31st Annual International Conference of the IEEE EMBS
Minneapolis, Minnesota, USA, September 2-6, 2009

Bayesian Auxiliary Particle Filters for Estimating Neural Tuning Parameters

John Mountney, Marc Sobel, Iyad Obeid

Abstract— **A common challenge in neural engineering is to track the dynamic parameters of neural tuning functions. This work introduces the application of Bayesian auxiliary particle filters for this purpose. Based on Monte-Carlo filtering, Bayesian auxiliary particle filters use adaptive methods to model the prior densities of the state parameters being tracked. The observations used are the neural firing times, modeled here as a Poisson process, and the biological driving signal. The Bayesian auxiliary particle filter was evaluated by simultaneously tracking the three parameters of a hippocampal place cell and compared to a stochastic state point process filter. It is shown that Bayesian auxiliary particle filters are substantially more accurate and robust than alternative methods of state parameter estimation. The effects of time-averaging on parameter estimation are also evaluated.**

I. INTRODUCTION

Changes that occur in the organization and function of the brain, especially in response to external factors, are referred to as neural plasticity. Neural plasticity can be the result of environmental changes, learning, normal experience or brain injury. It has been shown that thinking, learning, and acting actually change the brain's functional anatomy if not also its physical anatomy [1].

In order to decode and maintain an accurate estimate of the intended biological signal from a dynamic neuron, the evolution of a neuron's parameters must be approximated. Whenever there is a requirement to process signals that result from operation in an environment of unknown statistics, adaptive signal processing provides a means of tracking the temporal evolution of system parameters. Adaptive filters have been successfully applied in such diverse fields as digital communications, digital control, radar and biomedical engineering [2].

Traditionally, adaptive filtering algorithms update system parameter estimates by recursively combining the previously estimated system parameters with new information. This new information is processed to minimize a cost function, which is often a quadratic expression of the error, which are appropriate for continuous-valued functions. Since neural spike train data is a point process, approximating system parameters with a quadratic cost function in the absence of high firing rates is of limited applicability. Instead, it has been recently shown that the instantaneous log-likelihood of neural firing provides an appropriate cost function for adaptive filter algorithms for spike train model measurements, using instantaneous steepest descent and stochastic state models [3], [4].

An alternative to gradient-based approximations is the solution based on Bayesian estimation. Given the previous state x_{t-1} and the current observations $y_1, \ldots, y_t$, recursive Bayesian estimation uses a two-stage process to solve for the posterior distribution $p(x_t|y_1, \ldots, y_t)$. In the first stage, Bayes rule is used to update the posterior from the previous step. In the second stage, the current posterior is calculated using this updated posterior. However, it is often prohibitive to use this methodology since a closed form solution of the integrals required for a Bayesian recursive filter does not exist.

One specific class of Bayesian estimators is the particle filter. Particle filters are used to estimate the current state of a system x_t using numerical simulation methods that approximate the often difficult to solve integrals of the recursive Bayesian estimation problem [5]. If the integrals cannot be solved for analytically, Monte Carlo integration can be used to provide discrete support to represent the posterior probability as a set of randomly chosen weighted samples or *particles* from a *proposal* density that is chosen to approximate the posterior $p(x|y)$. The particle samples are assigned weights based on how likely they are to be drawn from the current state posterior.

A limitation of particle filters is that the proposal densities are static and hence unable to adapt to changing system dynamics. This leads to degeneracy, in which most particles yield no useful information, thereby negating the advantages of the particle filter approach. *Auxiliary particle filters*, introduced by Pitt and Shephard, address this limitation by constructing proposal densities that better correspond to the true posterior distribution [6]. An extended version of the auxiliary particle filter was developed by Liu and West, in which an additional hyper-parameter was added to better adapt the proposal to the posterior [7].

The goal of this work is to apply the extended auxiliary particle filter of Liu and West to track the state vector of a dynamic place cell. It is shown that this parameter tracking method is robust to periods of neural quiescence and is generally more stable than existing methods.

II. PARAMETER TRACKING THEORY

A number of parameterized models of the form $\lambda_t = f(u_t, x_t)$ have been proposed for describing a neuron's instantaneous likelihood of firing. λ_t represents the instantaneous firing rate, u_t represents the neuron's driving signal

I. Obeid (iobeid@temple.edu) and J. Mountney (jmm@temple.edu) are with the Department of Electrical & Computer Engineering, Temple University, Philadelphia, PA 19122

M. Sobel (marc.sobel@temple.edu) is with the Department of Statistics, Temple University, Philadelphia, PA 19122

All authors contributed equally to this work.

978-1-4244-3296-7/09/$25.00 ©2009 IEEE

(i.e. position, speed, reach angle, etc), and x_t represents the model's time-varying parameters (i.e. *state*). The goal of this work is to estimate the model state x_t given the driving signal u_t and the observed neural firings N_t. For notational simplicity, we combine the observed parameters into one variable, $y_t = (N_t, u_t)$, and introduce the abbreviation $y_{1:t}$ for the vector $(y_1, \ldots, y_t)$.

A. Iterative Parameter Tracking

Estimating the model states x_t from the observations $y_{1:t}$ requires the computation of $p(x_t|y_{1:t})$ using Bayes rule:

$$p(x_t|y_{1:t}) = \frac{p(y_{1:t}|x_t)p(x_t)}{p(y_{1:t})} \tag{1}$$

The term $p(y_{1:t}|x_t)$ is the *likelihood* of the neuron firing at various values of the driving signal given the model state. The term $p(x_t)$ refers to the *prior density* for the model states, while the term $p(y_{1:t})$ refers to the *marginal density*, which is the probability of the observed measurements averaged over all possible states.

In parameter tracking problems, (1) is often solved iteratively, since knowledge of the state at any given time affects future model predictions. An iterative implementation of (1) is formulated as follows:

$$p(x_t|y_{1:t}) \propto p(x_{t-1}|y_{1:t-1}) \cdot \frac{p(y_t|x_t)p(x_t)}{p(y_t|y_{1:t-1})} \tag{2}$$

which assumes that observations depend only on the current model state.

B. Particle Filtering

Monte Carlo-style *particle filtering* is an implementation of (2) that has been applied to neural tracking problems [8]. A series of states (particles) is simulated at each time t; those states that best predict the observations are selectively retained (resampled). Although state values can be simulated directly from the prior $p(x_t)$, a more flexible method is to simulate state values from a simpler *proposal* density, π, that approximates the posterior at time t [5]. When simulating values of $p(y_t|x_t)p(x_t)$ using values of x_t taken from a proposal density, those simulated values must be weighted by the density of the proposal:

$$p(y_t|x_t)p(x_t) \sim \frac{p(y_t|x_t^{(k)})p(x_t^{(k)})}{\pi(x_t^{(k)}|x_{t-1}, y_t, \theta)} \tag{3}$$

where the superscript (k) indexes the simulated state values and θ is a *hyper-parameter*, which is commonly used to support multi-level statistical models [5], [7]. Substituting (3) into (2) results in the Sequential Importance Sampling equation:

$$w_t^{(k)} \propto w_{t-1}^{(k)} \frac{p(y_t|x_t^{(k)})p(x_t^{(k)}|x_{t-1}^{(k)}, \theta)q(\theta)}{\pi(x_t^{(k)}|x_{t-1}, y_t, \theta)} \tag{4}$$

where the posterior density $p(x_t|y_{1:t})$ has been recast as a weight $w_t^{(k)}$ subject to the constraint that all weights at time t sum to unity. We use the simplifying assumption that the current model state x_t depends exclusively on the preceding state x_{t-1} and on θ; the $q(\theta)$ term refers to the prior density of θ. Note that the term $p(y_t|y_{1:t-1})$ from (2) is a constant with respect to x_t (at each timestep) and is therefore subsumed into the constant of proportionality.

C. Auxiliary Particle Filtering

Equation 4 shows that one of the main problems arising from classical particle filters is that new weights (at time t) are calculated as a product of the old weights (at time $t-1$) and other terms. This tends to cause weight degeneracy, characterized by weights progressively approaching zero over time. Auxiliary particle filters (see [6]) solve this problem by selecting the proposal densities $\pi(x_t|x_{t-1}^{(k)}, y_t, \theta)$ to be the product of (a) the old weight $w_{t-1}^{(k)}$, (b) an approximation to the new likelihood, and (c) the joint distribution of the state x_t and hyperparameter θ. Substituting this choice of π into (4) yields a new weight $w_t^{(k)}$ that does not depend on the old weight $w_{t-1}^{(k)}$.

Many approaches to the task of state estimation employing models with hyperparameters have been offered (see [9]). Among these approaches, the most natural involves treating θ as a parameter having it's own prior distribution. This approach (see [7]) incorporates a Bayesian formulation in which, at each time stage t, both the state x_t and the hyper-parameter θ are simulated using the product of the proposal density π and the prior $q(\theta)$. The simulated state values are retained while their hyper-parameter counterpart values are discarded. This simulation requires the use of Gibbs sampling (see [10]). Gibbs sampling is a Markov Chain Monte Carlo (MCMC) technique used to do joint simulation by concatenating many single conditional simulations.

It is frequently advantageous to formulate the firing rate model in terms of *time blocks* rather than individual time steps. The neural model parameters are assumed to be stationary over the duration of each block. This has the advantage of making each parameter estimate more accurate at the expense of reducing accuracy when firing rates change over small time intervals.

III. METHODS

We tested the ability of the auxiliary particle filter to track the time-varying receptive field of CA1 hippocampal pyramidal neurons tuned to the location of a rat running back and forth along a linear track [4]. The neural firing rate was assumed to be an intensity function taking the form,

$$\lambda_t = \exp\left\{\alpha_t - \frac{(u_t - \mu_t)^2}{2\sigma_t^2}\right\} \tag{5}$$

where u_t denotes the position of the animal on the track at time t. Three parameters characterize the receptive field model and comprise the state vector x_t:

- α_t signifies the log of the maximum instantaneous firing rate;
- μ_t signifies the center of the receptive field;
- σ_t signifies the standard deviation of the width of the receptive field.

Neuron spiking activity was modeled as a Poisson process with an arrival rate of $\lambda_t \Delta_t$ spikes per timestep (with $\Delta_t = 5$ ms).

A. Dataset

Three different simulated experiments were conducted to assess the performance of the extended auxiliary particle filter. In all three cases, a virtual rat ran back and forth along a 300 cm linear track at a constant velocity of 120 cm/s for 125 s. The parameters α_t and σ_t were the same for all three experiments, with $\alpha_t = 3 + 0.004t$, and $\sigma_t = 11.5 - 0.008t$. In the first experiment, termed *Normal*, μ_t varied smoothly over time as $\mu_t = 150 - 100\cos(2\pi t/125)$. In the second, or *No Firing* experiment, the place cell was temporarily tuned to a position that was outside of the 300 cm track, resulting in neural inactivity for a period of time. Here, $\mu_t = 100 + 5.6t$ for $0 \leq t < 62.5$s, and $\mu_t = 800 - 5.6t$ for $62.5 \leq t < 125$s. This tests the filter's ability to retain accurate estimation when the neuron does not fire and its ability to recapture estimation when firing returns. In the third, or *Jump* experiment, μ_t jumps abruptly, which tests the filter's ability to adapt to rapidly changing parameters. In this case, $\mu_t = 50 + 3.2t$ for $0 \leq t < 62.5s$ and $\mu_t = -150 + 3.2t$ for $62.5 \leq t < 125$s.

B. Filter Implementation

The auxiliary particle filter was implemented according to the method specified by Liu and West (for more detail, see section 10.4 of [7]). We assumed that the prior distribution of the state x_t is Gaussian with mean $(x_{t-1} + \theta)$ and a diagonal covariance matrix Q that quantifies the prior error sizes of the parameters to be estimated. The proposal densities chosen for α_t, σ_t and μ_t were normal distributions with mean θ and variances of $\sigma_\alpha^2 = 0.5$, $\sigma_\sigma^2 = 0.05$, and $\sigma_\mu^2 = 3$. The number of particles was chosen to be 500. Two different block lengths (see section II-C) were considered: $n = 50$ and $n = 500$ samples, which correspond to block durations of $t = 0.25$ and $t = 2.5$s. The results of each experiment were averaged over 20 trials.

The results of the auxiliary particle filter were compared against the stochastic state point process adaptive filter, which Eden et al developed and demonstrated using the same neural firing model (i.e. Equation 5) used in the present study [4]. This non-Monte Carlo (i.e. deterministic) method is a two-step process in which the mean and variance of the state vector are first estimated from the prior density and then refined using current observations. The essence of this method is contained in Equations 2.7-10 of [4].

IV. RESULTS

The three experiments (*Normal*, *No Firing*, and *Jump*) were conducted using each of the three filters: the auxiliary particle filter with time blocks of durations $n = 50$ (APF_{50}) and $n = 500$ (APF_{500}) samples, and the stochastic state point process filter (SSPPF). Each experiment was averaged over 20 trials. Experiments were evaluated by comparing the estimated intensity function (5) to the true intensity function.

TABLE I

MSE FOR ESTIMATED VS. TRUE INTENSITY VALUES (λ_t)

	Normal	*No Firing*	*Jump*
APF$_{50}$	325	$\sim 10^{13}$	137
APF$_{500}$	28.1	28.2	28.0
SSPPF	$\sim 10^{30}$	NaN	NaN

The mean square error (MSE) was used as the metric of comparison.

The results of this study are summarized in Table I. The most accurate filter was found to be APF_{500}, followed by APF_{50}, both of which were robust to large and sudden changes in neural firing patterns. In contrast, SSPPF estimates frequently diverged significantly, producing effectively unusable results.

The results from the *Normal* experiment are summarized in Figs. 1 and 2; these show the estimated and true intensity functions (Fig. 1) and receptive field centers (Fig. 2). Fig. 2, in particular, demonstrates that both of the auxiliary particle filters tracked the receptive field center to within acceptable accuracy. As expected, the parameter estimates trail the true values because of the time block durations which effectively enforce a time lag. The estimate of μ_t affects where the center of the intensity function lies; the time lag in estimating μ_t causes a shift in the estimated intensity functions, as seen in Fig. 1.

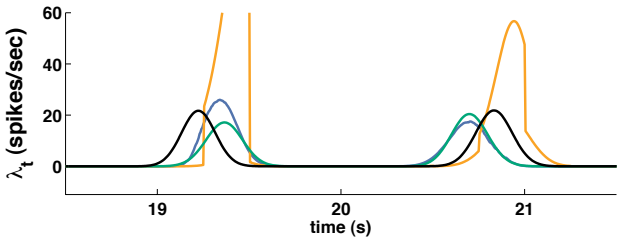

Fig. 1. Estimated vs. true firing intensity functions for the *Normal* experiment over a representative time window chosen prior to the divergence of the SSPPF. Black is True; Green is APF_{500}; Orange is APF_{50}; Blue is SSPPF.

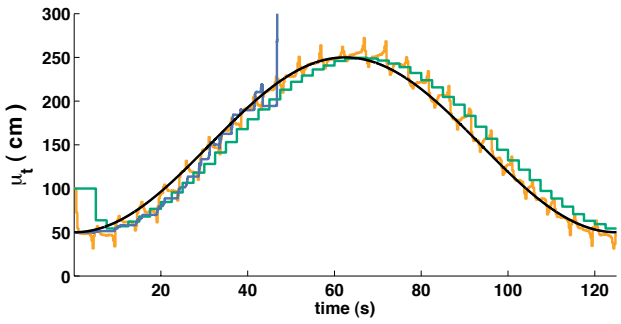

Fig. 2. Estimated vs. true receptive field center (μ_t) for the *Normal* experiment. Black is True; Green is APF_{500}; Orange is APF_{50}; Blue is SSPPF.

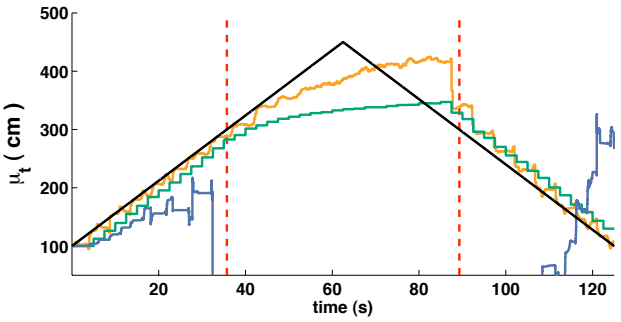

Fig. 3. Estimated vs. true receptive field center (μ_t) for the *No Firing* experiment. The red hash marks delineate the beginning and end of the No Firing interval, which occurs when μ_t is beyond the limits of the 300 cm track. Black is True; Green is APF_{500}; Orange is APF_{50}; Blue is SSPPF.

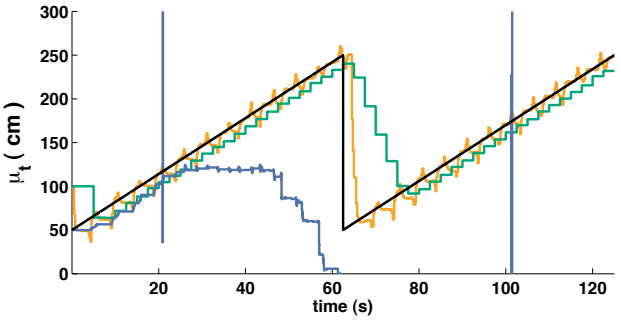

Fig. 4. Estimated vs. true receptive field center (μ_t) for the *Jump* experiment. Black is True; Green is APF_{500}; Orange is APF_{50}; Blue is SSPPF.

The results from the *No Firing* experiment are summarized in Fig. 3. A tracking filter would ideally indicate little or no change in estimated parameter values in the absence of neural activity. The auxiliary particle filters exhibited this property and were rapidly able to resume accurate tracking once firing recommenced. The *Jump* experiment is summarized in Fig. 4. Again, both auxiliary particle filters were able to adapt to the instantaneous shift in the receptive field center. The APF_{50} filter, with its shorter time block duration, adapted more rapidly than the APF_{500} filter. Figs. 2-4 show that the SSPPF filter is consistently and inherently unstable in the context of the three experiments presented here.

V. DISCUSSION

This paper demonstrates the application of auxiliary particle filtering to the problem of neural parameter estimation. This problem is of vital interest both for neuroscientists studying brain functionality at the individual neuron level and for engineers developing brain-machine interfaces. Auxiliary particle filtering has been shown to be more stable than other parameter tracking methods, especially for noisy datasets or complicated intensity models. In this context, it is appropriate for neural parameter tracking, which is characterized by noisy recordings, inaccurate spike detection and sorting, and rapidly changing parameters. The results of this work clearly demonstrate that the auxiliary particle filter is well suited for neural parameter tracking in neural models with multiple parameters. In particular, the APF_{50} and APF_{500} algorithms were stable during periods of neural quiescence.

The length of the time block over which tracking is averaged is an important design parameter. Although this work did not attempt an exhaustive analysis of time block duration, it was shown that APF_{500}, (duration $t = 2.5$ s) outperformed APF_{50}, (duration of $t = 0.25$ s). This improvement in accuracy came at the cost of an increased time lag between true and estimated parameter values (see Figs. 2-4). However, under some circumstances, increased time block duration may be to the advantage of the filter. For example, Fig. 3 indicates that the APF_{500} has a ten fold slower drift than that of APF_{50} during the No Firing interval. This is as expected, since the APF_{500} state vector is updated only one tenth as often as that of APF_{50}. It remains unclear whether further increases in time block duration would continue to yield improved tracking.

The SSPPF algorithm was found to be sensitive with respect to small changes in the data set, providing good tracking results on some runs and filter divergence on others. A major deficiency of the SSPPF is that it requires inversion of a covariance matrix which tends to become singular (and hence uninvertible) over time, thus causing the simulation to effectively collapse. This causes either highly inaccurate simulation results (i.e. $\sim 10^{30}$ for the *Normal* simulation) or no results at all as with the *No Firing* and *Jump* cases (see Table I).

VI. CONCLUSION

The auxiliary particle filter has been shown to be a stable and accurate method for tracking neural parameters. In the future, this work will be extended into a multi-neuron framework in which both the neural parameters and the track position will be estimated. Future testing with actual neural recordings will provide further validation of this method.

REFERENCES

[1] MR Mehta, CA Barnes, and BL McNaughton, "Experience-dependent, asymmetric expansion of hippocampal place fields," *P Natl Acad Sci Usa*, vol. 94, no. 16, pp. 8918–21, Aug 1997.

[2] S Haykin, *Adaptive Filter Theory*, Prentice Hall, Englewood Cliffs, NJ, 4th edition, 1984.

[3] EN Brown, DP Nguyen, LM Frank, MA Wilson, and V Solo, "An analysis of neural receptive field plasticity by point process adaptive filtering," *P Natl Acad Sci Usa*, vol. 98, no. 21, pp. 12261–12266, Jan 2001.

[4] UT Eden, LM Frank, R Barbieri, V Solo, and EN Brown, "Dynamic analysis of neural encoding by point process adaptive filtering," *Neural Comput*, vol. 16, no. 5, pp. 971–998, Jan 2004.

[5] NJ Gordon, DJ Salmond, and AFM Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *Iee Proc-F*, vol. 140, no. 2, pp. 107–113, Jan 1993.

[6] MJ Pitt and N Shephard, "Filtering via simulation: Auxiliary particle filters," *J Am Stat Assoc*, vol. 94, no. 446, pp. 590–599, Jan 1999.

[7] J Liu and M West, "Combined parameter and state estimation in simulation-based filtering," in *Sequential Monte Carlo Methods in Practice*, chapter 10. Springer, 1st edition, 2001.

[8] Y Wang, A Paiva, and J Principe, "A monte carlo sequential estimation for point process optimum filtering," *Neural Networks, 2006. IJCNN '06. International Joint Conference on*, pp. 1846 – 1850, Jun 2006.

[9] A Doucet, N deFreitas, and N Gordon, *Sequential Monte Carlo Methods in Practice*, Springer, 1st edition.

[10] JS Liu, "The Gibbs Sampler," in *Monte Carlo Strategies in Scientific Computing*, chapter 6. Springer, 2002.