Neural Decoding of Movements: From Linear to Nonlinear Trajectory Models

Byron M. Yu[1,2], John P. Cunningham[1],
Krishna V. Shenoy[1], and Maneesh Sahani[2]

[1] Dept. of Electrical Engineering and Neurosciences Program,
Stanford University, Stanford, CA, USA
[2] Gatsby Computational Neuroscience Unit, UCL, London, UK
{byronyu,jcunnin,shenoy}@stanford.edu
maneesh@gatsby.ucl.ac.uk

Abstract. To date, the neural decoding of time-evolving physical state – for example, the path of a foraging rat or arm movements – has been largely carried out using linear trajectory models, primarily due to their computational efficiency. The possibility of better capturing the statistics of the movements using *nonlinear* trajectory models, thereby yielding more accurate decoded trajectories, is enticing. However, nonlinear decoding usually carries a higher computational cost, which is an important consideration in real-time settings. In this paper, we present techniques for nonlinear decoding employing modal Gaussian approximations, expectatation propagation, and Gaussian quadrature. We compare their decoding accuracy versus computation time tradeoffs based on high-dimensional simulated neural spike counts.

Key words: Nonlinear dynamical models, nonlinear state estimation, neural decoding, neural prosthetics, expectation-propagation, Gaussian quadrature

1 Introduction

We consider the problem of decoding time-evolving physical state from neural spike trains. Examples include decoding the path of a foraging rat from hippocampal neurons [1, 2] and decoding the arm trajectory from motor cortical neurons [3–8]. Advances in this area have enabled the development of neural prosthetic devices, which seek to allow disabled patients to regain motor function through the use of prosthetic limbs, or computer cursors, that are controlled by neural activity [9–15].

Several of these prosthetic decoders, including population vectors [11] and linear filters [10, 12, 15], linearly map the observed neural activity to the estimate of physical state. Although these direct linear mappings are effective, recursive Bayesian decoders have been shown to provide more accurate trajectory estimates [1, 6, 7, 16]. In addition, recursive Bayesian decoders provide confidence regions on the trajectory estimates and allow for nonlinear relationships between

the neural activity and the physical state variables. Recursive Bayesian decoders are based on the specification of a probabilistic model comprising 1) a *trajectory model*, which describes how the physical state variables change from one time step to the next, and 2) an *observation model*, which describes how the observed neural activity relates to the time-evolving physical state.

The function of the trajectory model is to build into the decoder prior knowledge about the form of the trajectories. In the case of decoding arm movements, the trajectory model may reflect 1) the hard, physical constraints of the limb (for example, the elbow cannot bend backward), 2) the soft, control constraints imposed by neural mechanisms (for example, the arm is more likely to move smoothly than in a jerky motion), and 3) the physical surroundings of the person and his/her objectives in that environment. The degree to which the trajectory model captures the statistics of the actual movements directly affects the accuracy with which trajectories can be decoded from neural data [8].

The most commonly-used trajectory models assume linear dynamics perturbed by Gaussian noise, which we refer to collectively as linear-Gaussian models. The family of linear-Gaussian models includes the random walk model [1, 2, 6], those with a constant [8] or time-varying [17, 18] forcing term, those without a forcing term [7, 16], those with a time-varying state transition matrix [19], and those with higher-order Markov dependencies [20]. Linear-Gaussian models have been successfully applied to decoding the path of a foraging rat [1, 2], as well as arm trajectories in ellipse-tracing [6], pursuit-tracking [7, 20, 16], "pinball" [7, 16], and center-out reach [8] tasks.

Linear-Gaussian models are widely used primarily due to their computational efficiency, which is an important consideration for real-time decoding applications. However, for particular types of movements, the family of linear-Gaussian models may be too restrictive and unable to capture salient properties of the observed movements [8]. We recently proposed a general approach to constructing trajectory models that can exhibit rather complex dynamical behaviors and whose decoder can be implemented to have the same running time (using a parallel implementation) as simpler trajectory models [8]. In particular, we demonstrated that a probabilistic mixture of linear-Gaussian trajectory models, each accurate within a limited regime of movement, can capture the salient properties of goal-directed reaches to multiple targets. This mixture model, which yielded more accurate decoded trajectories than a single linear-Gaussian model, can be viewed as a discrete approximation to a single, unified trajectory model with nonlinear dynamics.

An alternate approach is to decode using this single, unified nonlinear trajectory model without discretization. This makes the decoding problem more difficult since nonlinear transformations of parametric distributions are typically no longer easily parametrized. State estimation in nonlinear dynamical systems is a field of active research that has made substantial progress in recent years, including the application of numerical quadrature techniques to dynamical systems [21–23], the development of expectation-propagation (EP) [24] and its application to dynamical systems [25–28], and the improvement in the com-

putational efficiency of Monte Carlo techniques (e.g., [29–31]). However, these techniques have not been rigorously tested and compared in the context of neural decoding, which typically involves observations that are high-dimensional vectors of non-negative integers. In particular, the tradeoff between decoding accuracy and computational cost among different neural decoding algorithms has not been studied in detail. Knowing the accuracy-computational cost tradeoff is important for real-time applications, where one may need to select the most accurate algorithm given a computational budget or the least computationally intensive algorithm given a minimal acceptable decoding accuracy. This paper takes a step in this direction by comparing three particular deterministic Gaussian approximations. In Section 2, we first introduce the nonlinear dynamical model for neural spike counts and the decoding problem. Sections 3 and 4 detail the three deterministic Gaussian approximations that we focus on in this report: global Laplace, Gaussian quadrature-EP (GQ-EP), and Laplace propagation (LP). Finally, in Section 5, we compare the decoding accuracy versus computational cost of these three techniques.

2 Nonlinear dynamical model and neural decoding

In this report, we consider nonlinear dynamical models for neural spike counts of the following form:

$$\mathbf{x}_t \mid \mathbf{x}_{t-1} \sim \mathcal{N}\left(\mathbf{f}\left(\mathbf{x}_{t-1}\right), Q\right) \tag{1a}$$

$$y_t^i \mid \mathbf{x}_t \sim \text{Poisson}\left(\lambda_i\left(\mathbf{x}_t\right) \cdot \Delta\right), \tag{1b}$$

where $\mathbf{x}_t \in \mathbb{R}^{p \times 1}$ is a vector containing the physical state variables at time $t = 1, \ldots, T$, $y_t^i \in \{0, 1, 2, \ldots\}$ is the corresponding observed spike count for neuron $i = 1, \ldots, q$ taken in a time bin of width Δ, and $Q \in \mathbb{R}^{p \times p}$ is a covariance matrix. The functions $\mathbf{f} : \mathbb{R}^{p \times 1} \to \mathbb{R}^{p \times 1}$ and $\lambda_i : \mathbb{R}^{p \times 1} \to \mathbb{R}_+$ are, in general, nonlinear. The initial state $\mathbf{x}_1$ is Gaussian-distributed. For notational compactness, the spike counts for all q simultaneously-recorded neurons are assembled into a $q \times 1$ vector $\mathbf{y}_t$, whose ith element is y_t^i. Note that the observations are discrete-valued and that, typically, $q \gg p$. Equations (1a) and (1b) are referred to as the trajectory and observation models, respectively.

The task of neural decoding involves finding, at each timepoint t, the likely physical states $\mathbf{x}_t$ given the neural activity observed up to that time $\{\mathbf{y}\}_1^t$. In other words, we seek to compute the *filtered state posterior* $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right)$ at each t. We previously showed how to estimate the filtered state posterior when $\mathbf{f}$ is a linear function [8]. Here, we consider how to compute $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right)$ when $\mathbf{f}$ is nonlinear.

The extended Kalman filter (EKF) is a commonly-used technique for nonlinear state estimation. Unfortunately, it cannot be directly applied to the current problem because the observation noise in (1b) is not additive Gaussian. Possible alternatives are the unscented Kalman filter (UKF) [21, 22] and the closely-related quadrature Kalman filter (QKF) [23], both of which employ quadrature

techniques to approximate Gaussian integrals that are analytically intractable. While the UKF has been shown to outperform the EKF [21, 22], the UKF requires making Gaussian approximations in the observation space. This property of the UKF is undesirable from the standpoint of the current problem because the observed spike counts are typically 0 or 1 (due to the use of relatively short binwidths Δ) and, therefore, distinctly non-Gaussian. As a result, the UKF yielded substantially lower decoding accuracy than the techniques presented in Sections 3 and 4 [28], which make Gaussian approximations only in the state space. While we have not yet tested the QKF, the number of quadrature points required grows geometrically with $p+q$, which quickly becomes impractical even for moderate values of p and q. Thus, we will no longer consider the UKF and QKF in the remainder of this paper.

The decoding techniques described in Sections 3 and 4 naturally yield the *smoothed state posterior* $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$, rather than the filtered state posterior $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^t\right)$. Thus, we will focus on the smoothed state posterior in this work. However, the filtered state posterior at time t can be easily obtained by smoothing using only observations from timepoints $1, \ldots, t$.

3 Global Laplace

The idea is to estimate the joint state posterior across the entire sequence (i.e., the *global* state posterior) as a Gaussian matched to the location and curvature of a mode of $P\left(\{\mathbf{x}\}_1^T \mid \{\mathbf{y}\}_1^T\right)$, as in Laplace's method [32]. The mode is defined as

$$\{\mathbf{x}^\star\}_1^T = \underset{\{\mathbf{x}\}_1^T}{\operatorname{argmax}} P\left(\{\mathbf{x}\}_1^T \mid \{\mathbf{y}\}_1^T\right) = \underset{\{\mathbf{x}\}_1^T}{\operatorname{argmax}} L\left(\{\mathbf{x}\}_1^T\right), \tag{2}$$

where

$$\begin{aligned} L\left(\{\mathbf{x}\}_1^T\right) &= \log P\left(\{\mathbf{x}\}_1^T, \{\mathbf{y}\}_1^T\right) \\ &= \log P\left(\mathbf{x}_1\right) + \sum_{t=2}^{T} \log P\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) + \sum_{t=1}^{T} \sum_{i=1}^{q} \log P\left(y_t^i \mid \mathbf{x}_t\right). \end{aligned} \tag{3}$$

Using the known distributions (1), the gradients of $L\left(\{\mathbf{x}\}_1^T\right)$ can be computed exactly and a local mode $\{\mathbf{x}^\star\}_1^T$ can be found by applying a gradient optimization technique. The global state posterior is then approximated as:

$$P\left(\{\mathbf{x}\}_1^T \mid \{\mathbf{y}\}_1^T\right) \approx \mathcal{N}\left(\{\mathbf{x}^\star\}_1^T, -\nabla^2 L\left(\{\mathbf{x}^\star\}_1^T\right)^{-1}\right). \tag{4}$$

4 Expectation Propagation

We briefly summarize here the application of EP [24] to dynamical models [25–28]. More details can be found in the cited references. The two primary distributions of interest here are the marginal $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$ and pairwise joint

$P\left(\mathbf{x}_{t-1}, \mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$ state posteriors. These distributions can be expressed in terms of forward α_t and backward β_t messages as follows

$$P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right) = \frac{1}{P\left(\{\mathbf{y}\}_1^T\right)} \alpha_t\left(\mathbf{x}_t\right) \beta_t\left(\mathbf{x}_t\right) \tag{5}$$

$$P\left(\mathbf{x}_{t-1}, \mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right) = \frac{\alpha_{t-1}\left(\mathbf{x}_{t-1}\right) P\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) P\left(\mathbf{y}_t \mid \mathbf{x}_t\right) \beta_t\left(\mathbf{x}_t\right)}{P\left(\{\mathbf{y}\}_1^T\right)}, \tag{6}$$

where $\alpha_t\left(\mathbf{x}_t\right) = P\left(\mathbf{x}_t, \{\mathbf{y}\}_1^t\right)$ and $\beta_t\left(\mathbf{x}_t\right) = P\left(\{\mathbf{y}\}_{t+1}^T \mid \mathbf{x}_t\right)$. The messages α_t and β_t are typically approximated by an exponential family density; in this paper, we use an unnormalized Gaussian. These approximate messages are iteratively updated by matching the expected sufficient statistics[3] of the marginal posterior (5) with those of the pairwise joint posterior (6). The updates are usually performed sequentially via multiple forward-backward passes. During the forward pass, the α_t are updated while the β_t remain fixed:

$$P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right) = \int \frac{\alpha_{t-1}\left(\mathbf{x}_{t-1}\right) P\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) P\left(\mathbf{y}_t \mid \mathbf{x}_t\right) \beta_t\left(\mathbf{x}_t\right)}{P\left(\{\mathbf{y}\}_1^T\right)} d\mathbf{x}_{t-1} \tag{7}$$

$$\approx \int \hat{P}\left(\mathbf{x}_{t-1}, \mathbf{x}_t\right) d\mathbf{x}_{t-1} \tag{8}$$

$$\alpha_t\left(\mathbf{x}_t\right) \propto \int \hat{P}\left(\mathbf{x}_t, \mathbf{x}_{t-1}\right) d\mathbf{x}_{t-1} \Big/ \beta_t\left(\mathbf{x}_t\right), \tag{9}$$

where $\hat{P}\left(\mathbf{x}_{t-1}, \mathbf{x}_t\right)$ is an exponential family distribution whose expected sufficient statistics are matched to those of $P\left(\mathbf{x}_{t-1}, \mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$. In this paper, $\hat{P}\left(\mathbf{x}_{t-1}, \mathbf{x}_t\right)$ is assumed to be Gaussian. The backward pass proceeds similarly, where the β_t are updated while the α_t remain fixed. The decoded trajectory is obtained by combining the messages α_t and β_t, as shown in (5), after completing the forward-backward passes. In Section 5, we investigate the accuracy-computational cost tradeoff of using different numbers of forward-backward iterations.

Although the expected sufficient statistics (or moments) of $P\left(\mathbf{x}_{t-1}, \mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$ cannot typically be computed analytically for the nonlinear dynamical model (1), they can be approximated using Gaussian quadrature [26, 28]. This EP-based decoder is referred to as *GQ-EP*. By applying the ideas of Laplace propagation (LP) [33], a closely-related decoder has been developed that uses a modal Gaussian approximation of $P\left(\mathbf{x}_{t-1}, \mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$ rather than matching moments [27, 28]. This technique, which uses the same message-passing scheme as GQ-EP, is referred to here as *LP*.

In practice, it is possible to encounter invalid message updates. For example, if the variance of $\mathbf{x}_t$ in the numerator is larger than that in the denominator in (9) due to approximation error in the choice of $\hat{P}$, the update rule would assign $\alpha_t\left(\mathbf{x}_t\right)$ a negative variance. A way around this problem is to simply skip that message update and hope that the update is no longer invalid during the next

[3] If the approximating distributions are assumed to be Gaussian, this is equivalent to matching the first two moments.

forward-backward iteration [34]. An alternative is to set $\beta_t(\mathbf{x}_t) = 1$ in (7) and (9), which guarantees a valid update for $\alpha_t(\mathbf{x}_t)$. This is referred to as the *one-sided update* and its implications for decoding accuracy and computation time are considered in Section 5.

5 Results

We evaluated decoding accuracy versus computational cost of the techniques described in Sections 3 and 4. These performance comparisons were based on the model (1), where

$$\mathbf{f}(\mathbf{x}) = (1-k)\,\mathbf{x} + k \cdot W \cdot \mathrm{erf}(\mathbf{x}) \tag{10}$$

$$\lambda_i(\mathbf{x}) = \log\left(1 + e^{\mathbf{c}_i'\mathbf{x}+d_i}\right) \tag{11}$$

with parameters $W \in \mathbb{R}^{p\times p}$, $\mathbf{c}_i \in \mathbb{R}^{p\times 1}$, and $d_i \in \mathbb{R}$. The error function (erf) in (10) acts element-by-element on its argument. We have chosen the dynamics (10) of a fully-connected recurrent network due to its nonlinear nature; we make no claims in this work about its suitability for particular decoding applications, such as for rat paths or arm trajectories. Because recurrent networks are often used to directly model neural activity, it is important to emphasize that $\mathbf{x}$ is a vector of physical state variables to be decoded, not a vector of neural activity.

We generated 50 state trajectories, each with 50 time points, and corresponding spike counts from the model (1), where the model parameters were randomly chosen within a range that provided biologically realistic spike counts (typically, 0 or 1 spike in each bin). The time constant $k \in \mathbb{R}$ was set to 0.1. To understand how these algorithms scale with different numbers of physical state variables and observed neurons, we considered all pairings (p, q), where $p \in \{3, 10\}$ and $q \in \{20, 100, 500\}$. For each pairing, we repeated the above procedure three times.

For the global Laplace decoder, the modal trajectory was found using Polack-Ribière conjugate gradients with quadratic/cubic line searches and Wolfe-Powell stopping criteria (`minimize.m` by Carl Rasmussen, available at `http://www.kyb.tuebingen.mpg.de/bs/people/carl/code/minimize/`).
To stabilize GQ-EP, we used a modal Gaussian proposal distribution and the custom precision 3 quadrature rule with non-negative quadrature weights, as described in [28]. For both GQ-EP and LP, `minimize.m` was used to find a mode of $P\left(\mathbf{x}_{t-1}, \mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$.

Fig. 1 illustrates the decoding accuracy versus computation time of the presented techniques. Decoding accuracy was measured by evaluating the marginal state posteriors $P\left(\mathbf{x}_t \mid \{\mathbf{y}\}_1^T\right)$ at the actual trajectory. The higher the log probability, the more accurate the decoder. Each panel corresponds to a different number of state variables and observed neurons. For GQ-EP (dotted line) and LP (solid line), we varied the number of forward-backward iterations between one and three; thus, there are three circles for each of these decoders. Across all panels, global Laplace required the least computation time and yielded state

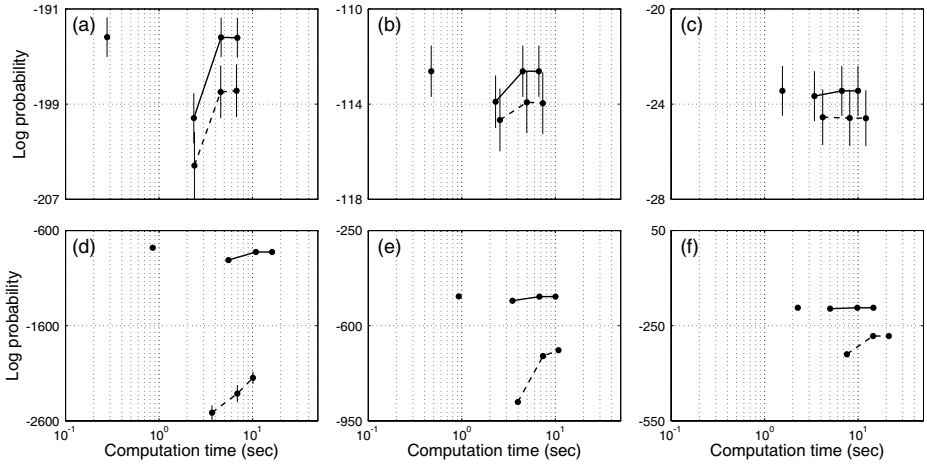

Fig. 1. Decoding accuracy versus computation time of global Laplace (no line), GQ-EP (dotted line), and LP (solid line). (a) $p = 3,\ q = 20$, (b) $p = 3,\ q = 100$, (c) $p = 3,\ q = 500$, (d) $p = 10,\ q = 20$, (e) $p = 10,\ q = 100$, (f) $p = 10,\ q = 500$. The circles and bars represent mean±SEM. Variability in computation time is not represented on the plots because they were negligible. The computation times were obtained using a 2.2-GHz AMD Athlon 64 processor with 2 GB RAM running MATLAB R14. Note that the scale of the vertical axes is not the same in each panel and that some error bars are so small that they can't be seen.

estimates as accurate as, or more accurate than, the other techniques. This is the key result of this report.

We also implemented a basic particle smoother [35], where the number of particles (500 to 1500) was chosen such that its computation time was on the same order as those shown in Fig. 1 (results not shown). Although this particle smoother yielded substantially lower decoding accuracy than global Laplace, GQ-EP, and LP, the three deterministic techniques should be compared to more recently-developed Monte Carlo techniques, as described in Section 6.

Fig. 1 shows that all three techniques have computation times that scale well with the number of state variables p and neurons q. In particular, the required computational time typically scales sub-linearly with increases in p and far sub-linearly with increases in q. As the q increases, the accuracies of the techniques become more similar (note that different panels have different vertical scales), and there is less advantage to performing multiple forward-backward iterations for GQ-EP and LP. The decoding accuracy and required computation time both typically increase with the number of iterations. In a few cases (e.g., GQ-EP in Fig. 1(b)), it is possible for the accuracy to decrease slightly when going from two to three iterations, presumably due to one-sided updates.

In theory, GQ-EP should require greater computation time than LP because it needs to perform the same modal Gaussian approximation, then use it as a proposal distribution for Gaussian quadrature. In practice, it is possible for LP

to be slower if it needs many one-sided updates (cf. Fig. 1(d)), since one-sided updates are used only when the usual update (9) fails. Furthermore, LP required greater computation time in Fig. 1(d) than in Fig. 1(e) due to the need for many more one-sided updates, despite having five times fewer neurons.

It was previously shown that $\{\mathbf{x}^\star\}_1^T$ is a local optimum of $P\left(\{\mathbf{x}\}_1^T \mid \{\mathbf{y}\}_1^T\right)$ (i.e., a solution of global Laplace) if and only if it is a fixed-point of LP [33]. Because the modal Gaussian approximation matches local curvature up to second order, it can also be shown that the estimated covariances using global Laplace and LP are equal at $\{\mathbf{x}^\star\}_1^T$ [33]. Empirically, we found both statements to be true if few one-sided updates were required for LP. Due to these connections between global Laplace and LP, the accuracy of LP after three forward-backward iterations was similar to that of global Laplace in all panels in Fig. 1. Although LP may have computational savings compared to global Laplace in certain applications [33], we found that global Laplace was substantially faster for the particular graph structure described by (1).

6 Conclusion

We have presented three deterministic techniques for nonlinear state estimation (global Laplace, GQ-EP, LP) and compared their decoding accuracy versus computation cost in the context of neural decoding, involving high-dimensional observations of non-negative integers. This work can be extended in the following directions. First, the deterministic techniques presented here should be compared to recently-developed Monte Carlo techniques that have yielded increased accuracy and/or reduced computational cost compared to the basic particle filter/smoother in applications other than neural decoding [29]. Examples include the Gaussian particle filter [31], sigma-point particle filter [30], and embedded hidden Markov model [36]. Second, we have compared these decoders based on one particular non-linear trajectory model (10). Other non-linear trajectory models (e.g., a model describing primate arm movements [37]) should be tested to see if the decoders have similar accuracy-computational cost tradeoffs as shown here.

Acknowledgments. This work was supported by NIH-NINDS-CRCNS-R01, NDSEG Fellowship, NSF Graduate Research Fellowship, Gatsby Charitable Foundation, Michael Flynn Stanford Graduate Fellowship, Christopher Reeve Paralysis Foundation, 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.

References

1. Brown, E.N., Frank, L.M., Tang, D., Quirk, M.C., Wilson, M.A.: A statistical paradigm for neural spike train decoding applied to position prediction from the

ensemble firing patterns of rat hippocampal place cells. J Neurosci **18**(18) (1998) 7411–7425
2. Zhang, K., Ginzburg, I., McNaughton, B.L., Sejnowski, T.J.: Interpreting neuronal population activity by reconstruction: Unified framework with application to hippocampal place cells. J Neurophysiol **79** (1998) 1017–1044
3. Wessberg, J., Stambaugh, C.R., Kralik, J.D., Beck, P.D., Laubach, M., Chapin, J.K., Kim, J., Biggs, J., Srinivasan, M.A., Nicolelis, M.A.L.: Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. Nature **408**(6810) (2000) 361–365
4. Schwartz, A.B., Taylor, D.M., Tillery, S.I.H.: Extraction algorithms for cortical control of arm prosthetics. Curr Opin Neurobiol **11** (2001) 701–707
5. Serruya, M., Hatsopoulos, N., Fellows, M., Paninski, L., Donoghue, J.: Robustness of neuroprosthetic decoding algorithms. Biol Cybern **88**(3) (2003) 219–228
6. Brockwell, A.E., Rojas, A.L., Kass, R.E.: Recursive Bayesian decoding of motor cortical signals by particle filtering. J Neurophysiol **91**(4) (2004) 1899–1907
7. Wu, W., Black, M.J., Mumford, D., Gao, Y., Bienenstock, E., Donoghue, J.P.: Modeling and decoding motor cortical activity using a switching Kalman filter. IEEE Trans Biomed Eng **51**(6) (2004) 933–942
8. Yu, B.M., Kemere, C., Santhanam, G., Afshar, A., Ryu, S.I., Meng, T.H., Sahani, M., Shenoy, K.V.: Mixture of trajectory models for neural decoding of goal-directed movements. J Neurophysiol **97** (2007) 3763–3780
9. Chapin, J.K., Moxon, K.A., Markowitz, R.S., Nicolelis, M.A.L.: Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. Nat Neurosci **2** (1999) 664–670
10. Serruya, M.D., Hatsopoulos, N.G., Paninski, L., Fellows, M.R., Donoghue, J.P.: Instant neural control of a movement signal. **416** (2002) 141–142
11. Taylor, D.M., Tillery, S.I.H., Schwartz, A.B.: Direct cortical control of 3D neuroprosthetic devices. Science **296** (2002) 1829–1832
12. Carmena, J.M., Lebedev, M.A., Crist, R.E., O'Doherty, J.E., Santucci, D.M., Dimitrov, D.F., Patil, P.G., Henriquez, C.S., Nicolelis, M.A.L.: Learning to control a brain-machine interface for reaching and grasping by primates. PLoS Biology **1**(2) (2003) 193–208
13. Musallam, S., Corneil, B.D., Greger, B., Scherberger, H., Andersen, R.A.: Cognitive control signals for neural prosthetics. Science **305** (2004) 258–262
14. Santhanam, G., Ryu, S.I., Yu, B.M., Afshar, A., Shenoy, K.V.: A high-performance brain-computer interface. Nature **442** (2006) 195–198
15. Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D., Donoghue, J.P.: Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature **442** (2006) 164–171
16. Wu, W., Gao, Y., Bienenstock, E., Donoghue, J.P., Black, M.J.: Bayesian population decoding of motor cortical activity using a Kalman filter. Neural Comput **18**(1) (2006) 80–118
17. Kemere, C., Meng, T.: Optimal estimation of feed-forward-controlled linear systems. In: Proc IEEE ICASSP. (2005) 353–356
18. Srinivasan, L., Eden, U.T., Willsky, A.S., Brown, E.N.: A state-space analysis for reconstruction of goal-directed movements using neural signals. Neural Comput **18**(10) (2006) 2465–2494
19. Srinivasan, L., Brown, E.N.: A state-space framework for movement control to dynamic goals through brain-driven interfaces. IEEE Trans Biomed Eng **54**(3) (2007) 526–535

20. Shoham, S., Paninski, L.M., Fellows, M.R., Hatsopoulos, N.G., Donoghue, J.P., Normann, R.A.: Statistical encoding model for a primary motor cortical brain-machine interface. IEEE Trans Biomed Eng **52**(7) (2005) 1313–1322
21. Wan, E., van der Merwe, R.: The unscented Kalman filter. In Haykin, S., ed.: Kalman Filtering and Neural Networks. Wiley Publishing (2001)
22. Julier, S., Uhlmann, J.: Unscented filtering and nonlinear estimation. Proceedings of the IEEE **92**(3) (2004) 401–422
23. Arasaratnam, I., Haykin, S., Elliott, R.: Discrete-time nonlinear filtering algorithms using Gauss-Hermite quadrature. Proceedings of the IEEE **95**(5) (2007) 953–977
24. Minka, T.: Expectation propagation for approximate Bayesian inference. In: Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI). (2001) 362–369
25. Heskes, T., Zoeter, O.: Expectation propagation for approximate inference in dynamic Bayesian networks. In Darwiche, A., Friedman, N., eds.: Proceedings UAI-2002. (2002) 216–223
26. Zoeter, O., Ypma, A., Heskes, T.: Improved unscented Kalman smoothing for stock volatility estimation. In Barros, A., Principe, J., Larsen, J., Adali, T., Douglas, S., eds.: Proceedings of the IEEE Workshop on Machine Learning for Signal Processing. (2004)
27. Ypma, A., Heskes, T.: Novel approximations for inference in nonlinear dynamical systems using expectation propagation. Neurocomputing **69** (2005) 85–99
28. Yu, B.M., Shenoy, K.V., Sahani, M.: Expectation propagation for inference in non-linear dynamical models with Poisson observations. In: Proc IEEE Nonlinear Statistical Signal Processing Workshop. (2006)
29. Doucet, A., de Freitas, N., Gordon, N., eds.: Sequential Monte Carlo Methods in Practice. Springer-Verlag (2001)
30. van der Merwe, R., Wan, E.: Sigma-point Kalman filters for probabilistic inference in dynamic state-space models. In: Proceedings of the Workshop on Advances in Machine Learning. (2003)
31. Kotecha, J.H., Djuric, P.M.: Gaussian particle filtering. IEEE Transactions on Signal Processing **51**(10) (2003) 2592–2601
32. MacKay, D.: Information Theory, Inference and Learning Algorithms. Cambridge University Press (2003)
33. Smola, A., Vishwanathan, V., Eskin, E.: Laplace propagation. In Thrun, S., Saul, L., Schölkopf, B., eds.: Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA (2004)
34. Minka, T., Lafferty, J.: Expectation-propagation for the generative aspect model. In: Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI). (2002) 352–359
35. Doucet, A., Godsill, S., Andrieu, C.: On sequential Monte Carlo sampling methods for Bayesian filtering. Statistics and Computing **10**(3) (2000) 197–208
36. Neal, R.M., Beal, M.J., Roweis, S.T.: Inferring state sequences for non-linear systems with embedded hidden Markov models. In Thrun, S., Saul, L., Schölkopf, B., eds.: Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA (2004)
37. Chan, S.S., Moran, D.W.: Computational model of a primate arm: from hand position to joint angles, joint torques and muscle forces. J Neural Eng **3** (2006) 327–337