33rd Annual International Conference of the IEEE EMBS
Boston, Massachusetts USA, August 30 - September 3, 2011

Exploring Time-Scales of Closed-Loop Decoder Adaptation in Brain-Machine Interfaces

Amy L. Orsborn*, *Student Member, IEEE*, Siddharth Dangi*,
Helene G. Moorman, and Jose M. Carmena, *Senior Member, IEEE*

Abstract— **Performing closed-loop modifications of a brain-machine interface (BMI) decoder is a technique that shows great promise for improving performance. We compare two algorithms for implementing adaptations that update decoder parameters on different time-scales (discrete batches vs. online), and present experimental results of a non-human primate performing a standard center-out BMI task. To ensure that our experimental training models are representative of a broad range of paralyzed patients, our decoders were initially trained using neural activity recorded during subject observation of cursor movement. We find that both closed-loop adaptation algorithms can be used to boost BMI performance from 20-30% to 80%, yielding movement kinematics similar to natural arm movements. Based on insights derived from the performance of each algorithm, we propose that a hybrid of batch and online decoder adaptation may be the best approach.**

I. INTRODUCTION

Brain-machine interfaces (BMI) aim to help patients with motor disabilities by allowing users to control external actuators using their neural activity. BMI research has shown tremendous potential, with many demonstrations of primates and humans controlling devices using neural activity [1], [2]. However, significant challenges must be addressed before these systems can become clinically viable.

BMI decoding algorithms are typically trained offline by fitting neural activity against actual or desired movements. Training decoders offline, however, ignores differences between closed-loop and open-loop BMI. During closed-loop BMI, the subject can alter their neural activity in response to feedback to improve performance. Offline decoding does not account for this difference, and as a result, offline prediction power does not directly correlate with online performance [3]. Recent work has instead taken a closed-loop view of BMI, emphasizing the important roles of brain and machine adaptation. For instance, adaptation of neural activity can yield performance improvements [4]. Alternately, others have shown that closed-loop decoder adaptation (CLDA), where closed-loop errors are used to update the decoder, can also yield performance improvements [1], [5]–[7].

The high-performance achieved with CLDA proves its validity as a method for improving decoder performance. However, there are many potential methods of performing these adaptations. For instance, different methods have used various time-scales for decoder updates – [5] updated decoder parameters in real-time, while [6], [7] used a batch-based method with periodic decoder updates. The best time-scale for online decoder updates has not been investigated.

Closed-loop decoder modifications and choice of training algorithm may be particularly important for BMIs in disabled patients. For patients that cannot move, initial decoders are often seeded using neural activity evoked from imagined or intended movements [8]. This seeding may yield less robust initial performance than decoders seeded with actual movement, since sensory feedback experienced during movement contributes significantly to decoding power [9]. CLDA methods do not depend, in theory, on the decoder's initial seeding. In practice, however, boosting performance starting from a poor initial decoder may be more difficult, making it critical to develop a high-performance CLDA algorithm.

Towards developing clinically viable, robust BMI systems for patients, we present experimental results that investigate the appropriate time-scale for CLDA. We compare batch-based and online algorithms' ability to improve decoder performance using initial decoders seeded with neural activity from movement observation.

II. KALMAN FILTER MODEL

The goal of a BMI decoding algorithm, or decoder, is to estimate a subject's intended movements from observed neural activity. In our experiments, we used the Kalman Filter (KF), a common decoder for BMI applications [10]. Let x_t represent the kinematic state of a computer cursor and let y_t represent binned neuron spike counts. The KF assumes the following state evolution and state observation models:

$$x_t = Ax_{t-1} + w_t$$
$$y_t = Cx_t + q_t$$

where $w_t \sim \mathcal{N}(0, W)$ and $q_t \sim \mathcal{N}(0, Q)$ are noise terms. We use a position-velocity KF, as in [6], [7], where x_t is defined to include the cursor's position and velocity in both

This work was supported in part by the Natiaonal Science Foundation Graduate Research Fellowship (ALO, HGM), the DARPA Contract N66001-10-C-2008 (JMC), and the National Science Foundation CAREER CBET-0954243 (JMC).

*ALO and SD contributed equally to this work.

A. L. Orsborn is with the University of California, Berkeley - Unviversity of California, San Francisco Graduate Program in Bioengineering, University of California, Berkeley, CA 94729 USA. amyorsborn@berkeley.edu

Siddharth Dangi is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA. sdangi@eecs.berkeley.edu

Helene G. Moorman is with the Helen Wills Neuroscience Institute, University of California, Berkeley, CA 94720 USA. helenem@berkeley.edu

J. M. Carmena is with the Department of Electrical Engineering and Computer Sciences, Helen Wills Neuroscience Institute and Program in Cognitive Science, University of California, Berkeley, CA 94720 USA. carmena@eecs.berkeley.edu

978-1-4244-4122-8/11/$26.00 ©2011 IEEE

the horizontal and vertical directions, as well as a constant 1 to account for non-zero mean observations y_t (i.e. the neurons' baseline firing-rates).

Since we aim to achieve BMI control that mimics natural arm movements, the state transition matrices A and W (which model cursor dynamics) are estimated using arm kinematic data as the subject performs a center-out reaching task (see Section IV). For all decoder adaptations, this dynamics model is held fixed, and only the observation matrices C and Q, (which model how neural activity relates to cursor kinematics) are updated.

III. ONLINE TRAINING PARADIGMS

In offline decoder training, where intended kinematics are explicitly observed, decoder parameters are estimated to minimize the discrepancy between predicted and actual arm kinematics. In the online regime, however, errors arise due to the difference between the BMI's decoded kinematics and the user's intended (desired) kinematics, where the latter is unobserved. Thus, to retrain a decoder online, we must estimate the user's intended kinematics.

For a standard center-out task, various groups have devised clever methods of guessing a subject's intended kinematics [5]–[7]. We use the cursorGoal method developed by Gilja et al., which assumes the monkey always intends to move towards the target, and thus estimates intended velocities that point toward the current target [6], [7].

Given an estimate of intended kinematics, there are multiple ways to adjust a decoder's parameters. One paradigm is the "batch" approach, which updates decoder parameters only after collecting a certain amount of data [6], [7]. Alternatively, one could apply parameter updates on a much faster time-scale in the extreme case, during every KF iteration. We explore methods for both approaches below.

A. Batch estimation of KF parameters

The batch approach of parameter estimation entails collecting data and processing the entire batch at once to update the decoder's parameters. One standard batch estimate of the KF parameters C and Q is the maximum likelihood estimate:

$$C = YX^T(XX^T)^{-1}$$
$$Q = \frac{1}{N}(Y - CX)(Y - CX)^T$$

where the Y and X matrices are formed by tiling recorded neural activity and intended kinematics, respectively [11].

To implement batch-based decoder updates, the subject uses a decoder to perform closed-loop control while cursor kinematcs and neural data are collected. Once enough data is collected, a new decoder is trained using the inferred intended kinematics and observed neural activity.

B. Online updates of KF parameters

Decoder adaptation can also be implemented by updating parameters at every KF iteration. The Adaptive Kalman Filter, a recently developed KF decoder training variant for closed-loop BMI [12], is one example of an adaptive algorithm that updates decoder parameters in real-time.

The Adaptive KF's update equations for C and Q are

$$C^{(i+1)} = C^{(i)} - \frac{\rho}{\|x_t\|^2}\left(C^{(i)}x_t - y_t\right)x_t^T \quad (1)$$
$$Q^{(i+1)} = \alpha Q^{(i)} + (1-\alpha)q_t q_t^T \quad (2)$$

where $q_t \triangleq y_t - C^{(i+1)}x_t$, and x_t is the estimate of intended kinematics. The step-size ρ is typically chosen to be closer to 0, and α is typically chosen closer to 1. The update (1) is based on stochastic gradient descent, while the update (2) effectively implements an exponentially weighted moving average (see [12] for derivations).

To implement online updates, the subject uses the current decoder to perform closed-loop control. At each time-step, observed kinematics are transformed to intended kinematics, and the decoder parameters are updated using the above equations. The updated decoder is then used to predict cursor kinematics at the next time-step, and so on.

IV. METHODS

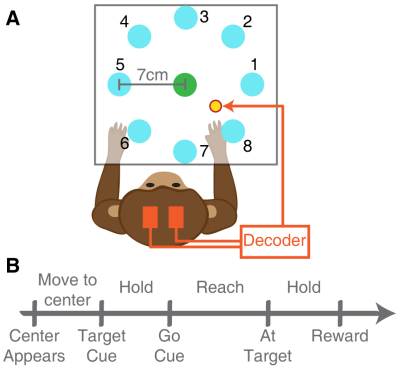

Fig. 1. Center-out task schematic (A) and structure (B).

One rhesus macaque (*macaca mulatta*) was trained to perform a self-paced center-out (CO) reaching task to 8 uniformly spaced radial targets (Fig. 1A) with its right arm inside a KINARM exoskeleton (BKIN Technologies, Ontario). Fig. 1B outlines the task structure. After training, the subject underwent neural implant surgery. Electrophysiology and data collection were performed using methods outlined in [4]. Briefly, arrays of 128 electrodes were chronically implanted in both hemispheres, targeting dorsal premotor (PMd) and primary motor (M1) cortical areas. Single and multi-unit activity were recorded and sorted online (Plexon Inc, Dallas, TX). All experimental procedures conformed to the *Guide for Care and Use of Laboratory Animals* and were approved by the University of California, Berkeley Institutional Animal Care and Use Committee.

Closed-loop BMI control was implemented by using PlexNet (Plexon Inc, Dallas TX) to stream neural data on a local intranet from the Plexon system to a dedicated computer running Matlab (The Mathworks, Natick, MA). The KF was used for all decoding, using a 100ms bin width. Neural ensembles of 24-33 units from the left hemisphere were used. Consistent with a paralyzed patient model, decoders were seeded using neural activity recorded during observation of a

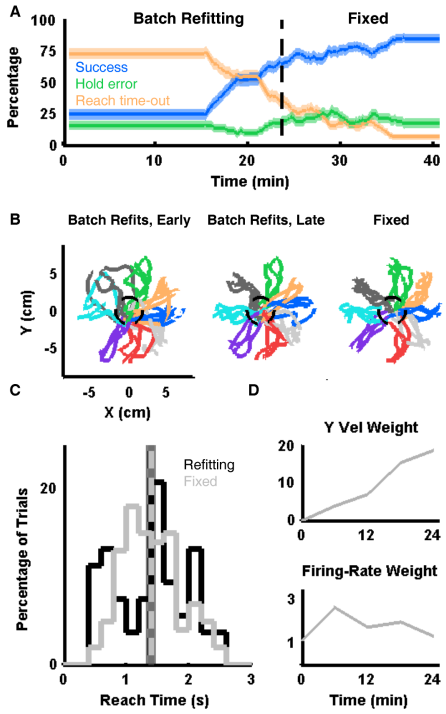

Fig. 2. Performance of batch CLDA for one representative session where 4 batch refittings were performed. (A) Task performance rates (quantified using a 100-trial sliding average). Only successfully initiated trials were analyzed. Note that the decoder was updated between the batch refitting and fixed session. (B) Successful reach trajectories (4 per direction) at the beginning and end of decoder adaptation (left and center), and immediately after the decoder was held fixed (right). (C) Distribution of reach-times for the first 100 trials during decoder adaptation, and after fixing the decoder. (D) Progression of decoder weights (C matrix terms) for one representative neuron, for Y velocity (top) and baseline firing-rate states (bottom).

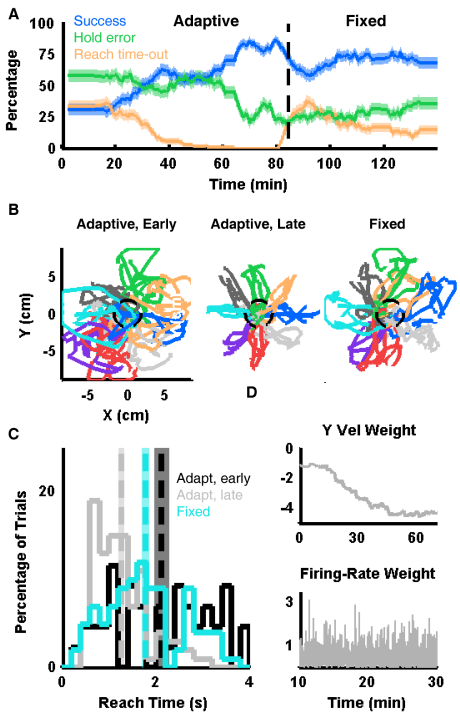

Fig. 3. Performance of online CLDA for one representative session. Format as in Fig. 2. (C) includes reach times for first and last 100 trials during adaptation, and the first 100 trials after fixing the decoder. Note that time-scale in the lower pannel of (D) is reduced to show more detail.

cursor moving through the CO task for 8-10 minutes (visual feedback, VFB). The subject viewed artificially generated movements (Gaussian speed profiles, straight reaches, 800ms movement duration) while sitting in a primate chair (arms at his side). Although no effort was made to constrain subject behavior during this period, the subject typically sat quietly and looked at the screen.

After a decoder was trained with VFB, the decoder was retrained during closed-loop BMI operation using either the batch or online adaptive methods (Sections III-A and III-B, respectively). During BMI, the subject's arm was at its side, unconstrained within the primate chair. Batch sizes of 6-10 minutes were used. Learning rates for the online adaptive updates were: $\rho \in [0.1, 0.2]$ and $\alpha = 0.999834979$ (corresponding to a "half-life" decay of 7 min). Task requirements were held fixed throughout the decoder refittings (target sizes 1.5-1.8cm radii, hold times 300-400ms, reach time-limit 2.5-4s). Adaptation was performed until the subject achieved control to all reach targets and performed approximately $\geq$ 6-10 trials per minute. Algorithms were tested on separate days. No effort was made to conserve units across days.

V. RESULTS

Figures 2 and 3 summarize typical session performance of the batch and online decoder adaptation methods, respectively. Despite starting from poorly performing decoders, both methods were successful in improving performance. For the batch method, multiple batch sessions were required to achieve adequate task performance (the example session required 4 6-minute refitting sessions). Task performance (Fig. 2A) for the batch method shows very non-linear ramps due to the discrete decoder updates, and little improvement was seen until after 2 batches. By comparison, the online adaptive method shows slow, but steady, improvements in task performance (Fig. 3A). Both methods improved reach kinematics significantly, with reaches becoming straighter, faster, and more stereotyped (Figs. 2B,C and 3B,C).

The batch method showed faster overall improvement. Approximating a linear slope of improvement (using start and end performance calculated with a 100-trial sliding window), batch improved at a rate of 1.34 ± 0.39 %/minute (mean $\pm$ standard deviation) ($n = 5$), while the online method improved at 0.71 ± 0.11 %/minute ($n = 2$). However, because the online method updates more frequently, the subject sees improvement more immediately. The batch method requires the subject to persist with a poorly performing decoder during the entire batch session (6-10 minutes), which can reduce the subjects level of task engagement. In the first 10 minutes, the subject attempted to initiate nearly twice as many trials in the online sessions as compared to batch (batch: 56 ± 21 ($n = 5$), online: 96.5 ± 4.9 ($n = 2$)).

The ultimate goal of decoder adaptation is to create a decoder the subject can easily control to perform many tasks. In many cases, particularly unstructured tasks, the

subject's intended kinematics cannot always be inferred. Thus, it is important that a subject is able to maintain task performance after decoder adaptation has stopped. To test this, after the subject achieved adequate performance ($\geq$6-10 trials/min), the decoder was held fixed as the subject continued to perform the task. The batch method yielded stable task performance (Fig. 2A) and reach kinematics (Fig. 2B). The online adaptive method, however, showed significant degradation in performance after the decoder was fixed. While performance rates only show a slight drop, movement quality is significantly reduced in the form of more variable trajectories and slower reaches (Fig. 3B,C). Inspection of the decoder weights across time (Fig. 3D) give insight into one possible reason for this drop in performance. Kinematic coefficients (e.g. y-velocity) for neurons show clear trends over time, though somewhat noisy. Coefficients for the baseline firing rates, which would be expected to remain similar across the session, show high frequency oscillations. This suggests the online adaptive method may be overfitting the data in time, thus reducing the decoder's performance once fixed.

VI. DISCUSSION

We have demonstrated that CLDA methods operating on different time-scales can improve online BMI performance, thus showing these methods' potential for producing high-performance neuroprostheses for patients. Despite starting with initial decoders that exhibited poor performance (20-30%), closed-loop adaptations allowed the subject to achieve performance over 80%. Furthermore, the subject was able to generate straight, rapid ($\leq$ 1.5s) movements to all targets, approaching performance levels of natural arm movements.

Our results also highlight an important observation about using CLDA when starting with poor initial decoders. Previous work using batch decoder adaptation that started with moderatel closed-loop performance required only one batch update to significantly improve performance [6], [7]. However, our results indicate that multiple batch updates are required when starting with significantly reduced performance. This suggests that CLDA effectively boot-straps BMI performance, and that significant improvements do not occur until some minimum level of performance is achieved (see the non-linear performance slope in Fig. 2). This phenomenon may be due, in part, to the fact that poor initial decoder performance can reduce subject engagement. Furthermore, in response to low performance, subjects may alter their control strategies more frequently, effectively adding variance to the data and making it more difficult to update the decoder. The fact that subject engagement increased significantly with online updates indeed suggests that decoder improvement may involve a positive-feedback mechanism.

To be useful in practical situations, an ideal closed-loop adaptation algorithm should: 1) rapidly improve performance, 2) actively engage the subject in the task, and 3) update the decoder so that the subject can maintain performance after adaptation is stopped. While both the batch and online adaptive methods do indeed improve performance, neither method fulfills all of these requirements.

Batch adaptation achieves faster improvement and yields stable performance after the decoder is fixed, but its low decoder update frequency can reduce subject engagement when starting with with poorly performing decoders. Online adaptation also improves performance (albeit more slowly), and its high decoder update frequency helps keep the subject continually engaged. However, it overfits decoder parameters on short temporal scales, leading to deterioration of performance after adaptation stops. While the learning rate could be reduced to avoid overfitting, this would be at the cost of slowing performance improvements during adaptation.

Given that neither algorithm is optimal, we instead propose a new, hybrid approach. Estimating decoder parameters with batches of data avoids temporal overfitting, but reduces the decoder update frequency. Using small batches reduces the accuracy of parameter estimates, and thus is not a feasible solution on its own. Instead, one could use small (1-2 minute) batches to modify decoder weights using update equations similar to those of the online adaptive method. This hybrid algorithm may combine the observed benefits of batch and online adaptive methods, yielding a more optimal approach to rapidly and reliably boost decoder performance.

REFERENCES

[1] D. M. Taylor, S. I. H. Tillery, and A. B. Schwartz, "Direct cortical control of 3d neuroprosthetic devices," *Science*, vol. 296, no. 5574, pp. 1829–1832, 2002. [Online]. Available: http://www.sciencemag.org/content/296/5574/1829.abstract

[2] J. M. Carmena *et al.*, "Learning to control a brain-machine interface for reaching and grasping by primates," *PLoS Biol*, vol. 1, no. 2, p. e42, 10 2003.

[3] S. Koyama *et al.*, "Comparison of braincomputer interface decoding algorithms in open-loop and closed-loop control," *J Comp Neurosci*, vol. 29, pp. 73–87, 2010, 10.1007/s10827-009-0196-9.

[4] K. Ganguly and J. M. Carmena, "Emergence of a stable cortical map for neuroprosthetic control," *PLoS Biol*, vol. 7, p. e1000153, 07 2009.

[5] L. Shpigelman, H. Lalazar, and E. Vaadia, "Kernel-arma for hand tracking and brain-machine interfacing during 3d motor control," *in Proc. Neural Information Processing Systems*, pp. 1489–1496, 2008.

[6] V. Gilja *et al.*, "A high-performance continuous cortically-controlled prosthesis enabled by feedback control design," *2010 Neuroscience Meeting Planner*, San Diego, CA, 2010.

[7] ——, "High-performance continuous neural cursor control enabled by feedback control perspective," *Fronteirs in Neuroscience. Conference abstract: Comp and Sys Neurosci (COSYNE)*, Salt Lake City, UT, 2010.

[8] J. P. D. Wilson Truccolo, Gerhard M. Friehs and L. R. Hochberg, "Primary motor cortex tuning to intended movement kinematics in humans with tetraplegia," *J Neurosci*, vol. 28, no. 5, pp. 1163–1178, January 2008.

[9] A. H. F. Aaron J. Suminski, Dennis C. Tkach and N. G. Hatsopoulos, "Incorporating feedback from multiple sensory modalities enhances brainmachine interface control," *J Neurosci*, vol. 30, no. 50, p. 16777, 2010.

[10] W. Wu *et al.*, "Bayesian population decoding of motor cortical activity using a Kalman filter." *Neural Computation*, vol. 18, no. 1, pp. 80–118, January 2006.

[11] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*. Wiley, March 1996.

[12] S. Dangi, S. Gowda, R. Héliot, and J. M. Carmena, "Adaptive kalman filtering for closed-loop brain-machine interfaces," *in Proc. of 5th International IEEE EMBS Conf on Neural Eng*, Cancun, Mexico, 2011.