

Learning Hybrid System Models for Supervisory Decoding of Discrete State, with applications to the Parietal Reach Region

Nicolas Hudson
Mechanical Engineering
California Institute of Technology
hudson@caltech.edu

Joel W. Burdick
Mechanical Engineering and Bioengineering
California Institute of Technology
jwb@robotics.caltech.edu

Abstract—Based on Gibbs sampling, a novel method to identify mathematical models of neural activity in response to temporal changes of behavioral or cognitive state is presented. This work is motivated by the developing field of neural prosthetics, where a supervisory controller is required to classify activity of a brain region into suitable discrete modes. Here, neural activity in each discrete mode is modeled with nonstationary point processes, and transitions between modes are modeled as hidden Markov models. The effectiveness of this framework is first demonstrated on a simulated example. The identification algorithm is then applied to extracellular neural activity recorded from multi-electrode arrays in the Parietal reach region of a rhesus monkey, and the results demonstrate the ability to decode discrete changes even from small data sets.

I. INTRODUCTION

Cortical neuroprostheses are being developed to restore motor function in individuals with high level spinal cord injuries or severe motor disorders (e.g. Lou Gehrig's disease). Neuroprostheses work by recording the spiking activity of multiple neurons in cortex and decoding movement intent or movement plans from this neural activity to generate control signals that can be used to drive devices such as prosthetic arms or computer interfaces [1], [2], [3]. Future practical clinical neuroprostheses will require a *supervisory decoder* whose job is to classify, in real time, the *discrete* cognitive or behavioral *state* of the brain region from which the neural signals are recorded. For example, the supervisory decoder must determine: (1) if the prosthetic patient is awake or conscious; (2) if the patient wants to use the prosthetic; (3) if the brain is currently planning a movement that should be decoded by the prosthetic; (4) if or when the movement is to be executed; (5) if the patient wants to change or scrub a plan while it is being executed, etc. The knowledge of the current state in the evolution of the planning process can be used in a variety of ways. For example, depending upon the current state, different algorithms, or different parameters in the algorithm, can be applied to the decoding of movement plans.

This paper presents a new method, based on recently developed techniques in hybrid systems model identification [4], to design a supervisory discrete state neural decoder. The design of the supervisory decoder will consist of two parts: first, the identification (or learning) of a hybrid model that represents neural activity in each discrete cognitive state as well as the transitions rules between cognitive states.

The identification of this hybrid model will typically be conducted off-line on a training data set. The second part is a real time decoder which uses the identified hybrid model to classify the current neural activity into discrete cognitive states. The focus of this paper is on the model identification process.

The term "Hybrid System" refers to a system with both continuous and discrete states. Here the discrete state will refer to the cognitive state of the brain, and the continuous state refers to the observed neural activity, such as firing rate. There are a number of reasons we use the framework of hybrid systems theory to formulate our approach to the design of supervisory decoding systems. First, the supervisory decoding problem is naturally formulated in this framework. Second, the process of learning a supervisory decoding model requires both the identification of the parameters of the supervisory decoding model and the simultaneous classification of neural activity into discrete cognitive modes. These distinct computational processes are easily handled in a Hybrid System Identification framework [5], [4]. Third, our formalization of the problem in hybrid systems terms allows for scaling of our method to reasonably complex hybrid supervisory decoding systems. Fourth, if neuroprostheses are to become widely used in clinical applications, a formal and automated approach to their design is necessary so that the process of adapting a prosthetic to each patient is not so labor intensive.

The idea of using discrete state, or supervisory, decoders in neural prosthetic systems is not original to us. It dates at least to the work of Shenoy et al. [3], who developed, using an ad hoc approach, a finite state machine model and decoder that classified plan activity from the Parietal Reach Region into three discrete states; a baseline state, a plan state and a reach state. Using off-line analysis, they showed that the imposition of a supervisory decoder on the decoding process could improve overall system accuracy. Recently, Kemere et. al [6] have demonstrated, using signals from dorsal premotor cortex, decoding two different discrete states. Their work assumed a homogeneous Poisson rate model for neural firing, and used an expectation-maximization framework to find the model parameters.

The work presented in this paper easily handles an arbitrary number of discrete states, uses a Gibbs sampling approach suitable for large dimensional problems, and is

not restricted to a homogeneous Poisson firing rate model. While their work is not directly related to the subject of this paper, it should be noted that Wu et al. [7] have used a switching Kalman filter, a type of hybrid system, for decoding continuous arm movements.

We present two examples to illustrate our approach. The first is a simple simulated example to show some of the characteristics of our approach. The second example applies our method to data recorded from the parietal cortex of a macaque monkey. Previous work [3] suggests that the parietal reach region (PRR) may be well suited for generating signals useful for decoding the discrete cognitive state in prosthetics applications, as it encodes plan activity selective for arm movements, which is not dependent upon actual movement occurring. This particular example shows that our method can identify a model even from a very small training set.

II. HYBRID MODEL DEFINITION

We choose a hybrid model to represent neural activity in each discrete cognitive state of the brain, as well as to describe transition rules between each discrete state. To be practical, these models need be of a form that allows for efficient identification routines to be developed, yet rich enough to capture all important aspects of the observed neural activity.

The neural activity in each cognitive state will be represented using a nonstationary point process [8]. Spike arrival times are discretized into sufficiently small time bins (typically 1 ms) so that only one spike at most is assigned to each bin. Each bin is indexed by k , and corresponds to the discrete time instant t_k to t_{k+1} . The signal y_k is the number of spikes arriving in the interval $(t_k, t_{k+1}]$. The probability of a spike occurring during the k^{th} time bin is governed by a non-stationary firing rate $\lambda_k(t)$:

$$y_k \sim f(\lambda_k(t)) \quad , \quad (1)$$

where $f(\cdot)$ is a Poisson distribution. Following Tuccolo et al [8], we assume that the firing rate depends on previous time history and other covariates of interest captured by the log linear function:

$$\lambda_k = \exp(\beta_0 + \beta^T x_k) \quad , \quad (2)$$

where x_k is a vector containing the previous spiking history and other covariates. The model (1)-(2) is a Generalized Linear Model (GLM).

It is assumed that there are a finite number N of cognitive states in the brain, that are of interest. Changes in cognitive state will be modeled using a hidden Markov model (HMM). This will be extended in future work to nonstationary or variable duration HMMs [9], which should more adequately describe discrete state transitions, yet are able to be incorporated into the presented algorithm. Moreover, while this paper focuses on inhomogenous Poisson firing rate models, our method can be extended to any firing rate model that assumes a GLM form [4].

In recent work, the authors have developed a new method to identify hybrid system models combining generalized

linear model dynamics with a Hidden Markov switching structure [4]. Here we restrict our attention to a sub-class of such models, termed Poisson Generalized Linear Hidden Markov Models (PGLHMM).

A. PGLHMM Definition

A PGLHMM is formed around a set of N unobservable discrete states, $\mathcal{S} = \{S_1, S_2, \dots, S_N\}$, whose evolution is governed by a first order Markov process. Let discrete instants of time be indexed by $\{t_1, t_2, \dots, t_k, \dots, t_T\}$. At each t_k , let m_k denote the mode index, i.e., at t_k the system is in state S_{m_k} . The probability of switching between modes of the system is governed by a first order Markov chain with transition matrix $A = [a_{i,j}]$:

$$P(m_k = j | m_{k-1} = i) = a_{i,j} \quad . \quad (3)$$

The observed system output y_k , at time t_k , is generated by a Poisson GLM, which depends on the discrete mode, m_k :

$$y_k \sim f(\lambda_k), \text{ where } \lambda_k = \begin{cases} \exp(\theta_1^T x_k) & \text{if } m_k = 1 \\ \vdots & \vdots \\ \exp(\theta_N^T x_k) & \text{if } m_k = N \end{cases} \quad , \quad (4)$$

where $f(\cdot)$ is the Poisson distribution, and vector x_k is a regressor of previous outputs and other covariates u_k :

$$x_k = [1, y_{k-1}, \dots, y_{k-n_y}, u_{k-1}, \dots, u_{k-n_u}]^T \quad . \quad (5)$$

The parameters $\theta_i \in \mathbb{R}^{n_y+n_u}$, $i = 1, \dots, N$ are associated with each discrete mode, and govern the dynamics within a mode.

For convenience define: $\Theta = [\theta_1, \dots, \theta_N]$ and $M = [m_1, \dots, m_k]$.

III. HYBRID MODEL IDENTIFICATION

Identifying the hybrid model is complex because it involves simultaneously classifying observed data, $\{y_k\}_{k=1}^T$ and $\{x_k\}_{k=1}^T$, into discrete modes S , and identifying the parameters associated with each mode Θ , as well as the parameters associated with discrete mode transition A .

The identification problem is equivalent to either approximating or maximizing the joint posterior pdf:

$$p(\Theta, A, M | \text{observed data}) \quad . \quad (6)$$

In practice (6) is impossible to solve analytically, and thus numerical methods must be used. In this paper we will use Gibbs sampling, a type of Markov Chain Monte Carlo (MCMC) to estimate the posterior pdf (6).

A. Gibbs Sampling

Gibbs Sampling provides a unique form which allows the joint pdf (6) to be decomposed in a natural way, into component parts of identification, $p(\Theta, A | M, \text{observed data})$, and classification, $p(M | \Theta, A, \text{observed data})$. This method is well suited to high dimensional problems, and has asymptotic convergence properties.

Gibbs sampling is a MCMC method for sampling from a potentially complicated joint pdf, $p(\phi_1, \dots, \phi_n)$, where

ϕ_1, \dots, ϕ_n are system states or parameters. Gibbs sampling can be usefully applied when the joint pdf $p(\phi_1, \dots, \phi_n)$, has associated conditional pdfs,

$$p(\phi_1 | \phi_2, \dots, \phi_n), \dots, p(\phi_i | \phi_1, \dots, \phi_{i-1}, \phi_{i+1}, \dots, \phi_n), \dots, p(\phi_n | \phi_1, \dots, \phi_{n-1}),$$

that can be efficiently sampled from (e.g., there is a closed form solution for the pdf). A single step in the Gibbs sampling cycle requires one sample to be drawn sequentially from each of the conditional pdfs, using the most recent sampled value in subsequent conditional arguments. At the end of each step, a new sample $\hat{\phi} = [\hat{\phi}_1, \dots, \hat{\phi}_n]$ has been drawn. As the Gibbs sampler iterates through many steps, the samples $\{\hat{\phi}\}$ tend to the joint distribution [10]. In theory this property implies that the maximum of $p(\phi_1, \dots, \phi_n)$, can always be found using Gibbs sampling, as opposed to Expectation Maximization methods where only a local maximum is guaranteed. In practice, only a finite number of samples are drawn, and multiple runs of the Gibbs sampling algorithm from different starting points are usually conducted to test for convergence.

We first describe the Gibbs sampling algorithm, as applied to identifying PGLHMMs, and then discuss issues related to its design and use.

B. Algorithm

Draw z_{max} number of samples from the joint distribution $P(\Theta, A, M | X, Y)$ of a PGLHMM (3)-(5), given the data set $X = \{x_k\}, x_k \in \mathbb{R}^n, Y = \{y_k\}, k = 1, \dots, T$, and the number of discrete states $N: \mathcal{S} = \{S_1, \dots, S_N\}$.

1) Define parameterized prior distributions for Θ and A , using prior information to select parameters:

- a) set $p(\theta_i(j))$ as normal distributions for $i = 1, \dots, N$ and $j = 1, \dots, n$.
- b) each row of $A = [a_{(i,j)}]$ is assumed to be independent and the prior for A is defined as a set of Dirichlet distributions $\mathcal{D}(\cdot)$:

$$p(a_{(i,1:N)}) = \mathcal{D}(\alpha_1^i, \dots, \alpha_N^i),$$

where α_j^i are the parameters of the Dirichlet prior [11].

2) Initialize parameter samples: $\hat{A}^{(0)}, \hat{M}^{(0)}, \hat{\Theta}^{(0)}$. These initial samples can either be drawn from the priors or be set to an arbitrary initial guess. The z^{th} sample of a variable, ϕ , is denoted by: $\hat{\phi}^{(k)}$.

3) set $z = 1$

4) Sample from $p(M | \Theta, A, Y, X)$, the conditional discrete mode density. The discrete modes are sampled from sequentially:

- a) set $\hat{m}_1^{(z)} = 1$
- b) for $k = 2 : T-1$, draw a sample $\hat{m}_k^{(z)}$ from the discrete distribution:

$$\hat{m}_k^{(z)} \sim P(m_k | y_k, \Theta, m_{k-1}, m_{k+1}, A)$$

$$= \frac{\hat{a}_{(\hat{m}_{k-1}^{(z)}, m_k)}^{(z-1)} f(y_k | \hat{\theta}_{m_k}^{(z-1)}, x_k) \hat{a}_{(m_k, \hat{m}_{k+1}^{(z-1)})}^{(z-1)}}{\sum_{j=1}^s \hat{a}_{(\hat{m}_{k-1}^{(z)}, j)}^{(z-1)} f(y_k | \hat{\theta}_j^{(z-1)}, x_k) \hat{a}_{(j, \hat{m}_{k+1}^{(z-1)})}^{(z-1)}}$$

end

c) draw sample $\hat{m}_T^{(z)}$ from the discrete distribution:

$$\hat{m}_k^{(z)} \sim P(m_T | y_T, \Theta, m_{T-1}) = \frac{\hat{a}_{(\hat{m}_{T-1}^{(z)}, m_T)}^{(z-1)} f(y_T | \hat{\theta}_{m_T}^{(z-1)}, x_T)}{\sum_{j=1}^s \hat{a}_{(\hat{m}_{T-1}^{(z)}, j)}^{(z-1)} f(y_T | \hat{\theta}_j^{(z-1)}, x_T)}.$$

5) Sample from $p(A | \Theta, M, Y, X)$. Each row of A is sampled independently:

for $i = 1 : N$

$$\hat{a}_{(i,1:N)}^{(k)} \sim p(a_{i,1:N} | M) = \mathcal{D}\left(\alpha_1^i + \sum_{k=2}^T \delta(\hat{m}_{k-1}^{(z)} = i) \delta(\hat{m}_k^{(z)} = 1), \dots, \alpha_N^i + \sum_{k=2}^T \delta(\hat{m}_{k-1}^{(z)} = i) \delta(\hat{m}_k^{(z)} = N)\right)$$

end,

where δ is the delta function

6) Sample from $p(\Theta | M, A, Y, X)$:

a) assign data into discrete modes using $\hat{M}^{(k)}$

for $i = 1, \dots, N$

$$\mathcal{X}^i = \{x_k : \hat{m}_k^{(z)} = i, k = 1, \dots, T\}$$

$$\mathcal{Y}^i = \{y_k : \hat{m}_k^{(z)} = i, k = 1, \dots, T\}$$

end

b) conditioning on the sets $\mathcal{X}^i, \mathcal{Y}^i$, the distributions for the regressor parameters θ_i in each mode are sampled independently.

for $i = 1, \dots, N$

$$\hat{\theta}_i^{(k)} \sim p(\theta_i | \mathcal{Y}^i, \mathcal{X}^i) \propto P(\mathcal{Y}^i, \mathcal{X}^i | \theta_i) p(\theta_i). \quad (7)$$

end

The likelihood $P(\mathcal{Y}^i, \mathcal{X}^i | \theta_i)$ is a poisson GLM, and samples are drawn from (7) using adaptive rejection sampling [12].

7) set $z = z + 1$, **if** $z > z_{max}$ stop, **else** goto step 4.

C. Notes on Derivation of Algorithm

The derivation of this algorithm is described more completely in previous work [4]. A large part of this algorithm, sampling from both the discrete state conditional density function and the Markov transition parameter density function, algorithm steps 4) and 5), follows from [11].

One the reasons that this algorithm is tractable is because the likelihood function, $P(\mathcal{Y}^i, \mathcal{X}^i | \theta_i)$, takes the form of a GLM. It can be shown [13] that this GLM likelihood is log-concave, which is a requirement to use adaptive rejection sampling [12] which is significantly more efficient than general rejection sampling.

It is also important to note that the stated algorithm can be trivially extended to multi-output systems, where y_k is a vector, if we assume that each element of the output is independent. All that is required is to give each element of the output an independent model of the form (4). This results in the likelihood $f(y_k|\theta_{m_k}, x_k)$ reducing to the product of the likelihood of each element of y_k , each of which are Poisson distributions.

IV. DECODING USING THE MODEL

Although the focus of this paper is not on the real-time decoding algorithm, two decoding algorithms are presented for completeness. First a causal Bayesian filter is derived, appropriate for real time use. Secondly the non-causal Viterbi algorithm is used. It is emphasized that these decode algorithms are not optimal for this application, and require further research.

A. Simple Bayesian Decoding

In the Bayesian philosophy, we are interested in estimating the discrete distribution $P(m_k) = [P(m_k = 1), \dots, P(m_k = N)]$, at step k , conditioned on the observation y_k , and the distribution of the previous state $P(m_{k-1})$. Using Bayes' rule and the Total Probability Theorem, the recursion formula is derived:

$$P(m_k = i) = P(y_k | m_k = i) \sum_{j=1}^N P(m_k = i | m_{k-1} = j) P(m_{k-1} = j) \text{ for } i = 1, \dots, N. \quad (8)$$

Note that the likelihood $P(y_k | m_k = i)$ is the Poisson distribution, and the prior $P(m_k = i | m_{k-1} = j) = a_{ij}$ is defined by the Markov transition matrix A (i.e., the model identification step estimates the important parameters of these distributions). Starting with an initial distribution of the mode at the first time step $P(m_1)$, the recursion (8) is used to update the discrete state estimate in real time.

B. Viterbi Algorithm

The Viterbi Algorithm [14] is typically used to find the single best hidden discrete state sequence $M = \{m_1, \dots, m_T\}$ for Hidden Markov Models. It should be noted that it is not a causal filter, as it uses all observations $\{y_1, \dots, y_k, \dots, y_T\}$ when estimating the k^{th} discrete mode m_k . Due to the Markov property of the PGLHMM model, it is simple to extend and apply here.

V. SINGLE NEURON RECORDING, A SIMULATED EXAMPLE

To illustrate some key characteristics of our approach, a simulation of recorded spiking activity from a single neuron present in a higher brain cortex is created. This neuron's spiking activity is dependent on the unobservable discrete state of the surrounding cortex. For this simple example, the cortex has two discrete states; S_1 , an 'attention' state (i.e., the patient wants to actively use the neural prosthetic)

and S_2 , a 'baseline' state (i.e., sleep or disinterest in using the neural prosthetic). The transition of this cortical region between the 'attention' and 'baseline' states is assumed to follow Markov transition probabilities. The number of spikes in sequential 0.01s time bins, for a 10s interval is simulated.

The discrete modes are modeled by setting $m_1 = 1$ and evolving the discrete state m_k , $k = 1, \dots, 1000$, using Markov transitions with parameters $A = [a_{i,j}]$:

$$P(m_{k+1} = j | m_k = i) = a_{ij}, \text{ where } A = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}. \quad (9)$$

The neurons spiking activity in each mode is modeled with Poisson-GLMs. The firing rate, λ_k , in each mode S_i , is determined by two components: $\theta_i(1)$, the nominal firing rate of the mode, and $\theta_i(2)$, representing a change in rate depending on the spiking history. $\theta_i(2)$ can model refractory periods, a dwell period in spiking activity that is experienced immediately after spike firing.

$$\lambda_k = \begin{cases} e^{(\theta_1(1) + \theta_1(2)y_{k-1})} & \text{if } m_k = 1 \\ e^{(\theta_2(1) + \theta_2(2)y_{k-1})} & \text{if } m_k = 2 \end{cases}. \quad (10)$$

The following regressor parameters are used:

$$\theta_1 = [-1 \quad -10]^T, \quad \theta_2 = [-2 \quad 0]^T. \quad (11)$$

The parameters (11), correspond to a nominal firing rate of 36.78 Hz in the 'attention' state, and a nominal rate of 13.53 Hz in the 'baseline' state.

The number of spikes in the current time bin are generated from a Poisson distribution with rate λ_k :

$$y_k \sim \text{Poisson}(\lambda_k). \quad (12)$$

An output sequence y_k , $k = 1, \dots, 1000$ was generated from the single neuron model by using Poisson and discrete random number generators in Matlab. There were a total of 211 spike events over the simulated 10 second duration.

The algorithm in Sec. III-B was run, setting $z_{max} = 5000$; the last 3000 generated samples were used for statistical analysis. Regressor parameter priors are set to dispersed normal distributions: $\theta_{i,j} = \mathcal{N}(0, 10^2)$ for $i \in \{1, 2\}$, $j \in \{1, 2\}$. Dirichlet priors are used for each row of A : $[a_{11} \quad a_{12}] \sim \mathcal{D}([\alpha_1 \quad \alpha_2])$, $[a_{21} \quad a_{22}] \sim \mathcal{D}([\alpha_2 \quad \alpha_1])$. Several different informative parameterizations were chosen that incorporate the assumption that sequential modes values m_k , m_{k+1} are more likely to belong to the same mode S_i :

$$[\alpha_1 \quad \alpha_2] = [90 \quad 10], [80 \quad 20], [70 \quad 30]. \quad (13)$$

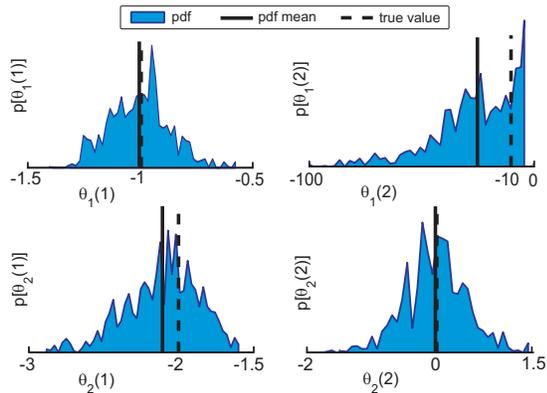
The solution was invariant when using different informative priors (13), and the key parameter estimates matched the model values (see Table I).

The only wide discrepancy between the MAP and Expectation estimates is for the refractory parameter $\theta_1(2)$. Gibbs sampling allows analysis of the posterior densities, by constructing a histogram of the samples. The posterior density for $\theta_1(2)$, shown in Fig. 1, has a large support, indicating that the parameter is unidentifiable from the generated data

TABLE I

MODEL PARAMETER ESTIMATES. EXPECTED VALUE ($E[\cdot]$) AND THE MAXIMUM A POSTERI (MAP) ESTIMATES ARE USED, COMPARED WITH ACTUAL PARAMETER VALUES (MODEL).

	Model	MAP	$E[\cdot]$
A	$\begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$	$\begin{bmatrix} 0.898 & 0.102 \\ 0.102 & 0.898 \end{bmatrix}$	$\begin{bmatrix} 0.897 & 0.102 \\ 0.096 & 0.904 \end{bmatrix}$
θ_1	$\begin{bmatrix} -1 & -10 \end{bmatrix}^T$	$\begin{bmatrix} -0.964 & -2.704 \end{bmatrix}^T$	$\begin{bmatrix} -1.003 & -25.01 \end{bmatrix}^T$
θ_2	$\begin{bmatrix} -2 & 0 \end{bmatrix}^T$	$\begin{bmatrix} -2.013 & 0.201 \end{bmatrix}^T$	$\begin{bmatrix} -2.108 & -0.021 \end{bmatrix}^T$

Fig. 1. Posterior distributions for model parameters Θ .

set. This posterior distribution remains bounded, because of the proper prior distribution used by the algorithm. This unidentifiability problem arises because the refractory physics of spike firing dictate that no sequential outputs y_k and y_{k+1} in S_2 both contain spikes. Hence the only information that can be deduced from the posterior distribution is the refractory parameter $\theta_1(2)$ significantly lowers the firing rate after a spike event has just occurred. The posterior densities thus allow the user to realize when a parameter is unidentifiable, or nearly unidentifiable.

VI. NEUROPHYSIOLOGICAL APPLICATION

The developed algorithm is applied to pre-recorded neural data from two 16 electrode arrays implanted in the Parietal reach region (PRR) of a rhesus monkey. This data set was part of a large study on cognitive control signals, [15], in which the details on the clinical procedure, include array types, and experimental procedures including the recording equipment are detailed in full.

In this previous study, extracellular action potentials were recorded during a delayed center-out reaching task, with eight potential targets. For each trial in the reaching experiment, there were several periods of interest: First a baseline state, which precedes a cue period, where a visual target is displayed in one of the eight reach directions, followed by a planning (or memory) period with no visual cues. After a variable amount of time, a go signal is presented, allowing the onset of movement, with the goal being a reach in the direction of the previous presented cue. This movement period is followed by a hold on the intended target, preceding redisplay of the target. Juice is rewarded if the actual reach direction and the cue match.

A subset of the available data, consisting of all successful

trials (meaning the actual reach direction and the cue aligned) in a single direction was used for the model identification, or training phase, of the supervisory decoder.

To test the performance of our approach even on very limited amounts of training data, the first ten trials of the subset were used to identify a PGLHMM model, with the remaining 50 trials used to test the performance of the identified model. A low number of training trials was chosen for two reasons: First, to test that the model identified from the training set provides decode performance over a long period of time. In this case, the 50 testing trials were conducted over a period of approximately two hours following the 10 training trials. Second, it is important to establish that in a clinical situation, large training data sets, and correspondingly large training times, are not required. It would be discouraging for a clinical patient to endure a large duration of training before starting to use a prosthetic device. Instead, if a small training set yields a usable model, the patient can then proceed to get visible feedback from the prosthesis, and the model could be adaptively updated as more data is received.

For testing the approach defined in this paper, three neurons recorded by the array that showed a change in activity between different experimental periods were chosen, by comparing the means and variances of the firing rates in each temporal experimental period as defined by the presented cues. All the selected neurons had a behavioral change in the interval between the go period and when the target is redisplayed. The cognitive state or behavioral state during this interval will be referred to as the *reach state*.

The goal of this analysis is to see if, given neural data from the training trials, with no cue information, a hybrid model can be generated that can discriminate between different temporal experimental periods solely from neural data, without reference to the behavioral or experimental markers. For simplicity the hybrid model will only decode two states, a *baseline state* (Mode 1) and the *reach state* (Mode 2).

The performance of the identified hybrid model will be evaluated by analyzing the decoding of discrete state from both the Viterbi and Bayesian decode algorithms.

A. Decoding Results

A successful decode will be defined to occur when the decoded reach period is contained in the interval between the onset of the experimental go signal and the time when the target is revealed. For each decoded trial, a *missed positive* is when no decode of the reach state occurred during the defined interval, and a *false positive* is when the decode algorithm determined the cognitive reach state occurred outside this interval. A *correct* decode will have neither a missed positive or any false positives. False positives were only checked during each trial between the cue on and the end of the reward, and not in the inter trial interval, as during this time it is uncertain what is actually taking place. Figure 2 shows a correctly decoded trial. Figure 3 shows a decode of a trial where the Bayesian decode algorithm gave false positives.

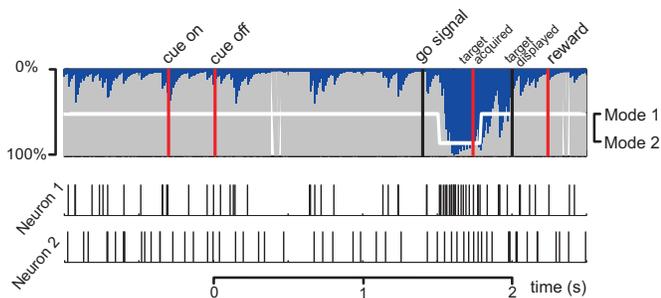


Fig. 2. Correct decode: Both the Bayesian decode algorithm and the Viterbi algorithm decode the cognitive reach state. The mode estimate generated by the Viterbi algorithm is shown by the white line. The Bayesian decode is shown by the dark blue histogram. The neural activity used for decoding is shown below the mode estimates.

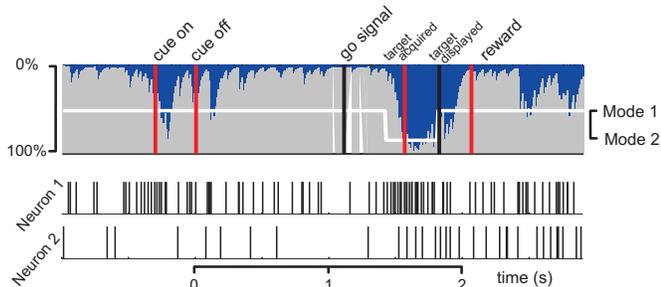


Fig. 3. Decode with false positives: The Bayesian decode indicated the cognitive reaching state occurs during the cue period.

The percentage of correctly decoded trials, as well as the percentage of trials with missed positives and false positives is shown in Tab. II. The number of neurons used in identifying the hybrid model and for real-time decoding was varied between one and three.

B. Discussion

When using actual neural data there is no “ground truth” to compare identified parameters. Instead, we use the real-time decoding performance to infer the suitability of the identified hybrid model and identification algorithm.

Decode performance utilizing the identified model approaches 80%, which strongly supports the suitability of the proposed supervisory decode framework. Furthermore, this was achieved using the presented Bayesian and Viterbi algorithms (Sec. IV), which are not optimal for our specified performance criterion. Also, multi-electrode array data, and not neurons specifically tuned for the task were used, and training was undertaken on only on a small data set.

While the presented examples validate the use of the proposed identification algorithm, there are several research steps to be addressed before implementation in prosthetic devices.

TABLE II
PERCENTAGE OF TRIALS CORRECTLY DECODED

		Number of Neurons		
		1	2	3
Bayesian	Missed Positives	%0	%0	%0
	False Positives	%70	%56	%48
	Correct	%30	%44	%52
Viterbi	Missed Positives	%14	%10	%12
	False Positives	%18	%16	%14
	Correct	%72	%78	%78

First, and most importantly, a suitable real-time decode algorithm is required. When considering future work in this area a promising observation is made: Note that the Bayesian decoder had no missed positives, and (as seen in Fig. 3), the false positives tend to be of short duration and low probability; any prosthesis action initiated could be stopped within a short time. An intelligently designed supervisory decode algorithm would trade off fewer missed positives for (possibly less detrimental) false positives.

Second, to increase performance, other signals, such as local field potentials should be incorporated into the current model. Note that this can be easily accomplished in the current framework if a suitable log-likelihood model (e.g. Gaussian) is defined. Third, the abilities of the current algorithm need to be analyzed by considering larger data sets, more discrete modes, and more complicated regressor formulation.

REFERENCES

- [1] B. Pesaran, S. Musallam, and R. Andersen, “Cognitive neural prosthetics,” *Current Biology*, vol. 16, pp. 77–80, 2006.
- [2] M. A. L. Nicolelis, “Brainmachine interfaces to restore motor function and probe neural circuit,” *Nature Reviews Neuroscience*, vol. 4, pp. 417–422, 2003.
- [3] K. Shenoy, D. Meeker, S. Cao, S. Kureshi, B. Pesaran, C. Buneo, A. Batista, P. Mitra, J. Burdick, and R. Andersen, “Neural prosthetic control signals from plan activity,” *Neuroreport*, vol. 14, no. 4, pp. 591–596, 2003.
- [4] N. Hudson and J. Burdick, “A stochastic framework for hybrid system identification with application to neurophysiological systems,” *Lecture Notes in Computer Science Hybrid Systems Computation and Control*, pp. 568–582, 2007, *To Appear*.
- [5] A. Juloski, G. Ferrari-Trecate, R. Vidal, S. Paoletti, and J. Niessen, “Comparison of four procedures for the identification of hybrid systems,” *Lecture Notes in Computer Science*, vol. 3414, pp. 354 – 369, 2005.
- [6] C. Kemere, B. Yu, G. Santhanam, S. Ryu, A. Afshar, T. Meng, and K. Shenoy, “Hidden markov models for spatial and temporal estimation for prosthetic control,” in *Society for Neuroscience, Abstract Viewer / Itinerary Planner*, Atlanta, GA, 2006, in press.
- [7] W. Wu, M. Black, D. Mumford, Y. Gao, E. Bienenstock, and J. Donoghue, “Modeling and decoding motor cortical activity using a switching kalman filter,” *IEEE Transactions on Biomedical Engineering*, vol. 51, pp. 933–942, June 2004.
- [8] W. Truccolo, U. T. Eden, M. R. Fellows, J. P. Donoghue, and E. N. Brown, “A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects,” *J Neurophysiol*, vol. 93, no. 2, pp. 1074–1089, 2005.
- [9] P. M. Djuric and J.-H. Chun, “An MCMC sampling approach to estimation of nonstationary hidden Markov models,” *IEEE Trans. on Signal Processing*, vol. 50, no. 5, pp. 1113–1123, May 2002.
- [10] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, Eds., *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.
- [11] C. Robert, G. Celeux, and J. Diebolt, “Bayesian estimation of hidden Markov chains: A stochastic implementation,” *Statistics & Probability Letters*, vol. 16, pp. 77–83, January 1993.
- [12] W. R. Gilks, N. G. Best, and K. K. C. Tan, “Adaptive rejection metropolis sampling within gibbs sampling,” *Applied Statistics*, vol. 44, no. 4, pp. 455–472, 1995.
- [13] P. Dellaportas and A. F. M. Smith, “Bayesian inference for generalized linear and proportional hazards models via gibbs sampling,” *Applied Statistics*, vol. 42, no. 3, pp. 443–459, 1993.
- [14] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.
- [15] S. Musallam, B. D. Corneil, B. Greger, H. Scherberger, and R. A. Andersen, “Cognitive control signals for neural prosthetics,” *Science*, vol. 305, pp. 258–262, 2004.