this report.

	Method					
	KS	KSA	KBO	GPFR	BARS	BB
Automatic smoothness detection	N	N	Y	Y	Y	Y
Probabilistic model	N	N	N	Y	Y	Y
Runtime/computational complexity	millisec.	millisec.	millisec.	seconds	seconds	minutes
Nonstationary smoothness	N	Y	N	N	Y	Y
Spike history effects	N	N	N	Y	N	N

Second, we also note that the methods outlined above are all *unsupervised*, in that they infer firing rates without knowledge of an extrinsic covariate such as the path of a rat foraging in a maze, or the kinematic parameters of a moving arm. Instead, if one has a good idea about how some measureable behavior translates to firing rate, one might assume a parametric form for firing rate based on behavior, learn the parameters from the data, and use that model to infer time-varying firing rate. Some studies using this approach include Barbieri et al. (2001), Brown et al. (1998), Brown, Barbieri, Ventura, Kass, and Frank (2002), Eden, Frank, Barbieri, Solo, and Brown (2004), Pillow et al. (2008), Stark, Drori, and Abeles (2006), Truccolo et al. (2004) and Ventura et al. (2002). These approaches are specific to particular neural areas, particular experimental set-ups, and they are susceptible to biases of their own. Thus, we chose not to review these techniques to again focus on methods that produce firing rates for a given spike train, using that spike train alone.

Third, we note that, although the methods described above are quite specific, there are many areas in which they can be extended or combined with other approaches. Simple first examples include: KBO and KSA could be combined in a two-stage method, or the method of Miura et al. (2007) could replace a part of the model selection method in GPFR. As a more interesting example, one advantage of probabilistic models (including BARS, GPFR, and BB) is that they can readily be extended to different spiking probability models. One spiking model, the so-called generalized linear model (GLM), has received much attention of late (Barbieri et al., 2001; Coleman & Sarma, 2007; Czanner et al., 2008; Eden et al., 2004; Koyama & Kass, 2008; Pillow et al., 2008; Srinivasan et al., 2007; Truccolo et al., 2004) for its ability to model neural spiking quite well and its flexibility in being extended to many different problem domains. This GLM spiking model may inform firing rate estimation as well.

Finally, we note that all the methods above are single-neuron firing rate estimators that are independent of the activity of other neurons. Firing rate estimation methods that consider multiple units (as is often collected with electrode arrays in BMI experiments) may be able to leverage the simultaneity of recordings to improve the quality of firing rate estimates. Some work has begun to investigate this general question, including (Brown et al., 2004; Chapin, 2004; Churchland et al., 2007; Pillow et al., 2008; Yu et al., 2009) (and the GLM model of Czanner et al. (2008) could also be readily extended for this purpose). However, none of these multidimensional approaches specifically address unsupervised firing rate estimation as do the methods of this report, so we will leave multidimensional extensions to future work.

In summary, the problem of firing rate estimation (and, more generally, inferring meaningful information from spiking data) is quite broad. The methods reviewed in this report are all directly comparable, but there are many opportunities for extensions and adaptations of these models.

3. Prosthetic paradigm for evaluating firing rate methods

Having reviewed several firing rate estimators, we now investigate their relevance for neural prosthetic applications. We first describe the experimental setting we employed to study this question (Section 3.1). We then describe two popular prosthetic decoding algorithms (Section 3.2) and performance metrics (Section 3.3) that we can use to evaluate the quality of our firing rate estimation.

3.1. Reach task and neural recordings

Animal protocols were approved by the Stanford University Institutional Animal Care and Use Committee. We trained an adult male monkey (*Macaca mulatta*) to perform point-to-point reaches on a 5-by-5 grid (25 targets) for juice rewards. Visual targets were back-projected onto a fronto-parallel screen 30 cm in front of the monkey. The monkey began each trial with his hand held at a particular target, which had to be held for a random time interval. These hold times were exponentially distributed with a mean of 300 ms (but shifted to be no less than 150 ms). This exponential distribution prevented the monkey from preempting the movement cue. After the hold time, a pseudo-randomly chosen target was presented at one of the target locations. The 25 targets were spaced evenly on an 8 cm by 8 cm grid. Concurrent with the target presentation, the current hold point disappeared, cueing the monkey to reach to the target (the "go cue"). The monkey was motivated to move quickly by a reaction time constraint (maximum allowable reaction time of 425 ms, minimum of 150 ms, again to prevent preemption). The monkey reached to the target and then held the target for 300 ms, after which the monkey received a liquid reward. The next trial started immediately after the successful hold period. In total, all trials are 850 to 1500 ms long (these times vary depending on the length and speed of the reach and the randomized hold time). Fig. 3 illustrates four sequential trials of the reaching task.

During experiments, the monkey sat in a custom chair (Crist Instruments, Hagerstown, MD) with the head braced. The presentation of the visual targets was controlled using the Tempo software package (Reflective Computing, St. Louis, MO). A custom photo-detector recorded the timing of the video frames with 5 ms resolution. The position of the hand was measured in three dimensions using the Polaris optical tracking system (Northern Digital, Waterloo, Ontario, Canada; 60 Hz, 0.35 mm accuracy), whereby a passive marker taped to the monkey's fingertip reflected infrared light back to the position sensor. Eye position

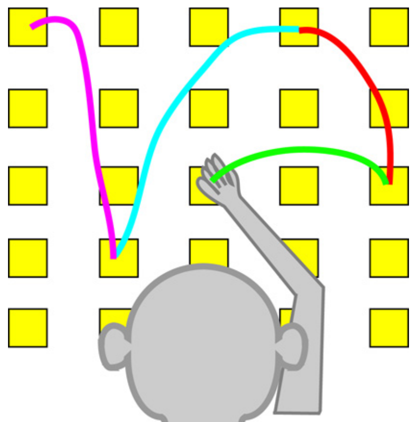


Fig. 3. Cartoon of the reaching task as in L2006A and L2006B. Four sample trials are shown (one each in magenta, cyan, red, and green).

was tracked using an overhead infrared camera (Iscan, Burlington, MA; 240 Hz, estimated accuracy of 1°).

A 96-channel silicon electrode array (Cyberkinetics, Foxborough, MA) was implanted straddling dorsal pre-motor (PMd) and motor (M1) cortex (left hemisphere), as estimated visually from local landmarks, contralateral to the reaching arm. Surgical procedures have been described previously (Churchland, Yu, Ryu, Santhanam, & Shenoy, 2006; Hatsopoulos, Joshi, & O'Leary, 2004; Santhanam, Ryu, Yu, Afshar, & Shenoy, 2006). Spike sorting was performed offline using techniques described in detail elsewhere (Sahani, 1999; Santhanam, Sahani, Ryu, & Shenoy, 2004; Zumsteg et al., 2005). Briefly, neural signals were monitored on each channel during a two minute period at the start of each recording session while the monkey performed the behavioral task. Data were high-pass filtered, and a threshold level of three times the RMS voltage was established for each channel. The portions of the signals that did not exceed threshold were used to characterize the noise on each channel. During experiments, snippets of the voltage waveform containing threshold crossings (0.3 ms pre-crossing to 1.3 ms post-crossing) were saved with 30 kHz sampling. After each experiment, the snippets were clustered as follows. First, they were noise-whitened using the noise estimate made at the start of the experiment. Second, the snippets were trough-aligned and projected into a four-dimensional space using a modified principal components analysis. Next, unsupervised techniques determined the optimal number and locations of the clusters in the principal components space. We then visually inspected each cluster, along with the distribution of waveforms assigned to it, and assigned a score based on how well separated it was from the other clusters. This score determined whether a cluster was labeled a single-neuron unit or a multi-neuron unit. For this report, as many firing rate methods are based on biophysical properties of single neurons, we use units labelled only as high quality, single-neuron units.

The monkey (monkey L) was trained over several months, and multiple data sets of the same behavioral task were collected. We chose two such data sets to evaluate prosthetic decode (L2006A and L2006B), from which we took 14 and 15 high quality, single-neuron units, respectively (note that more units would be available were we to consider "possible single units" or multi-units, as is often done in prosthesis studies). For the purposes of this study, we selected the first 300 successful trials (about five minutes of neural activity and physical behavior), which is ample for fitting the decoding models used here. Thus, we use two data sets, each with 14 or 15 neural units and 300 experimental trials. This produces a total of 8700 spike trains that were all analyzed by each of the eight firing rate methods (and subsequently by the two decoding algorithms). Across all these firing rate estimations and their subsequent prosthetic decodes, this analysis required roughly four weeks of fully dedicated processor time on five to ten 2006-era workstations (Linux Fedora Core 4 with 64 bit, 2.2–2.4 GHz AMD processors and 2–4 GB of RAM) running MATLAB.

3.2. Decoding algorithms

Having detailed the experimental collection of neural spike trains and physical behavior, and having reviewed methods for processing spike trains into firing rates, we now address how to decode arm trajectories from neural firing rates. As with the firing rate methods above, we discuss the methods at a high level and point to the relevant literature which offers more methodological description.

3.2.1. Linear Decode

The linear decode algorithm, as used for example in Carmena et al. (2005), Chestek et al. (2007) and Hochberg et al. (2006), is a simple first approach to decoding arm trajectories from neural activity. This algorithm assumes the physical behavior at a particular time t is a linear combination of all recorded neural activity (across all N recorded neural units) that precedes t by some amount of time. We chose to consider the preceding 300 ms of neural activity.[2] This period of neural activity can be considered a row in a matrix of firing rates (as many rows as time points in the experimental trials). If each dimension of the behavior (*e.g.*, horizontal hand position and vertical hand position) is a vector of length also equal to the number of time points, then simple least squares can solve for the linear weights that relate neural activity to physical behavior. These weights can then be applied to novel neural activity to produce a decoded reach trajectory, which hopefully matches the true reach well. More mathematical details can be found in, *e.g.*, Chestek et al. (2007).

For completeness, we note here a few specifics of our implementation of this algorithm. To provide the algorithm with a finely time-resolved firing rate, we sampled the firing rate estimates (from all firing rate methods) every 5 ms. We found that increasing this sampling rate did little more than increase the computational burden of the decode, and reducing this rate ignored features of the firing rate estimates, which would be detrimental to our comparison of methods. Further, because of the 300 ms integration window and the trial structure of the data (there is a time break in between each trial), for the decode analysis, we decode only the length of the trial beginning 300 ms after the beginning of the trial (this prevents the linear decode filter from going into a region of undefined neural activity). Owing to the random hold time and the reaction time of the monkey (both enforced to be no less than 150 ms, see Section 3.1), there was no movement for the first 300 ms of the trial, so this step is reasonable. Furthermore, we found that including this portion of the trials did not change the result considerably.

3.2.2. Kalman Filter

To employ the popular Kalman filter (Kalman, 1960), we assume that the arm state (in this case, horizontal and vertical position and velocity) evolves as a linear dynamical system: the arm state at discrete time t is a linear transformation of the arm state at time $t-1$, plus Gaussian noise. We also assume a linear relationship between arm state and neural activity at that time t (again, plus noise). With this done, the Kalman filter allows the inference of the hand state from the observation of neural data only. Starting from arm state at the beginning of the trial, the Kalman filter proceeds iteratively through time, updating its estimates of arm state and error covariance at every time step t, before and after the inclusion of neural data at that time step. These steps are entirely based on mathematical properties of the Gaussian, and the algorithm is fast and stable. Importantly, the Kalman filter has been previously and successfully used as a BMI decoding algorithm, and more explanation and mathematical detail can be found in Wu et al. (2002, 2004, 2006).

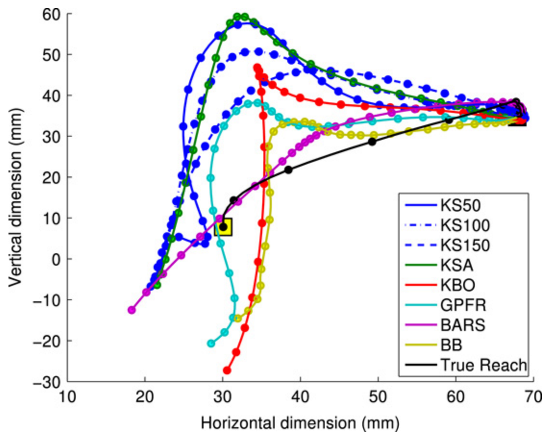
As above, we note here a few implementation specifics. To parallel with the linear decode, we also sampled firing rates at 5 ms intervals when fitting the Kalman filter model and when estimating reach trajectories from it. In the linear decode, we chose to remove the first 300 ms of the trial, during which the monkey did not move. In the Kalman filter decodes, we truncated 300 ms from the end of the trials. Choosing this slightly different time interval allows us to look across the linear decode and the Kalman filter and rule out any potential idiosyncracies with the starting and ending of a trial. We also varied this choice and found that it had no effect on the relative decode performance of the different firing rate methods. Next, we note that we included horizontal and vertical position and velocity in our arm state. Acceleration is sometimes included, but the inclusion of this data in our Kalman filter had little effect on the decode quality, so we chose not to consider it further. Finally, we note that we did not impose a temporal lag (or a group of lags) between neural data and physical behavior. Our testing with different lags produced minor differences that agreed generally with the results of Wu et al. (2006). As this aspect did not influence the comparisons between firing rate estimators, we do not report further on it.

To provide a sample of these decodes, we show in Fig. 4 four decoded trials from L2006A that use the Kalman filter. Each panel shows the true reach as a black trace moving from the black square hold point to the yellow square target. Trajectories decoded with each firing rate method (but *the same* neural data) are shown in colors corresponding to those in Fig. 2 (see legend). Marks are placed on each trajectory at 20 ms intervals to give an idea of decoded velocity profiles. Panels (a) and (b) show reasonably average decodes (in terms of the RMS error, see the panel captions). Panel (c), a trial which decodes rather well, shows the wide variety of decoded trajectories that can arise from different firing rate estimations (but the same spike trains). Finally, Fig. 4, panel (d), shows that indeed the Kalman filter, like the linear decode (not shown) does sometimes fail entirely to decode the true reach, regardless of the firing rate method used. In the following sections, we generalize these specific examples, calculating performance metrics across all trials, decode methods, and data sets.

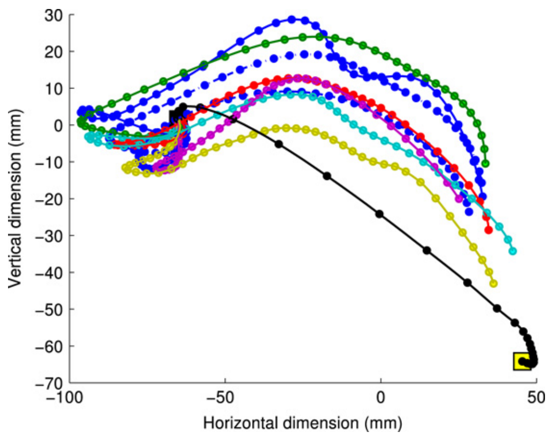
3.3. Calculating decode performance

Given any decoded arm trajectory, there are a number of possible metrics to evaluate accuracy. We use two of the most common metrics: root-mean-square error (RMSE) and correlation coefficient. For any given firing rate method, RMSE on each trial is the square root of the mean of the squared errors (across time) between the true arm trajectory and the decoded trajectory. RMSE is likely the most often-used performance metric; some examples of its use (or MSE, which is simply RMSE squared) include: Brockwell et al. (2004), Kemere et al. (2004), Serruya et al. (2002), Srinivasan et al. (2007), Wu et al. (2006) and Yu

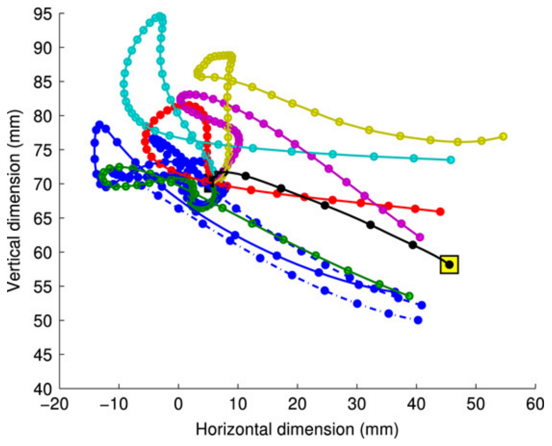
[2] We chose 300 ms as a number on par with the timescale of arm movements and motor processing. Ideally, one might run this analysis at a variety of temporal window sizes. However, we note that this choice has no discernable bias in favor of any particular firing rate estimation method. We also found that using 300 ms produced decode results of similar quality to using longer periods. Finally, we note that the Kalman filter does not make this assumption, providing yet another cross-check.



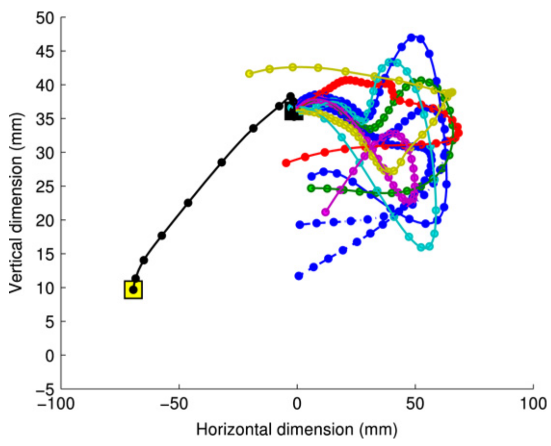
(a) Reasonable decodes (data from L2006A.231). Horizontal (vertical) RMSE had range 20.5–26.0 mm (11.0–15.7 mm).



(b) Reasonable decodes (L2006A.154). Horizontal (vertical) RMSE had range 20.1–30.3 mm (23.8–39.5 mm).



(c) Better decodes (L2006A.69). Horizontal (vertical) RMSE had range 5.6–10.7 mm (3.2–13.8 mm).



(d) Failed decodes (L2006A.169). Horizontal (vertical) RMSE had range 42.6–48.4 mm (5.2–9.8 mm).

Fig. 4. Example of decoded arm trajectories derived from different firing rate estimates of the same neural data (see legend). All data shown are decoded using a Kalman filter and the data set L2006A. In all cases the true reach is shown in black (moving from the black square hold point to the yellow square target). To give an idea of the velocity profile of the true reach and decoded trajectories, marks are placed on each trajectory at 20 ms intervals. (Note that the true reach, in black, has a cluster of marks at the trial start. These marks, which are obscured by other decodes, indicate that the arm is stationary for the early part of the trial. All decoders have difficulty decoding this stationary period). To compare these results to the results across all trials, each panel quotes the range of RMS errors (across the different decoders) in the horizontal and vertical dimensions (*cf.* Fig. 5, panels (a) and (b)).

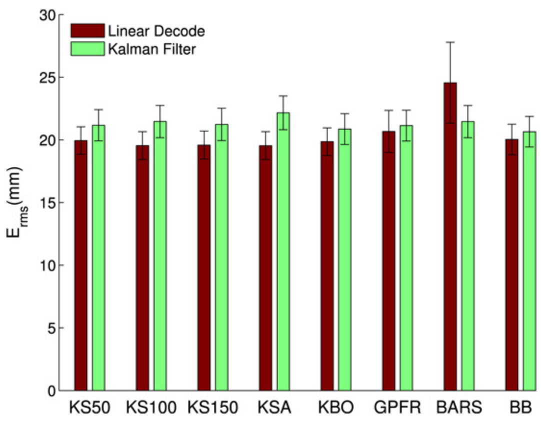
et al. (2007). Correlation coefficient (ρ or r^2) is another commonly used performance metric that reflects how well the decoded trajectory matches the true arm trajectory. Considering each time step as a draw from a random variable, this metric correlates the true and decoded trajectories across time to calculate how well one trajectory predicts the other ($\rho = 1$ implies perfect linear correlation). Some previous literature using correlation coefficients to evaluate decode performance includes (Carmena et al., 2005; Chestek et al., 2007; Wu et al., 2002, 2006).

To calculate these performance metrics, we use leave-one-out cross validation (LOOCV) (Bishop, 2006). That is, for each data set, we select one experimental trial (one arm trajectory) to test, and we exclude both that trial's neural activity and physical behavior. We then train a decoder model based on the other 299 trials collected in that data set (L2006A or L2006B). We can then use the decode algorithm (linear decode or Kalman filter) to decode the arm trajectory on the excluded trial, using only the neural activity from that trial. We repeat this same procedure 300 times (once per trial), which provides 300 decoded trials. We calculate the RMSE for each trial, and then we can average these and produce 95% confidence intervals (Zar, 1999). We also correlate all the decoded arm trajectories with the true trajectories, producing one overall correlation coefficient ρ and 95% confidence intervals on the estimate of this metric (see Zar (1999), Section 19.3 for details on calculating confidence intervals for a population correlation coefficient).

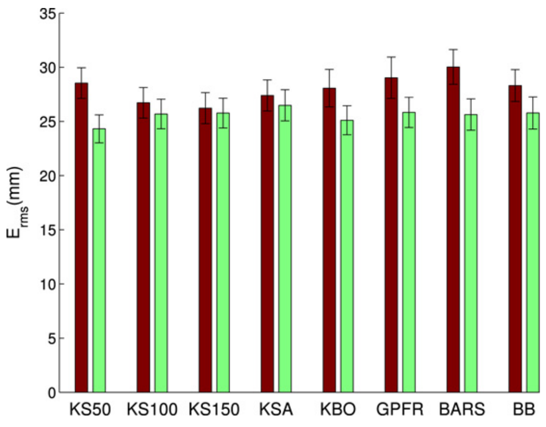
4. Performance results

In Section 2, we described a host of methods that estimate firing rates from experimentally gathered spike trains. We then used these firing rates to decode arm trajectories using two different decoding algorithms (Section 3.2) and two different performance measures (Section 3.3). We now compare different firing rate estimation methods in terms of their decode performance.

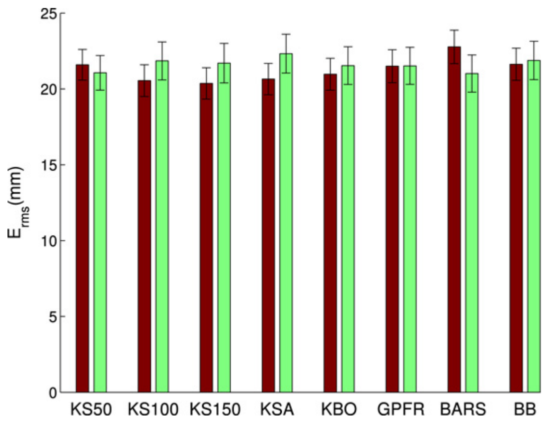
In Figs. 5 and 6, we show the RMSE and correlation coefficient results (respectively) from several different decoding scenarios. Each panel shows the decode performance across all eight of the reviewed firing rate methods (KS50, KS100, KS150, KSA, KBO, GPFR, BARS, and BB). Within each panel, red bars represent the decode error using the linear decode method, and green bars represent the decode error using the Kalman filter method. Panels (a) and (b) show decoding results from data set L2006A, and panels (c) and (d) show results from data set L2006B. Also, the left panels (a and c) and the right panels (b and d) show the results from decoding horizontal and vertical hand position, respectively. Thus,



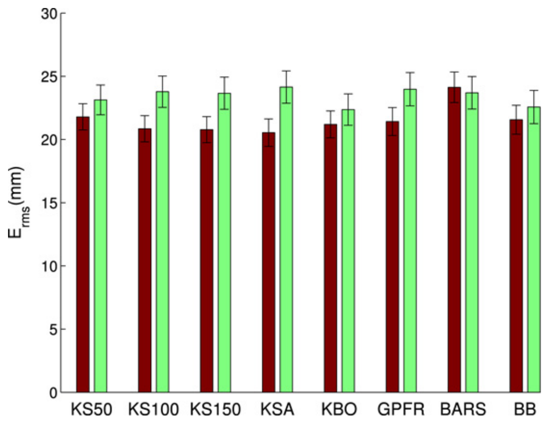
(a) Horizontal hand position, L2006A.



(b) Vertical hand position, L2006A.



(c) Horizontal hand position, L2006B.



(d) Vertical hand position, L2006B.

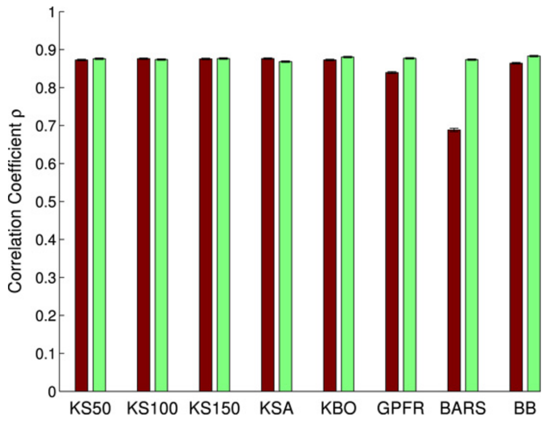
Fig. 5. The decode performance of spike trains smoothed with different firing rate methods. Error is root mean squared error (RMSE). In all panels, red bars are decode performance with a linear decode; green bars are performance numbers with a Kalman filter. Error bars indicate the 95% confidence interval.

each firing rate estimate has 16 performance metrics (two decode methods, two data sets, horizontal and vertical dimensions, RMSE and correlation coefficient). This variety is important to ensure that any effects are robust across data sets and decode algorithms and different strengths of neural tuning.

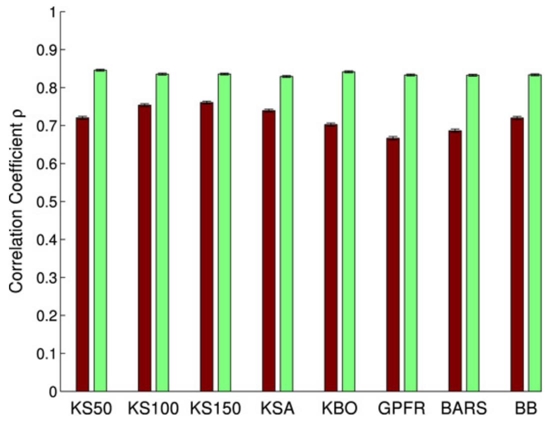
First, we note several important cross-checks with existing literature. The RMSE and correlation coefficient numbers match well to the results of, for example, Chestek et al. (2007), Wu et al. (2002, 2006) and Yu et al. (2007). The errors are in some cases higher than those seen in previous literature, which may be due to the complexity of this task (vs. a simpler, center-out task as in Yu et al. (2007)) or the restrictive choice of using only single neural units (rather than the many more multi-units which are often informative in a decode setting). Indeed, when we altered the number of neural units, the absolute decode performance changed as expected, but the relative differences between the decode results (from the various firing rate methods) did not. Accordingly, we are satisfied that the selected neural populations are representative. Specifically to the linear decode, our performance may also be different in that we used only 300 ms of preceding neural data vs. prior literature which has used, for example, 1000 ms (Chestek et al., 2007) or 550 ms (Wu et al., 2002). Specifically to the Kalman filter, as noted above, our performance may also be different in that we did not impose a temporal lag between neural data and physical behavior. Again, we tested changing the temporal lags and found relative performance between firing rate methods insensitive to this choice, so we are satisfied that this choice is also representative. We also visually compared trajectories decoded in this study (*e.g.*, Fig. 4) to decoded trajectories from Wu et al. (2002, 2006), and we found these to be similar, giving confidence that we are successfully reproducing similar decode quality as existing literature.

The most salient feature in Figs. 5 and 6 is the similarity in performance across all firing rate methods. Let us consider, for example, the Kalman filter results from Fig. 5, panel (a). Looking across these eight green bars, there is no statistically significant difference between the RMSE results produced by any of the methods. If we consider different decoding algorithms (linear decode — red bars or Kalman filter — green bars), different performance metrics (RMSE — Fig. 5 or correlation coefficient — Fig. 6), different dimensions of physical activity (horizontal — left panels or vertical — right panels), and different data sets (L2006A — upper panels or L2006B — lower panels), the story is unchanged: all seem to produce very similar performance results no matter what firing rate estimation method is used. In some cases the Kalman filter may generally outperform (Fig. 5, panel (d)) or underperform the linear decode (Fig. 6, panel (c)), or there may be generally higher error in data set L2006A than L2006B. In all cases though, there is very little trend that can be seen in the data suggesting that one firing rate method consistently outperforms any other. This finding is perhaps surprising, given the variety of firing rate estimates that are produced from the same spike trains using these different methods, as seen in Fig. 2.

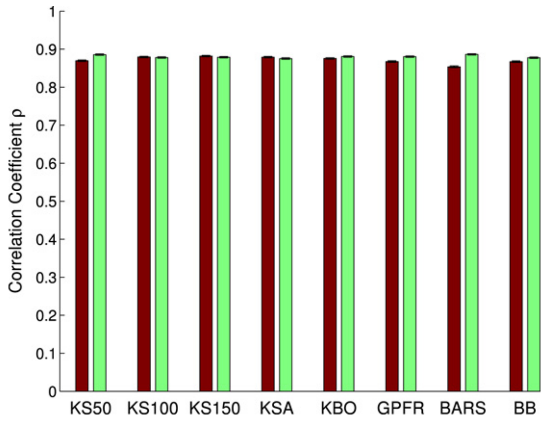
We further note that, from our testing, this similarity in decode performance remains if different numbers of neurons are used, or if different lengths of trials are considered, or if different temporal lags are imposed between neural activity and physical behavior (as is often done in BMI studies), or if the firing rate data is considered at finer or coarser time intervals. In addition to these



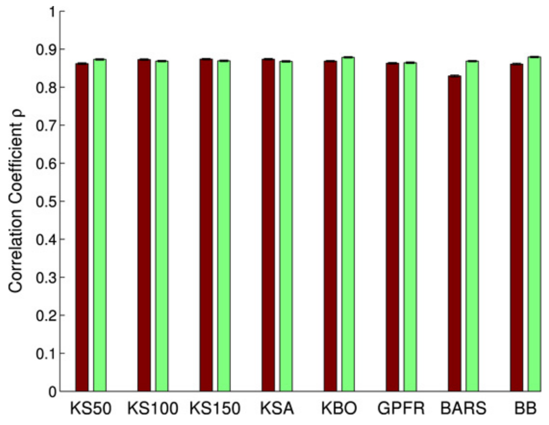
(a) Horizontal hand position, L2006A.



(b) Vertical hand position, L2006A.



(c) Horizontal hand position, L2006B.



(d) Vertical hand position, L2006B.

Fig. 6. The decoder performance of spike trains smoothed with different firing rate methods. Vertical axis is correlation coefficient with the true reach. In all panels, red bars are decode performance with a linear filter; green bars are performance numbers with a Kalman filter. Error bars (vanishingly small) indicate the 95% confidence interval on the estimate of the correlation coefficient (see Zar (1999)).

summary performance statistics, we note that, from our visual inspection of many decoded trials (*e.g.*, Fig. 4), all the firing rate estimators had the same performance in terms of how many decoded trajectories we described as "better" (*cf.* Fig. 4, panel (c)), "reasonable" (Fig. 4, panels (a) and (b)), and "failed" (Fig. 4, panel (d)). Thus, across all quantitative and qualitative analysis of the data that we have investigated, firing rate estimation offers little difference in terms of the quality of prosthetic decode. We discuss the implications of this seemingly general finding below.

5. Discussion and conclusions

Optimally inferring neural firing rates from spike trains is an unanswered research question, and many groups have addressed this interesting problem. In this paper, we reviewed some recent and some classical firing rate estimators. We discussed the theoretical motivation for each and discussed some potential advantages and disadvantages of competing methods. Firing rate estimation is a broad question that is applicable to neuroscientific and BMI applications, multiple and single trials, multiple and single neurons, and more. Each firing rate method should be considered specifically for its potential applications.

In this paper, after reviewing these methods, we investigated the relevance of firing rate estimation methods for an important BMI application: decoding individual arm movements from simultaneously recorded neural populations. We trained a monkey in a standard reaching paradigm (as described in Section 3.1 and in Fig. 3), and we used two standard decoding algorithms to estimate arm trajectories from neural activity. These algorithms – the linear decode and the Kalman filter (as described in Section 3.2) – accept as input neural firing rates over a population of neurons. Using the same neural spike trains, we inferred neural firing rates using eight different firing rate methods, and then we decoded arm trajectories using these firing rates.

Though the firing rates found by all eight methods appear quite different (see Fig. 2), the decoding test indicated that in fact firing rate estimation matters very little for this domain of prosthetic decode. We showed in Figs. 5 and 6 that RMSE and correlation coefficients of the decode are rather insensitive to the firing rate estimation method that is used to process the neural spike trains. Looking across two dimensions of decode (horizontal and vertical), two different data sets with different neural populations (L2006A and L2006B), and two different decoding algorithms (linear decode and Kalman filter), no discernable trend appears to indicate that one method (or one class of methods) is unambiguously better than any other. Thus, we believe the relevance of firing rate estimation, as it pertains to neural prosthetic decode, is in doubt.

Naturally the question then arises: how do such different firing rates (as in Fig. 2) produce such similar decode performance (as in Figs. 5 and 6)? We consider three possible explanations: (1) the decoding algorithms themselves are insensitive to differences in firing rate estimation; (2) the firing rate methods all have particular strengths and weaknesses but result in essentially the same signal-to-noise ratio (SNR); and (3) the ability to decode depends much more on factors other than firing rate estimation, and thus the firing rate estimator is not meaningful.

To the first point, if the decoding algorithms themselves smoothed over any differences in the firing rate estimations, we might expect very similar decoded trajectories. However, the different firing rate methods do in fact produce quite different decoded trajectories. Fig. 4 demonstrates this variety in four sample cases. Across the linear decode and the Kalman filter, we find that the RMSE between different decoded trajectories (estimates derived from different firing rate methods) is typically 30%–50% of the error with the true reach, and thus these estimates are indeed consistently different. Further, if the decoder was insensitive to firing rate estimates, we should be able to remove the firing rate estimator entirely (simply binning firing rate counts) without change to the decode quality. We tried a simple binning scheme, using both 50 ms (as used in, *e.g.*, Wu et al. (2002)) and 100 ms bins (as used in, *e.g.* Chestek et al. (2007)). Interestingly, we find this simplifying step can change error meaningfully, increasing error considerably in the case of the linear decode (but less so with the Kalman filter; indeed, sometimes binning reduces error in the Kalman filter case). Thus, temporal smoothing of firing rates seems valuable, and the method of smoothing influences the decoded arm trajectory meaningfully. Based on these findings, we see that the decode algorithms themselves are indeed sensitive to differences in firing rate estimation.

To the second possibility, each firing rate method does seem to make particular tradeoffs between signal and noise. In the simplest case, a low bandwidth kernel smoother (such as KS150) will produce a slowly varying firing rate with a similar time course to the arm activity. However, it also eliminates steep changes in firing rate, which likely provide a meaningful signal to the timecourse of arm movement. Fig. 2, panel (b), shows this possibility: while KS50, GPFR, and others pick up the sharp "ON" transient in the firing rate, they also pick up noise in the subsequent high firing rate. In contrast, KS150 smooths out both the noise and the step change in firing rate. Thus, it is likely that these firing rate methods and others each represent some balance between capturing or removing both signal and noise. Loosely, while each method may result in very different firing rate estimates, the SNR of each estimate may in fact be similar.

To the third possibility, it seems quite likely that the biggest effect on decode performance comes from aspects of the decoding system that are not neural firing rates. For example, the addition or removal of one or more very informative neurons to the neural population does often alter these performance numbers considerably (we found this effect in our additional testing), thus suggesting that recording technology (such as Wise et al. (2004)) may be more critical. Furthermore, the consideration of neural plan activity (before the movement begins) has been found to significantly reduce decoding error (Kemere et al., 2004; Yu et al., 2007). These are two examples of a host of avenues that may be significant determinants of prosthetic performance. Other avenues, as previously noted, may include prosthetic decode algorithms in general (Brockwell et al., 2004; Brown et al., 1998; Georgopoulos et al., 1986; Wu et al., 2004, 2006), the prosthetic interface itself (Cunningham, Yu, Gilja et al., 2008; Schwartz, 2004; Velliste et al., 2008), and multiple signal modalities (*e.g.*, EEG, ECoG, LFP, and spiking activity) (Mehring et al., 2003). Even if these other factors are much larger determinants of performance than firing rate estimation, one might still hope to see that certain firing rate estimators performed unambiguously better (albeit only slightly better) than others. Looking across decoders and data sets and error metrics, such a claim can not be made.

Despite the questionable relevance of firing rate estimation to the problem of neural prosthetic decode, we want to strongly clarify that we do not call into question the validity of firing rate estimation in general. Many of the excellent papers in this domain (several of which were reviewed in this study) may have important applications in neuroscientific studies or some other domain of neural signal processing. For example, these methods may be especially important in settings, unlike arm movements, where experimental conditions can be closely copied on each trial, producing similar neural responses (*e.g.*, visual stimuli shown to *in vitro* retinal neurons (Pillow, Paninski, Uzzell, Simoncelli, & Chichilinsky, 2005)).

Neural prostheses and BMI have received much attention in the last decade. As a result, many researchers from many fields have studied ways to improve our ability to understand and decode neural signals. Despite this preponderance of methodological development, very few systematic comparisons have been made in real experimental settings. The gold standard for such a comparison is perhaps online (closed loop) clinical trials, where the BMI user may engage learning, neural plasticity, and a host of other feedback mechanisms. Prior to that step, offline comparisons should be made on a variety of experimentally gathered data, and these comparisons can be made between all aspects of neural prosthetic systems. It behooves the field to review and compare available methods at each step in the BMI signal path. In this paper, we have made a first effort in that direction by reviewing and comparing different firing rate estimation methods. Prosthetic decoding algorithms may be another attractive target for such a review and comparison. The field should greatly benefit from such studies, both in terms of benchmarking the past and helping to set research agendas for the future.

Acknowledgments

We thank Cindy Chestek and Rachel Kalmar for valuable technical discussions. We thank Mackenzie Risch for veterinary care, Drew Haven for technical support, and Sandy Eisensee for administrative assistance.

Grants

This work was supported by the Michael Flynn Stanford Graduate Fellowship (JPC), NDSEG Fellowship (VG), NSF Graduate Research Fellowship (VG), NIH-NINDS-CRCNS-R01, Christopher and Dana Reeve Foundation (SIR & KVS), and the following grants to KVS: Burroughs Wellcome Fund Career Award in the Biomedical Sciences, Stanford Center for Integrated Systems, NSF Center for Neuromorphic Systems Engineering at Caltech, Office of Naval Research, Sloan Foundation, and Whitaker Foundation.

Disclosures

None.

References

Barbieri, R., Quirk, M., Frank, L., Wilson, M., & Brown, E. (2001). Construction and analysis of non-Poisson stimulus-response models of neural spiking activity. *Journal of Neuroscience Methods*, *105*, 25–37.

Behseta, S., & Kass, R. (2005). Testing equality of two functions using BARS. *Statistics in Medicine*, *24*, 3523–3534.

Bishop, C. (2006). *Pattern recognition and machine learning*. New York: Springer.

Borst, A., & Theunissen, F. E. (1999). Information theory and neural coding. *Nature Neuroscience*, *2*.

Bowman, A. W. (1984). An alternative method of cross-validation for the smoothing of density estimates. *Biometrika*, *71*, 353–360.

Brockwell, A., Rojas, A., & Kass, R. (2004). Recursive Bayesian decoding of motor cortical signals by particle filtering. *Journal of Neurophysiology*, *91*(4), 1899–1907.

Brown, E., Barbieri, R., Ventura, V., Kass, R., & Frank, L. (2002). The time-rescaling theorem and its application to neural spike train data analysis. *Neural Computation*.

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

Brown, E., Kass, R., & Mitra, P. (2004). Multiple neural spike train data analysis: State-of-the-art and future challenges. *Nature Neuroscience*, *7*, 456–461.

Carmena, J., Lebedev, M., Crist, R., O'Doherty, J., Santucci, D., Dimitrov, D., et al. (2003). Learning to control a brain-machine interface for reaching and grasping by primates. *PLoS Biology*, *1*, 193–208.

Carmena, J. M., Lebedev, M. A., Henriquez, C. S., & Nicolelis, M. A. (2005). Stable ensemble performance with single-neuron variability during reaching movements in primates. *Journal of Neuroscience*, *25*(46), 10712–10716.

Chapin, J. (2004). Using multi-neuron population recordings for neural prosthetics. *Nature Neuroscience*, *7*, 452–455.

Chestek, C., Batista, A., Santhanam, G., Yu, B., Afshar, A., Cunningham, J., et al. (2007). Single-neuron stability during repeated reaching in macaque premotor cortex. *Journal of Neuroscience*, *27*, 10742–10750.

Churchland, M. M., Yu, B. M., Ryu, S. I., Santhanam, G., & Shenoy, K. V. (2006). Neural variability in premotor cortex provides a signature of motor preparation. *Journal of Neuroscience*, *26*(14), 3697–3712.

Churchland, M. M., Yu, B. M., Sahani, M., & Shenoy, K. V. (2007). Techniques for extracting single-trial activity patterns from large-scale neural recordings. *Current Opinion in Neurobiology*, *17*.

Coleman, T. P., & Sarma, S. (2007). A computationally efficient method for modeling neural spiking activity with point processes nonparametrically. In *46th IEEE conf. on decision and control* (pp. 5800–5805).

Cunningham, J. P., Sahani, M., & Shenoy, K. V. (2008). Fast Gaussian process methods for point process intensity estimation. In *Proceedings of the 25th international conference on machine learning*.

Cunningham, J. P., Yu, B. M., Gilja, V., Ryu, S. I., & Shenoy, K. V. (2008). Toward optimal target placement for neural prosthetic devices. *Journal of Neurophysiology*, *100*(6), 3445–3457.

Cunningham, J. P., Yu, B. M., Shenoy, K. V., & Sahani, M. (2008). Inferring neural firing rates from spike trains using Gaussian processes. In J. Platt, D. Koller, Y. Singer, & Roweis S. (Eds.), *Advances in NIPS*: *Vol. 20*. Cambridge, MA: MIT Press.

Czanner, G., Eden, U. T., Wirth, S., Yanike, M., Suzuki, W., & Brown, E. N. (2008). Analysis of between-trial and within-trial neural spiking dynamics. *Journal of Neurophysiology*, *99*, 2672–2693.

Daley, D., & Vere-Jones, D. (2002). *An introduction to the theory of point processes*. New York: Springer.

DiMatteo, I., Genovese, C., & Kass, R. (2001). Bayesian curve-fitting with free-knot splines. *Biometrika*, *88*, 1055–1071.

Eden, U., Frank, L., Barbieri, R., Solo, V., & Brown, E. N. (2004). Dynamic analysis of neural encoding by point process adaptive filtering. *Neural Computation*, *16*, 971–998.

Endres, D., Oram, M., Schindelin, J., & Foldiak, P. (2008). Bayesian binning beats approximate alternatives: Estimating peri-stimulus time histograms. *Advances in NIPS*, *Vol. 20*.

Georgopoulos, A., Schwartz, A., & Kettner, R. (1986). Neuronal population coding of movement direction. *Science*, *233*, 1416–1419.

Hatsopoulos, N., Joshi, J., & O'Leary, J. G. (2004). Decoding continuous and discrete motor behaviors using motor and premotor cortical ensembles. *Journal of Neurophysiology*, *92*, 1165–1174.

Hochberg, L. R., Serruya, M. D., Friehs, G. M., Mukand, J. A., Saleh, M., Caplan, A. H., et al. (2006). Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature*, *442*, 164–171.

Johnson, D. (1996). Point process models of single-neuron discharges. *Journal of Computational Neuroscience*, *3*, 275–299.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*, 35–45.

Kass, R., & Ventura, V. (2003). A spike-train probability model. *Neural Computation*, *14*, 5–15.

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

Kaufman, C., Ventura, V., & Kass, R. (2005). Spline-based non-parametric regression for periodic functions and its application to directional tuning of neurons. *Statistics in Medicine*, *24*, 2255–2265.

Kemere, C., Shenoy, K., & Meng, T. (2004). Model-based neural decoding of reaching movements: A maximum likelihood approach [Special issue]. In *Brain–machine interfaces IEEE Transactions on Biomedical Engineering*, *51*, 925–932.

Koyama, S., & Kass, R. E. (2008). Spike train probability models for stimulus-drive leaky integrate-and-fire neurons. *Neural Computation*, *20*.

Lebedev, M. A., & Nicolelis, M. A. L. (2006). Brain-machine interfaces: Past, present, and future. *Trends in Neuroscience*, *29*.

Lewicki, M. S. (1998). A review of methods for spike sorting: The detection and classification of neural action potentials. *Network: Computation in Neural Systems*, *9*.

Mehring, C., Rickert, J., Vaadia, E., Deoliveira, S. C., Aertsen, A., & Rotter, S. (2003). Inference of hand movements from local field potentials in monkey motor cortex. *Nature Neuroscience*, *6*.

Miura, K., Tsubo, Y., Okada, M., & Fukai, T. (2007). Balanced excitatory and inhibitory inputs to cortical neurons decouple firing irregularity from rate modulations. *Journal of Neuroscience*, *27*, 13802–13812.

Nawrot, M., Aertsen, A., & Rotter, S. (1999). Single-trial estimation of neuronal firing rates: From single-neuron spike trains to population activity. *Journal of Neuroscience Methods*, *94*, 81–92.

Nirenberg, S., Carcieri, S. M., Jacobs, A. L., & Latham, P. E. (2001). Retinal ganglion cells act largely as independent encoders. *Nature*, *411*.

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

Paninski, L., Pillow, J. W., & Simoncelli, E. P. (2004). Maximum likelihood estimation of a stochastic integrate-and-fire neural model. *Neural Computation*, *16*, 2533–2561.

Papoulis, A., & Pillai, S. U. (2002). *Probability, random variables, and stochastic processes*. Boston: McGraw Hill.

Pillow, J. W., Paninski, L., Uzzell, V. J., Simoncelli, E. P., & Chichilinsky, E. J. (2005). Prediction and decoding of retinal ganglion cell responses with a probabilistic spiking model. *Journal of Neuroscience*, *25*.

Pillow, J. W., Shlens, J., Paninski, L., Sher, A., Litke, A., Chichilnisky, E. J., et al. (2008). Spatio-temporal correlations and visual signalling in a complete neural population. *Nature*.

Rasmussen, C., & Williams, C. (2006). *Gaussian processes for machine learning*. Cambridge: MIT Press.

Richmond, B., Optican, L., & Spitzer, H. (1990). Temporal econding of two-dimensional patterns by single units in primate primary visual cortex. i. stimulus-response relations. *Journal of Neurophysiology*, *64*(2).

Sahani, M. (1999). Latent variable models for neural data analysis. *Ph.D. thesis*, California Institute of Technology.

Santhanam, G., Ryu, S. I., Yu, B. M., Afshar, A., & Shenoy, K. V. (2006). A high-performance brain-computer interface. *Nature*, *442*, 195–198.

Santhanam, G., Sahani, M., Ryu, S. I., & Shenoy, K. V. (2004). An extensible infrastructure for fully automated spike sorting during online experiments. In *Proc. of the IEEE EMBS* (pp. 4380–4384).

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

Serruya, M., Hatsopoulos, N., Paninski, L., Fellows, M., & Donoghue, J. (2002). Instant neural control of a movement signal. *Nature*, *416*, 141–142.

Shimazaki, H., & Shinomoto, S. (2007a). Kernel width optimization in the spike-rate estimation. In *Neural Coding 2007*, Montevideo, Uruguay.

Shimazaki, H., & Shinomoto, S. (2007b). A method for selecting the bin size of a time histogram. *Neural Computation*, *19*(6), 1503–1527.

Shlens, J., Field, G., Gauthier, J., Grivich, M., Petrusca, D., Sher, A., et al. (2006). The structure of multi-neuron firing rate patters in primate retina. *Journal of Neuroscience*, *26*.

Srinivasan, L., Eden, U. T., Mitter, S. J., & Brown, E. N. (2007). General purpose filter design for neural prosthetic systems. *Journal of Neurophysics*, *98*, 2456–2475.

Stark, E., Drori, R., & Abeles, M. (2006). Partial cross-correlation analysis resolves ambiguity in the encoding of multiple movement features. *Journal of Neurophysiology*, *95*, 1966–1975.

Taylor, D., Tillery, S. H., & Schwartz, A. (2002). Direct cortical control of 3D neuroprosthetic devices. *Science*, *296*, 1829–1832.

Truccolo, W., Eden, U., Fellows, M., Donoghue, J., & Brown, E. (2004). A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology*, *93*, 1074–1089.

Velliste, M., Perel, S., Spalding, M. C., Whitford, A. S., & Schwartz, A. B. (2008). Cortical control of a prosthetic arm for self-feeding. *Nature*, *453*.

Ventura, V., Carta, R., Kass, R., Gettner, S., & Olson, C. (2002). Statistical analysis of temporal evolution in single-neuron firing rates. *Biostatistics*, *3*(1), 1–20.

Wise, K., Anderson, D., Hetke, J., Kipke, D., & Najafi, K. (2004). Wireless implantable microsystems: High-density electronic interfaces to the nervous system. *Proceedings of the IEEE*, *92*, 76–97.

Wu, W., Black, M., Gao, Y., Bienenstock, E., Serruya, M., & Donoghue, J. (2002). Inferring hand motion from multi-cell recordings in motor cortex using a Kalman filter. In *SAB'02-workshop on motor control in humans and robots: On the interplay of real brains and artificial devices* (pp. 66–73).

Wu, W., Black, M., Mumford, D., Gao, Y., Bienenstock, E., & Donoghue, J. (2004). Modeling and decoding motor cortical activity using a switching Kalman filter. *IEEE Transactions on TBME*, *51*(6), 933–942.

Wu, W., Gao, Y., Bienenstock, E., Donoghue, J., & Black, M. (2006). Bayesian population decoding of motor cortical activity using a Kalman filter. *Neural Computation*, *18*(1), 80–118.

Yu, B., Afshar, A., Santhanam, G., Ryu, S., Shenoy, K., & Sahani, M. (2005). Extracting dynamical structure embedded in neural activity. *Advances in NIPS*, *17*.

Yu, B. M., Cunningham, J. P., Santhanam, G., Ryu, S. I., Shenoy, K. V., & Sahani, M. (2009). Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. In *Advances in NIPS*: *Vol. 21*. Cambridge, MA: MIT Press.

Yu, B. M., Kemere, C., Santhanam, G., Afshar, A., Ryu, S. I., Meng, T. H., et al. (2007). Mixture of trajectory models for neural decoding of goal-directed movements. *Journal of Neurophysiology*, *97*, 3763–3780.

Zar, J. (1999). *Biostatistical analysis*. New Jersey: Prentice Hall.

Zumsteg, Z. S., Kemere, C., O'Driscoll, S., Santhanam, G., Ahmed, R. E., Shenoy, K. V., et al. (2005). Power feasibility of implantable digital spike sorting circuits for neural prosthetic systems. *IEEE TNSRE*, *13*, 272–279.