Closed-Loop Decoder Adaptation on Intermediate Time-Scales Facilitates Rapid BMI Performance Improvements Independent of Decoder Initialization Conditions

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

***Abstract*—Closed-loop decoder adaptation (CLDA) shows great promise to improve closed-loop brain–machine interface (BMI) performance. Developing adaptation algorithms capable of rapidly improving performance, independent of initial performance, may be crucial for clinical applications where patients have limited movement and sensory abilities due to motor deficits. Given the subject-decoder interactions inherent in closed-loop BMIs, the decoder adaptation time-scale may be of particular importance when initial performance is limited. Here, we present SmoothBatch, a CLDA algorithm which updates decoder parameters on a 1–2 min time-scale using an exponentially weighted sliding average. The algorithm was experimentally tested with one nonhuman primate performing a center-out reaching BMI task. SmoothBatch was seeded four ways with varying offline decoding power: 1) visual observation of a cursor ($n = 20$), 2) ipsilateral arm movements ($n = 8$), 3) baseline neural activity ($n = 17$), and 4) arbitrary weights ($n = 11$). SmoothBatch rapidly improved performance regardless of seeding, with performance improvements from 0.018 ± 0.133 successes/min to >8 successes/min within 13.1 ± 5.5 min ($n = 56$). After decoder adaptation ceased, the subject maintained high performance. Moreover, performance improvements were paralleled by SmoothBatch convergence, suggesting that CLDA involves a co-adaptation process between the subject and the decoder.**

***Index Terms*—Adaptive control, brain–machine interfaces (BMIs), closed loop systems, motor cortex.**

Manuscript received June 13, 2011; revised November 09, 2011; accepted January 04, 2012. Date of publication February 03, 2012; date of current version July 03, 2012. This work was supported in part by the National Science Foundation Graduate Research Fellowship (ALO), in part by the American Heart Association predoctoral fellowship (ALO), in part by the Defense Advanced Research Projects Agency contract N66001-10-C-2008 (JMC), and in part by the National Science Foundation under Grant CBET-0954243 (JMC).

A. L. Orsborn is with the University of California, Berkeley—University of California, San Francisco Graduate Group in Bioengineering, University of California Berkeley, Berkeley, CA 94720 USA (e-mail: amyorsborn@berkeley.edu).

S. Dangi is with the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, Berkeley, CA 94720 USA (e-mail: sdangi@eecs.berkeley.edu).

H. G. Moorman is with the Helen Wills Neuroscience Institute, University of California Berkeley, Berkeley, CA 94720 USA (e-mail: helenem@berkeley.edu).

J. M. Carmena is with the Department of Electrical Engineering and Computer Sciences, the Helen Wills Neuroscience Institute, the Program in Cognitive Science, and the University of California, Berkeley—University of California, San Francisco Graduate Group in Bioengineering, University of California Berkeley, Berkeley, CA 94720 USA (e-mail: carmena@eecs.berkeley.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNSRE.2012.2185066

I. Introduction

BRAIN-MACHINE interfaces (BMIs) have great potential to restore function to patients with motor disabilities including amputees and those suffering from spinal cord injury, stroke, and amyotrophic lateral sclerosis. Early work has provided a solid proof of concept for BMIs, with several groups showing demonstrations of rodents [1], nonhuman primates [2]–[12], and humans [13], [14] controlling artificial actuators using neural activity. However, in order to make BMI systems widely clinically viable, significant improvements in reliability (lifetime usability of the interface) and performance (achieving control and dexterity comparable to natural movements) are needed [15], [16].

BMI systems use an algorithm (the "decoder") to translate recorded neural activity (e.g., spike trains) into a control signal (e.g., position) for an external actuator such as a computer cursor. The BMI user receives performance feedback, typically in the form of visual observation, creating a closed feedback system. Thus, BMIs allow a user to modulate their neural activity to achieve a desired goal. BMI decoders are usually created in open-loop by first recording neural activity as a subject makes movements [2]–[6], [10], [11], [17]–[19] or imagines moving [7], [12]–[14], and then training a decoder to predict these movements from the neural activity. However, open-loop decoder prediction power does not directly correlate with closed-loop performance [20], [21], suggesting that improvements in BMI performance cannot be achieved solely by finding an optimal open-loop decoding algorithm. Instead, recent work shows that significant improvements in performance can come from insights into the closed-loop BMI system, in which brain and machine adaptation play pivotal roles. For instance, Ganguly and Carmena [10] showed that when a subjects practiced BMI control with a fixed decoder, they learned a stable neural representation of the decoder, and the development of these stable representations paralleled improvements in control. In other words, the brain can adapt to improve performance. Other researchers have taken the opposite approach, investigating methods of closed-loop decoder adaptation (CLDA) to improve performance [3], [17]–[19], [22], [23]. These studies show that closed-loop BMI performance can be significantly improved by using known or inferred task goals, or evaluative feedback, during closed-loop BMI control to modify the decoder.

CLDA algorithms typically have two components: 1) a method to infer a subject's intended movement goals during closed-loop control, and 2) a rule to update the decoder's parameters. One particularly interesting aspect of candidate decoder update algorithms is the time-scale on which they update the decoder. Gilja *et al.* [17], [18] used a batch-based algorithm that applies one discrete decoder update 10–15 min after the initial seeding, while Shpigelman *et al.* [19] used a real-time update rule that adjusts the decoder at every decoder iteration. Given that closed-loop BMI performance involves an inherent interplay between the subject and the decoder, the rate at which the decoder changes will likely influence performance and the algorithm's ability to improve control. Moreover, this time-scale of adaptation may be paramount in situations where initial closed-loop performance may be severely limited, such as clinical applications for patients that cannot enact natural movement because of spinal cord injury or other neurological disorders [24]. It is thus worthwhile to identify the most appropriate time-scale of decoder adaptation to yield efficient CLDA algorithms that rapidly and robustly improve BMI performance regardless of initial closed-loop performance.

Previous work [24] suggests that an algorithm using an intermediate (1–2 min) timescale may be ideal for situations with limited initial performance. Here, we present a new CLDA algorithm called SmoothBatch, which updates the decoder on this timescale. This algorithm implements a sliding average of decoder parameters estimated on small batches of data. SmoothBatch uses the method developed by Gilja *et al.* [17], [18] to infer the subject's intended kinematics in a center-out task. We present experimental validation using data from one nonhuman primate subject. We also explore the algorithm's ability to improve closed-loop BMI performance independent of the initial decoder performance (seed) by comparing SmoothBatch's performance using four different decoder seeding methods: 1) visual observation of cursor movement, 2) ipsilateral arm movement, 3) neural activity during quiet sitting, and 4) arbitrary weights.

II. Methods

A. Electrophysiology

One adult male rhesus macaque (macaca mulatta) was used in this study. The subject was chronically implanted with microwire electrode arrays for neural recording. One array of 128 teflon-coated tungsten electrodes (35 μm diameter, 500 μm wire spacing, 8×16 array configuration; Innovative Neurophysiology, Durham, NC) was implanted in each brain hemisphere, targeting the arm areas of primary motor cortex (M1) and dorsal premotor cortex (PMd). Localization was performed using stereotactic coordinates from rhesus brain anatomy [25]. Each array was positioned targeting M1, and due to the size of the array, extended rostrally into PMd. All procedures were conducted in compliance with the National Institutes of Health Guide for Care and Use of Laboratory Animals and were approved by the University of California, Berkeley Institutional Animal Care and Use Committee.

Unit activity was recorded using a combination of two 128-channel MAP and OmniPlex recording systems (Plexon Inc, Dallas, TX). Single and multiunit activity was sorted using an online sorting application (Plexon, Inc, Dallas, TX), and only neural activity with well-identified waveforms were used for BMI control.

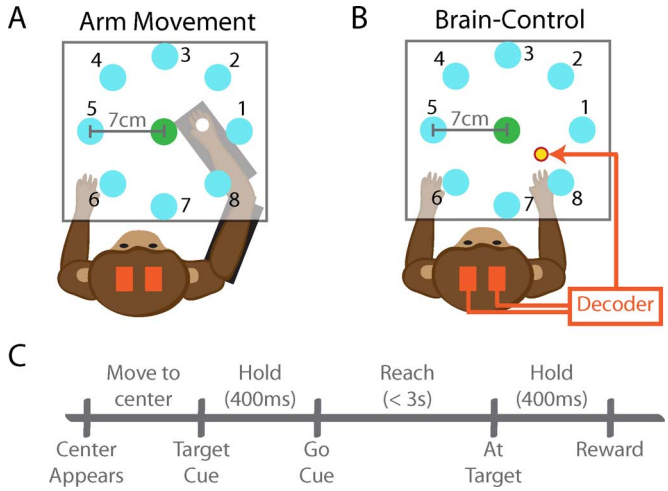

Fig. 1. Behavioral task. (A) and (B) show a top-view of the experimental setup. The nonhuman primate subject sat in a primate chair viewing a 2-D center-out task projected in front of them. Initial training (and ipsi decoder seeding) was performed with the animal performing the task with his arm inside a 2-D exoskeleton (A). In BMI, the task was performed with the animal's arm removed from the exoskeleton and the cursor position controlled via BMI predictions (B). (C) Time-line of task events.

B. Behavioral Task

The subject was trained to perform a self-paced delayed 2-D center-out reaching task to eight targets (1.7 cm radius) uniformly spaced about a 14-cm-diameter circle. The animal sat in a primate chair, head restrained, and observed reach targets displayed via a computer monitor projecting to a semi-transparent mirror parallel to the floor. Fig. 1 shows an illustration of the task setup and trial time-line. Trials are initiated by moving the cursor to the center target and holding for 400 ms. Upon entering the center, the reach target appears. After the center-hold period ends, the subject is cued to initiate the reach (via target flash), after which he must move the cursor to the peripheral target within a given 3 s time-limit and hold for 400 ms to receive a liquid reward. If the subject makes an error, defined as failure to hold at the center or target, or failure to reach the target within the time-limit, the trial is aborted and must be restarted. The subject has an unlimited amount of time to enter the center target to initiate a trial. Reach targets were presented in a block structure, with pseudo-randomized order within each block of eight targets.

Initial task-training was conducted with the animal making arm movements. The arm was placed in a 2D KINARM exoskeleton (BKIN Technologies, Kingston, ON, Canada), which constrained movements to the horizontal plane parallel to and just under the reach-target display. A cursor co-localized with the center of the subject's hand was displayed on the screen to provide visual feedback of hand location. During BMI operation, the subject performed the center-out task by moving the cursor under neural control. The subject's arm were removed from the KINARM and confined within the primate chair. The

subject was proficient (overtrained) in the center-out task with arm movements before BMI experiments commenced.

C. BMI Decoder and Implementation

Online closed-loop BMI control was implemented using a Kalman filter (KF) decoder [26], a common decoding algorithm choice in BMI [27]. 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 (1) and state observation models (2)

$$x_t = Ax_{t-1} + w_t \tag{1}$$

$$y_t = Cx_t + q_t \tag{2}$$

where $w_t \sim N(0, W)$ and $q_t \sim N\ (0, Q)$ are noise terms. Matrices A, W, C, and Q are estimated from the training data (see Section II-D below), and the KF provides a framework to recursively estimate the current cursor state (x_t) based on estimates of the past states $\{x_{t-1}, x_{t-2}, \ldots\}$ and observed neural activity $\{y_t, y_{t-1}, y_{t-2}, \ldots\}$ in real-time. We refer the reader to [26] and [27] for the filter time update equations of the KF. We used a position-velocity KF, as in [17] and [18], operating in endpoint coordinates. Thus, $x_t = [p_x\ p_y\ v_x\ v_y\ 1]\mathrm{T}$, where p_x, p_y, v_x, and v_y are the endpoint position (x and y components) and velocity (x and y components), respectively. The constant 1 term accounts for nonzero mean observations y_t (i.e., the neurons' baseline firing-rates).

Online BMI control was implemented using PlexNet (Plexon Inc., Dallas, TX) to stream neural data on a local intranet from the Plexon recording system to a dedicated computer running Matlab (The Mathworks, Natick, MA). Neural firing rates were estimated with a 100 ms bin width. Neural ensembles of 16–36 (26.5 ± 4.45; mean $\pm$ STD) neurons were used. Units were selected only based on waveform quality.

D. Decoder Seeding

Decoders were initialized in four ways: 1) using neural activity recorded as the subject passively viewed a cursor moving through the center-out task (visual feedback; VFB; $n = 20$), 2) using neural activity recorded during ipsilateral arm movements (ipsi; $n = 8$), 3) using neural activity recorded during quiet sitting (*baseline*; $n = 17$), and 4) arbitrary seeding by shuffling weights of past (SmoothBatch-adapted) decoder weights (*shuffled*; $n = 11$).

The state-transition model (A and W matrices in (1), describing cursor dynamics) were defined to model the system physics, with position components set as the integral of velocity, as in [17] and [18]

$$A = \begin{bmatrix} 1 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & dt & 0 \\ 0 & 0 & a_v^{xx} & a_v^{xy} & 0 \\ 0 & 0 & a_v^{yx} & a_v^{yy} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$W = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_v^{xx} & w_v^{xy} & 0 \\ 0 & 0 & w_v^{yx} & w_v^{yy} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where dt is the step-size of the Kalman filter update. The velocity state transition model (A_v and W_v) were fit using their maximum-likelihood estimates

$$A_v = \begin{bmatrix} a_v^{xx} & a_v^{xy} \\ a_v^{yx} & a_v^{yy} \end{bmatrix} = V_2 V_1^T \left(V_1 V_1^T\right)^{-1}$$

$$W_v = \begin{bmatrix} w_v^{xx} & w_v^{xy} \\ w_v^{yx} & w_v^{yy} \end{bmatrix} = \frac{1}{N-1}(V_2 - AV_1)(V_2 - AV_1)^T$$

where N is the total number of measured time-points, and V_1 and V_2 matrices are formed by tiling recorded velocity kinematics (v_x and v_y states) for times $[1,\ N-1]$ and $[2,\ N]$, respectively. Since we aimed to create a BMI control that mimics natural arm movements, A_v and W_v were fit using a 30 min data set of arm-movements from the center-out task for all seeds.

For seeding methods 1–3, the observation model (C and Q matrices in (2), describing neural activity's relation to cursor movement) were fit using their maximum-likelihood estimate

$$C = YX^T(XX^T)^{-1} \tag{3}$$

$$Q = \frac{1}{N}(Y - CX)(Y - CX)^T \tag{4}$$

where the Y and X matrices are formed by tiling recorded neural activity and kinematics (full state vector), respectively. Neural and kinematic data were collected for 8 min. For VFB seeding, a visual cursor was moved through artificially generated trajectories (straight trajectories, Gaussian speed profiles, 800 ms reach duration) to perform the center-out task. The subject viewed the cursor movement while seated in the primate chair, receiving pseudo-random rewards on 25% of all trials. While no effort was made to constrain subject behavior, the animal typically sat quietly and looked at the display. Recorded neural activity and the artificially generated cursor kinematics were used for decoder estimation. *Baseline* seeding was performed using the same protocol as VFB, but with the display turned off so that the subject did not see anything. Ipsi decoder seeds were created with neural data ipsilateral to the arm kinematics recorded as the subject performed the center-out task with his arm. For *shuffled* seeds, the observation models were constructed by taking the weights of an existing decoder (previously optimized using SmoothBatch CLDA; see below) from prior experimental sessions and shuffling the weight assignments for each unit.

E. Closed-Loop Decoder Adaptation Algorithm: SmoothBatch

Once a seed decoder was created, CLDA was used to improve performance. The algorithm used, SmoothBatch, updates the decoder on an intermediate (1–2 min) time-scale. The subject performed the center-out task under BMI control as the algorithm updated the decoder. The subject's intended kinematics were inferred from the observed BMI performance using the technique developed by Gilja *et al.* [17], [18], which assumed that the subjects intends to reach straight to the target at all times and rotates observed cursor velocities to point towards the current task target. The observed neural activity and intended kinematics were collected for a short (1–2 min) interval. Each batch of data was used to construct a new estimate of the C and Q

matrices [using (3) and (4)], $\hat{C}$ and $\hat{Q}$. The BMI decoder was then updated using an exponentially weighted moving average

$$C^{(i+1)} = \alpha C^{(i)} + (1-\alpha)\hat{C} \quad (5)$$

$$Q^{(i+1)} = \beta Q^{(i)} + (1-\beta)\hat{Q} \quad (6)$$

where i indexes discrete decoder iterations α and $\beta \in (0,1)$ determine the speed of adaptation. We report α and β in terms of the half-life of the update process—i.e., how long it takes for the influence of a $\hat{C}$ estimate in the BMI decoder to reduce by half. The half-lives for C and Q, denoted by h_c and h_q, are related to α and β on the open interval (0 1) as follows:

$$\alpha^{(h_c/b)} = 1/2 \text{ and } \beta^{h_q/b} = 1/2$$

where b is the batch size. Note that α and β values depend upon both the half-life and batch size.

To avoid conducting a vast parameter search of both batch-size and half-life experimentally, SmoothBatch was first implemented in a BMI simulator that utilizes an empirically verified model of the user's learning process during closed-loop BMI operation [28]. Preliminary results from the simulator allowed were used to narrow down the search space, which was then explored in experiments. Batch sizes of 40–100 s and C and Q half-lives of 90–300 s were tested experimentally. Rough optimization showed that batch sizes of 60–100 s and half-lives of 90–210 s produced the most rapid performance improvements. All presented data use parameters within this range; the majority $(n = 46)$ use an 80 s batch and 120 s half-life. In all cases, C and Q half-lives were set to be equal so that these matrices were updated at the same time. Also note that the A and W matrices (which parameterize the cursor dynamics model) were held fixed. As soon as a new decoder update was calculated, it was used for subsequent BMI predictions.

F. Data Analysis

1) Behavioral Performance Metrics: Only successfully initiated trials (i.e., entering the center and successfully holding until the go-cue) were considered for analysis, allowing for three possible outcomes: a successful reach, a target-hold error, or a reach time-out. The subject would typically leave the center early (a false-alarm) on approximately 10% of all attempts to initiate a reach, both during neural and arm control. The task structure did not penalize these errors, likely leading to the high false-alarm rate. Thus, they were excluded from analysis. Task behavior was quantified by analyzing trial outcomes both by percentage and event rates. The success, reach time-out, and error percentages over time were calculated using a sliding average (75 trial window) to estimate the evolution of performance within the session. The corresponding rates for these different trial outcomes were quantified by binning the event times (60 s wide, nonoverlapping bins). The event percentage metric provides an estimate of overall task performance, while the event rates provide more temporal information about task performance (e.g., periods of inactivity when the subject is unable to initiate a trial will not be reflected in the trial-based percentage). A threshold of 8 successes/min was used to assess SmoothBatch's improvement time. The task required the subject to hit each presented reach target before moving to the next target. Therefore, achieving 8 successes/min typically corresponds to the subject successfully acquiring all targets, demonstrating control across the entire workspace. Reach times were quantified as the time between the cursor leaving the center target and entering the peripheral target or occurrence of a time-out error. Evolution of trajectory quality were estimated using metrics adopted from literature for pointing device evaluation [29] and previously used to evaluate BMIs [14]. Average movement error (ME) and movement variability (MV) around the task-relevant axis (i.e., perpendicular to the reach target) were quantified. ME and MV are defined as

$$\text{ME} = \frac{\sqrt{\sum |y_i|}}{n} \quad \text{and} \quad \text{MV} = \sqrt{\frac{\sum (y_i - \bar{y})^2}{n}}$$

where y_i is the perpendicular distance from the task-axis at time-point i and $\bar{y}$ is the mean of y across the trajectory. ME and MV were computed for each trajectory using the time from leaving the center to arriving at the distal target or occurrence of a time-out error.

2) Decoder Metrics: The change in KF matrices C and Q were computed as the decoder was updated using SmoothBatch. The Frobenius norm of the differences of consecutive update step were used to look at overall matrix adaptation. The velocity terms (in x and y directions) in the C matrix and their evolution in time were also analyzed. While the Kalman filter decoder estimated both position and velocity, analysis of the Kalman gains obtained showed that neural activity makes the strongest contributions to the cursor velocity, making these terms of particular importance for the decoder. Our Kalman filter assumes

$$y_i = C_{i,1}p_x + C_{i,2}p_y + C_{i,3}v_x + C_{i,4}v_y + C_{i,5}$$

where y_i is the firing rate of unit i, and $C_{(i,1)}$ represents the $(i,1)$ entry of the C matrix. Ignoring position terms, this is a standard velocity tuning model [30]. A unit's velocity preferred-direction (PD) and modulation-depth (MD) are given by

$$\text{PD}_i = \tan^{-1}\left(\frac{C_{i,4}}{C_{i,3}}\right)$$

$$\text{MD}_i = \sqrt{C_{i,3}^2 + C_{i,4}^2}.$$

To assess how the decoder direction-tuning model evolved, the mean change in PD across neurons was quantified. More strongly tuned neurons (larger MD) contribute more to BMI performance than nontuned units (small MD). Hence, changes in PDs of neurons with high MD more directly influence decoder performance. To capture this, the magnitude of PD change for a given unit was weighted by that unit's MD in the final decoder. The average weighted change in PD for the full decoder $(\Delta w\text{PD})$ was calculated by averaging this weighted PD change across units$\Delta w\text{PD}$ is defined as

$$\Delta w\text{PD}(t) = \frac{1}{M}\sum_{i=1}^{M}\left(\frac{\text{MD}_i(T)}{\max_i(\text{MD}(T))} |\text{PD}_i(t) - \text{PD}_i(t-1)|\right)$$

where M is the total number of units in the decoder, and $t \in [1,T]$ indexes each decoder during adaptation.

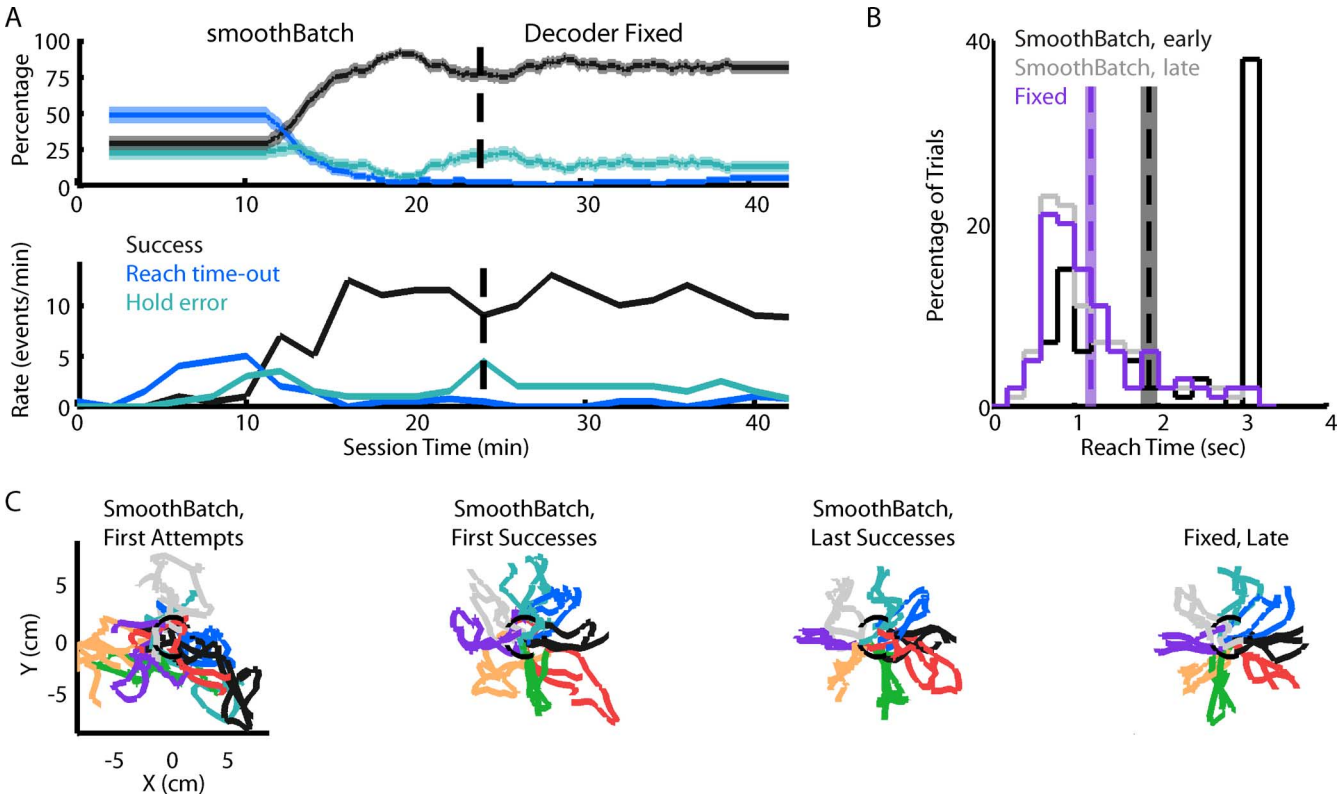

Fig. 2. BMI task performance improvements using SmoothBatch closed-loop decoder adaptation for one representative *baseline* seeded session. (A) Sliding average (75 trial window) of task performance rates (top) and binned event hit-rates (bottom, 120 s bin width) during SmoothBatch adaptation and upon fixing the decoder. (B) Distribution of reach-times for the first and last 100 reaches during SmoothBatch operation, and the first 100 reaches upon fixing the decoder. Dashed lines and shaded regions indicate mean and standard error of the mean, respectively, for each. (C) Progression of reach trajectories during the session. Three reaches to each target are shown for the first attempted reaches of the session (left), first successful reaches during SmoothBatch (left middle), last successful reaches during SmoothBatch (right middle), and the last successful reaches with a fixed decoder (right).

3) Offline Seed Decoder Prediction Power: The offline prediction power of seed decoders were computed using the R^2 correlation coefficient between measured kinematics and predicted kinematics (x and y components of position and velocity). In ipsi seeds, measured kinematics corresponded to observed arm movements. In VFB and *baseline* seeds, the display cursor's kinematics were used as the measured kinematics. Note that in *baseline* seeding, the cursor was not viewed by the subject. Finally, for shuffled decoders, the decoder was used to predict kinematics using neural activity recorded during a VFB condition recorded within the same session, and those predictions were compared with the display cursor kinematics.

III. Results

A. SmoothBatch BMI Performance Improvements

SmoothBatch CLDA rapidly and reliably improved closed-loop BMI performance, despite being seeded with decoders constructed in the absence of contralateral arm movements. Fig. 2 shows the evolution of task performance for a representative *baseline* seed session. As seen in the events per minute [quantified with a 120 s bin-width to reduce behavioral noise; Fig. 2(A)], the subject was not readily able to perform the task with the seed decoder (only a few trials were initiated). After a few minutes (1–2 decoder updates), the subject was able to initiate trials but with limited control as shown by the increased rate of reach time-out events [Fig. 2(A)] and the highly irregular reach trajectories in the first attempted reaches [Fig. 2(C)]. Performance improved gradually until approximately 10 min, when success percentage and rate both markedly improved.

These rapid, nonlinear performance improvements were observed across all sessions (including all decoder seed types). Initial performance started at 0.018 ± 0.133 successes/min and exceeded 8 successes/min (which roughly corresponds to attaining successful reaches to all targets) within 13.1 ± 5.5 min. Performance increased up to maximum rates of 14.3 ± 1.37 successes/minute (maximum $88.04\% \pm 5.3\%$ trial success percentage) within 20.75 ± 5.93 min (all $\text{mean} \pm \text{SD}$; $n = 56$, all seed types). Reach trajectories [Fig. 2(C)] showed improvement between the first success and late in SmoothBatch adaptation, becoming more stereotyped. Similarly, reach times showed a marked reduction from the first to last 100 trials during SmoothBatch, and remained low after the decoder was held fixed after adaptation ceased [Fig. 2(B)]. Across all sessions, SmoothBatch significantly improved reach trajectory kinematics. The mean reach times, ME, and MV for the last 100 trials during SmoothBatch were significantly smaller than those of the first 100 trials in the session (Wilcoxon paired test, $p > 0.05$). At the end of SmoothBatch adaptation, the subject achieved average reach times of 1.23 ± 0.16 s, ME of 0.771 ± 0.088 cm, and MV of 0.593 ± 0.067 cm (all $\text{mean} \pm \text{SD}$; $n = 56$, all seed types).

As seen in the example session Fig. 2, performance typically improved gradually, requiring multiple batch updates before sufficient task performance was achieved. Many decoder seeds, particularly arbitrary seeds like *baseline* and *shuffled*, yielded decoders the subject could not readily use—even limiting the ability to initiate trials. On average, 1.23 ± 1.1 batch updates were required before the subject successfully initiated a trial, 3.03 ± 2.1 updates occurred before the first successful reach, and 7.43 ± 3.6 updates occurred before the subject was able to successfully reach all eight targets. SmoothBatch improved the decoder gradually and did not jump to a high-performance solution in a single iteration.

The ultimate goal of CLDA is to find a decoder that will allow the subject to readily gain proficient control of the BMI to perform a variety of tasks. In many situations, particularly unstructured tasks, it may not always be possible to infer the user's movement goals. Hence, it is highly desirable for task performance to be maintained after decoder adaptation has ceased. To test this, after the subject attained adequate performance ($\geq$10 trials/min, approximated by the experimenter), SmoothBatch adaptation was stopped and the subject performed the BMI task with a fixed decoder. As seen in Fig. 2, the subject was able to maintain high task performance and accurate reaches after the decoder was held fixed. Across all sessions, no significant changes in performance were found after fixing the decoder. Because the subject used fixed decoders for varying periods of time across sessions, behavioral measures were compared between the final moments of SmoothBatch adaptation (last estimate of successes per minute; success percentage for the last 75 trials) and early fixed-decoder performance (successes per minute for the first 15 min; success percentages for the first 100 initiated trials). Neither the successes per minute nor the success percentages were significantly different between the end of SmoothBatch adaptation and fixing the decoder (Wilcoxon paired test, $p = 0.733$ and $p = 0.1$, respectively; $n = 54$, two sessions with insufficient data with a fixed decoder were excluded from analysis). Furthermore, mean reach times for the last 100 trials during SmoothBatch showed no significant difference from that of the first 100 trials with the decoder fixed (Wilcoxon paired test, $p = 0.55$). Mean ME and MV, however, showed a small (approximately 12%), significant increase between the last 100 trials in SmoothBatch and the first 100 trials with a fixed decoder (Wilcoxon paired test, $p < 0.05$), suggesting a very slight drop in trajectory precision.

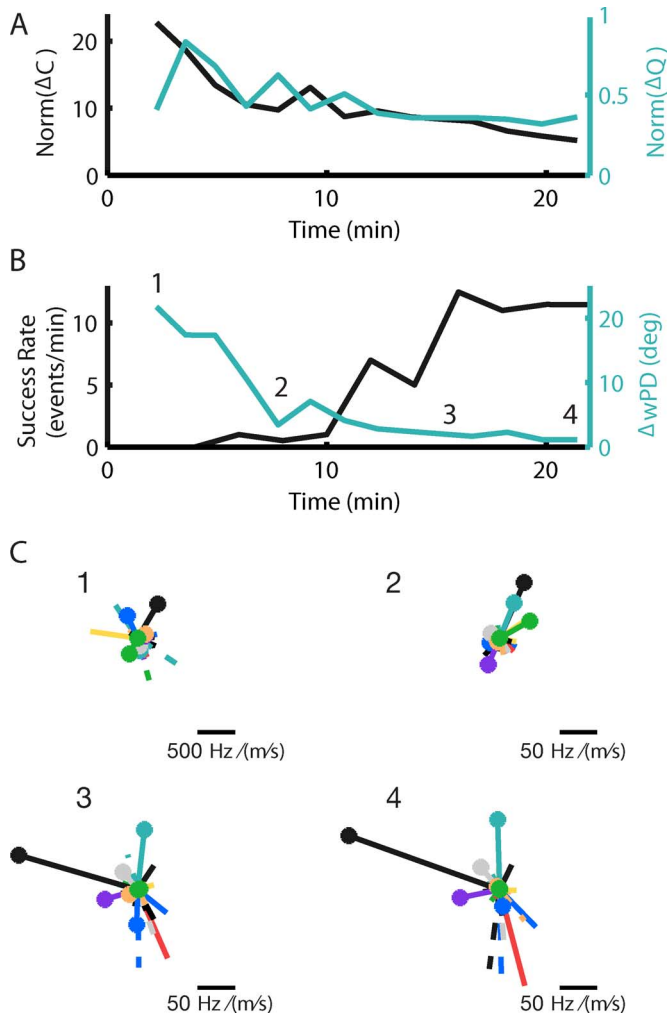

Fig. 3. Kalman filter decoder evolution during SmoothBatch closed-loop decoder adaptation for one representative session seeded with baseline activity (same session as shown in Fig. 2). (A) The matrix norm of changes in C and Q matrices between decoder updates converges as SmoothBatch continues. (B) The mean change in units' velocity preferred directions weighted by relative tuning strength ($\Delta w\mathrm{PD}$) also converges. The convergence of the velocity-tuning solution parallels behavioral performance improvements. (C) Velocity tuning decoder evolution for example time-points (1–4, indicated in B). The tuning of each unit is represented in polar coordinates, where the line length indicates modulation-depth and its orientation shows the preferred direction. Note that 1 (upper left) is on a different scale than 2–4.

B. Kalman Filter Decoder Evolution During SmoothBatch

The SmoothBatch algorithm showed convergence towards a stable decoder solution during adaptation. Fig. 3(A) displays the Frobenius norm in the element-wise change in consecutive C and Q decoder matrices across time, showing that the change rapidly decreased for both matrices. This convergence was also seen in the velocity-weights in C, with the preferred directions assigned to each unit stabilizing over time. Fig. 3(B) shows the average magnitude of PD change weighted by neuron modulation depth ($\Delta w\mathrm{PD}$; see Section II) during SmoothBatch adaptation. Task performance (successes/min) is overlaid, showing that convergence of the velocity PDs was strongly correlated with behavioral improvements. Polar plots of all decoder units' tuning (PD and MD) for example decoders during SmoothBatch adaptation are shown in Fig. 3(C) [corresponding time-points are indicated in Fig. 3(B)]. The initial baseline-seeded decoder assigned weak tuning to all units. Near the point of significant behavioral improvement, as the decoder weights were converging, the MDs of units had significantly increased and a few units had PDs near those of the final decoder. Once performance significantly improved [Fig. 3(C-3)], the tuning model very closely resembled that of the final decoder. Across all sessions, the average $\Delta w\mathrm{PD}$ between the decoder being used when the subject was first able to successfully reach all eight targets and the final decoder is $6.52° \pm 4.48°$, as compared to a $23.79° \pm 12.17°$ for the initial seed decoder ($\mathrm{mean} \pm \mathrm{SD}$; $n = 56$, all seed types).

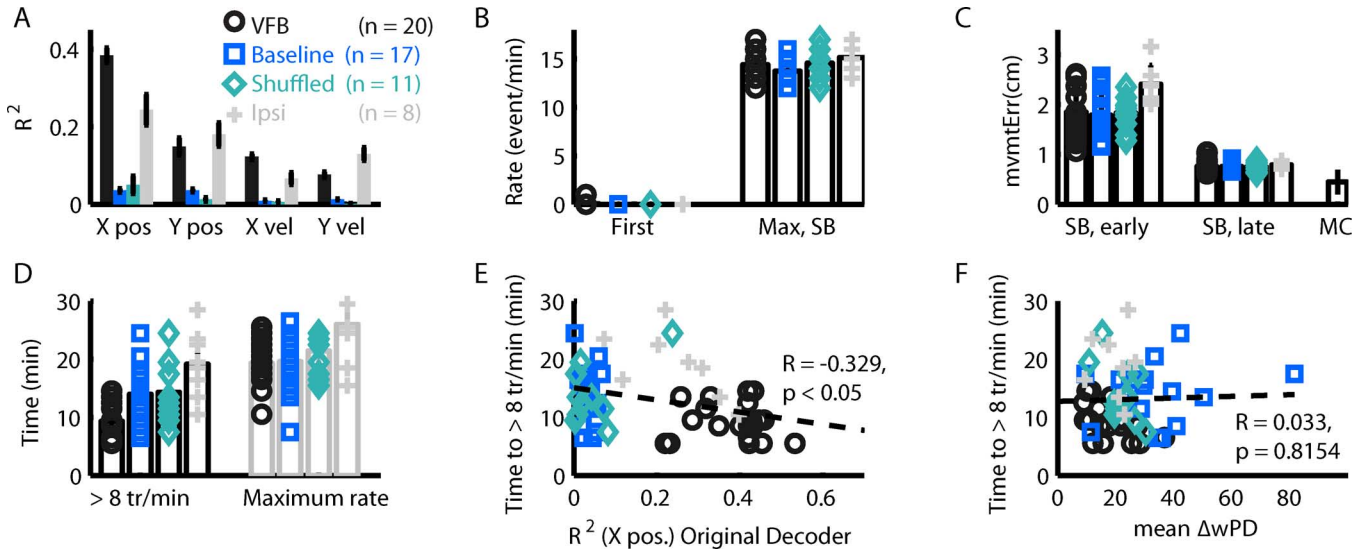

Fig. 4. SmoothBatch performance improvement across different decoder seeding types. (A) Offline decoder prediction power for endpoint position and velocity (x and y components) for all decoder seed types. Bars indicate mean across all performed sessions, error bars show standard error of the mean. (B) Initial and maximum performance rates of SmoothBatch sessions, separated by seed types. Points show individual sessions; bars and error bars show mean and standard error of the mean, respectively. (C) Average movement error (ME, see Section II) for the first (early) and last (late) 100 trials during SmoothBatch adaptation, separated by seed condition. Average ME across eight arm movement sessions (arm) are shown at right for comparison. Format as in panel C. (D) SmoothBatch performance improvement times (time for successes per minute to exceed 8 trials/min) and time to achieve the maximum performance rate for all sessions and all decoder seed types. Format as in panel C. (E) Relationship between seed decoder predictive power (R^2 for x-component of position) and time to reach threshold performance. Points are individual sessions, color-coded by seed type, and line shows linear regression ($R = -0.329$, $p < 0.05$). (F) Relationship between relative distance between seed decoder and final decoder's velocity tuning models ($\Delta w\mathrm{PD}$; see Section II) and time to reach threshold performance. Points are individual sessions, color-coded by seed type, and line shows linear regression (not significant).

C. Performance Improvement's Dependence on Decoder Seeding

SmoothBatch CLDA improved BMI performance independent of the initial decoder seeding method. Each seeding method had a varying amount of offline predictive power, as summarized in Fig. 4(A). VFB and ipsi seedings contained the most information about all kinematic variables while *baseline* and *shuffled* seeds had little to no predictive power. Despite differences in offline power of the decoders, all seeds reached similar final performance after adaptation. Fig. 4(B) compares early success rates (first estimate within the session) to the maximum rate achieved, separated by seed type. A Kruskal–Wallis analysis of variance (KW-ANOVA) showed no significant effect of seed type on maximum performance ($p > 0.05$). All seeds achieved similar reach trajectory precision as well. Fig. 4(C) shows the average ME (see Section II) for the first and last 100 trials within the SmoothBatch session, as well as average estimates from arm movements for comparison. The final reach kinematics parameters (MV, ME, nor reach time) showed a significant dependence on seed type (KW-ANOVA, $p > 0.05$).

Performance improvements did, however, occur on different time-scales depending on the initial decoder seed. Fig. 4(D) shows the amount of time it took for performance to exceed a threshold of 8 successful trials/min (left) and reach the maximum rate (right) for each session, sorted by seed decoder type. A KW-ANOVA showed that time to reach the 8 trials/min threshold depended on seed type. VFB seedings, which contained the strongest offline decoding power, improved significantly faster than all other types ($p < 0.05$). No other significant differences among groups were found. The offline R^2 power of the seed decoder showed weak correlations with time to reach threshold [Fig. 4(E)]. Pooling all data, significant correlations were only found for p_x and v_x ($R = -0.329, p = 0.014$; $R = -0.393, p = 0.003$, respectively). The time to reach threshold performance also showed no clear relation to the distance ($\Delta w\mathrm{PD}$) between the seed and final decoders [Fig. 4(F)], suggesting that algorithmic convergence/step size alone does not limit improvement. The time to achieve maximum performance, however, showed no significant differences across groups (Fig. 4(D); KW-ANOVA, $p > 0.05$) and no significant correlation with offline prediction power or $\Delta w\mathrm{PD}$ (not shown; $p > 0.05$).

As seen by the wide spread in $\Delta w\mathrm{PD}$ in Fig. 4(F), few initial seed decoders, even ones that contained significant offline decoding power, had velocity PD assignments close to those of the final SmoothBatch solution (note that on average, significant performance improvements were not seen until $\Delta w\mathrm{PD} < 6.52° \pm 4.48°$). A KW-ANOVA showed that $\Delta w\mathrm{PD}$s were significantly larger for *baseline* seeds than VFB ($p < 0.05$); no other groups showed statistically significant differences.

IV. Discussion

Use We found that the SmoothBatch CLDA algorithm, which updates on an intermediate timescale (1–2 min), was able to improve closed-loop BMI performance rapidly and robustly, independent of initial decoder seeding. The subject was readily able to achieve proficient center-out task performance ($88.04 \pm 5.3\%$ success rate, 1.23 ± 0.16 s reach times, 0.771 ± 0.088 cm ME, and 0.593 ± 0.067 cm MV) across 56 experimental sessions. Importantly, improvements were rapid (success rates exceeding

8 trials/min in an average of 13.1 ± 5.5 min), and occurred reliably in all sessions despite decoders being seeded using four different methods. Performance was also maintained upon ceasing adaptation and holding the decoder fixed, with no drop in task performance or reach times, and only slight (12%) increases in ME and MV.

Comparison across BMI studies represent a current challenge in the field [16]. Studies use incongruous task designs and different types of neural activity, obscuring direct comparisons [16]. The use of different animal models (e.g., animals with arms free to move versus partial or full movement restriction) across studies may further confound cross-study comparisons [31]. Furthermore, the field has not established standard metrics for performance evaluation, limiting the ability to quantitatively compare results. In an effort to make this work more readily comparable to future studies, we report performance metrics commonly used to evaluate pointing-devices like computer mice (MV and ME, see Section II) [29]. Comparisons to one human BMI study (not explicitly utilizing CLDA or across-session learning) reporting MV and ME metrics shows that SmoothBatch has markedly reduced mean ME and MV relative to their best results ([14] MV: 1.2 cm, SmoothBatch 0.587 cm; [14] ME: 2.3 cm, SmoothBatch: 0.759 cm). Moreover, MV and ME results for SmoothBatch approach those of natural movements (MV: 0.35 ± 0.17 cm, ME: 0.44 ± 0.20 cm; $\mathrm{mean} \pm \mathrm{SD}$ across 2025 trials in eight arm movement session).

The success of SmoothBatch is a clear testament to the power of the CLDA principle [3], [17]–[19], [22], [23]. In the limit of long batch sizes and $\alpha = \beta = 0$, Gilja *et al.*batch algorithm is a special case of SmoothBatch. The longer data batches used by Gilja *et al.* allow for more accurate parameter estimation, eliminating the need for averaging. However, waiting to collect sufficient data for a single model update greatly reduces the update frequency, thus reducing the rate at which the user sees performance improvements. In the work of Gilja *et al.*, using decoders seeded from contralateral arm movements and BMI performed during overt arm movements, their subjects were able to attain $>90\%$ task performance with the seed decoder before any CLDA intervention [17], [18]. This initial high level of performance allows for a significant improvement in reach kinematics after a single batch-based decoder update [17], [18].

SmoothBatch, however, is ideally suited to address a different problem—how can we use CLDA to generate a high-performance BMI given an initial decoder with severely limited closed-loop performance? CLDA may be a particularly useful tool in clinical applications, where many factors could limit initial closed-loop performance (see below). In these applications, collecting sufficient data for a batch-based decoder update is challenging. We recently showed that when starting from poorly performing seed decoders, as many as 3–56 min batch updates were required to produce adequate closed-loop BMI performance [24]. Furthermore, the slow update rates reduce subject motivation, since subjects were required to use poorly performing decoders for 6–10 min [24]. Though SmoothBatch sacrifices accuracy in each parameter estimation step due to a smaller batch-size than the Gilja *et al.* algorithm, the increased decoder update frequency provides the user with more rapid feedback and may facilitate faster improvement.

The progress of improvements seen in SmoothBatch also further suggest that using CLDA may involve a bootstrapping or co-adaptation process between the brain and decoder when starting from limited closed-loop BMI performance. SmoothBatch required multiple decoder update steps before performance improved significantly, progressing to gradually let the subject initiate trials, then reach a few targets, and finally reach all targets. This progression may be due in part to the exponentially weighted averaging and short data-batches used. However, the batch-sizes and half-lives used here correspond to relatively aggressive adaptation rates ($\alpha = \beta \in [0.55, 0.77]$) and on average, more than one half-life's worth of updates occurred before adequate performance was achieved. Moreover, we recently showed a similar progression using purely batch-based algorithms [24]. This suggests that creating an optimal decoder in a single update step may be highly infeasible when starting from severely limited closed-loop performance. Instead, intermediate adaptation time-scale algorithms like SmoothBatch provide the subject with more rapid feedback and gradually adapt the decoder, facilitating a co-adaptation process and yielding rapid performance improvements.

SmoothBatch uses a sliding average of model parameter estimates based on small amounts of data, which is similar to the approach used by Taylor *et al.* [3]. They also show that their co-adaptive algorithm is able to yield proficient closed-loop BMI control when seeded with randomly initialize parameters. This suggests that an intermediate time-scale adaptation approach may ideal for such applications.

One concern in using an adaptive algorithm is convergence. Our previous work [24] showed that a real-time adaptive CLDA algorithm [32] improved performance and increased subject motivation via rapid updates, but temporally overfit the data causing performance degradation after decoder adaptation ceased. Here, we show that SmoothBatch rapidly converges (Fig. 3). The subject was also readily able to maintain task performance after decoder adaptation ceased, showing that the algorithm converged towards a general task solution and avoided overfitting. C and Q matrices did exhibit small fluctuations (i.e., changes did not fully converge to zero, see Fig. 3), however these did not noticeably effect task performance, suggesting they may only be driven by behavioral and neural variability. It may also be possible to optimize batch size and half-life parameters to further improve convergence. For instance, increasing the batch size as a function of task performance could allow the algorithm to take large, rapid steps while the decoder is "far" from a high-performance decoder, and smaller steps to fine-tune the model as it nears the final solution. Additional work to fully optimize the batch and half-life parameters may also yield yet faster improvement speeds.

The idea of CLDA involving a co-adaptation between the subject and decoder is further implied by the fact that SmoothBatch's improvement rate depended upon the initial decoder seed. While SmoothBatch was able to improve performance regardless of the seed, decoders seeded with neural activity during VFB improved more rapidly than arbitrary (*baseline* and *shuffled*) or ipsilateral arm-movement (ipsi) seeds. VFB seeds had the highest level of offline decoding power, consistent with observations that imagined/viewed movements evoke motor

cortex activity [33]–[35]. Times to achieve a threshold of eight successful trials/min were weakly correlated with the decoder's offline prediction power, suggesting that the algorithm can more readily improve performance if initiated with decoder with some movement decoding power.

However, our results also suggest that offline prediction power is not the sole factor in how quickly SmoothBatch can improve performance. Ipsi decoder seeds had moderate offline prediction power, greater than that of baseline and shuffled seeds, consistent with findings of population-level representations of ipsilateral arm movements in motor cortex [36]. Yet, no statistically significant differences in improvement times were detected between ipsi, *baseline* or *shuffled* seeds. Furthermore, the times to achieve maximum performance rates were uncorrelated with offline prediction power. This is consistent with the Taylor *et al.* finding that the speed of adaptation did not differ when their co-adaptive population vector algorithm was initialized with PD estimates from arm movements or with random assignments [3].

Interestingly, we also found that on average the velocity tuning models for all seeds were "far" from that of the final decoder, despite containing varying degrees of offline prediction (Fig. 4(D); only *baseline* seeds had significantly larger $\Delta w\mathrm{PD}$ than VFB). This combined with the observation that offline prediction power alone is not indicative of SmoothBatch improvement rates, is consistent with the emerging notion that closed-loop BMI performance is not necessarily related to open-loop predictions [20], [21]. The many differences between BMI and natural movement—for instance congruent proprioceptive feedback is absent during BMI—may significantly alter the neural activity of the motor networks [12], [35]. Performance improvements in SmoothBatch did not appear to be limited by relative distance between initial and final decoders [Fig. 4(D)]. However, the difference between the subject's evoked neural activity between arm movements and closed-loop control *would* influence his performance in BMI, possibly affecting this co-adaptation process by influencing user strategy (e.g., causing frustration). We did see noticeable variability in improvement times across all decoder seed types, which could be related to subject strategies/motivation and the co-adaptation process. Additional work is needed to explore the exact mechanisms underlying performance improvements and the aspects of a seed decoder that influence final improvement. Such insights could yield further reductions in performance improvement times.

In conclusion, here we show that SmoothBatch can readily improve performance in a relatively short time, even with ill-conditioned seed decoders. Additional work to explore SmoothBatch's operation—the influence of adaptation parameters and the subject-decoder co-adaptation process—could also further improve the algorithm's operation. The ability to quickly and robustly generate high-performance BMIs independent of the subject's initial closed-loop BMI performance could be useful for clinical applications. For paralyzed patients, seeding decoders with arm movements is not feasible. Many patients will also lack proprioceptive feedback during closed-loop BMI operation, potentially limiting their performance [12]. Furthermore, noninvasive human BMI studies have shown significant inter-subject variability in the ability to volitionally modulate neural activity, which greatly impacts their ability to operate BMIs [37]. All of the decoder seeding methods presented here are compatible with many patient populations: VFB, *baseline*, and *shuffled* decoders could be created for any patient regardless of motor disability, and ipsi seedings could be used for unilaterally paralyzed patients such as stroke victims. Our work shows that SmoothBatch CLDA, an algorithm that updates the decoder on an intermediate time-scale during closed-loop operation, can reliably improve BMI performance in all of these cases.

Acknowledgment

The authors would like to thank Dr. K. Shenoy's laboratory at Stanford University for insightful discussions and comments on this manuscript.

References

[1] J. K. Chapin, K. A. Moxon, R. S. Markowitz, and M. A. Nicolelis, "Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex," *Nat. Neurosci.*, vol. 2, pp. 664–670, 1999.

[2] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue, "Instant neural control of a movement signal," *Nature*, vol. 416, pp. 141–142, 2002.

[3] 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.

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

[5] 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.

[6] G. Santhanam *et al.*, "A high-performance brain-computer interface," *Nature*, vol. 442, pp. 195–198, 2006.

[7] M. Velliste, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, "Cortical control of a prosthetic arm for self-feeding," *Nature*, vol. 453, pp. 1098–1101, 2008.

[8] C. T. Moritz, S. I. Perlmutter, and E. E. Fetz, "Direct control of paralysed muscles by cortical neurons," *Nature*, vol. 456, pp. 639–642, 2008.

[9] B. Jarosiewicz *et al.*, "Functional network reorganization during learning in a brain-computer interface paradigm," in *Proc. Nat. Acad. Sci.*, 2008, vol. 105, pp. 19486–19491.

[10] K. Ganguly and J. M. Carmena, "Emergence of a stable cortical map for neuroprosthetic control," *PLoS Biol.*, vol. 7, no. 7, p. e1000153, 2009.

[11] J. E. O'Doherty, M. A. Lebedev, T. L. Hanson, N. A. Fitzsimmons, and M. A. L. Nicolelis, "A brain-machine interface instructed by direct intracortical microstimulation," *Front. Integr. Neurosci.*, vol. 3, no. 20, 2009.

[12] A. J. Suminski, D. C. Tkach, A. H. Fagg, and N. G. Hatsopoulos, "Incorporating feedback from multiple sensory modalities enhances brain-machine interface control," *J. Neurosci.*, vol. 30, no. 50, pp. 16777–16787, 2010.

[13] L. R. Hochberg *et al.*, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, pp. 164–171, 2006.

[14] S. P. Kim, J. D. Simeral, L. R. Hochberg, J. P. Donoghue, and M. J. Black, "Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia," *J. Neural Eng.*, vol. 5, pp. 455–476, 2008.

[15] J. del R. Millán and J. M. Carmena, "Invasive or noninvasive: Understanding brain-machine interface technology," *IEEE Eng. Med. Biol. Mag.*, vol. 29, no. 1, pp. 16–22, 2010.

[16] V. Gilja, C. A. Chestek, I. Diester, J. M. Henderson, K. Deisseroth, and K. V. Shenoy, "Challenges and opportunities for next-generation intra-cortically based neural prostheses," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 7, pp. 1891–1899, Jul. 2011.

[17] V. Gilja *et al.*, "A high-performance continuous cortically-controlled prosthesis enabled by feedback control design," presented at the Soc. Neurosci. Annu. Meeting, San Diego, CA, 2010.

[18] V. Gilja, P. Nuyujukian, C. Chestek, J. Cunningham, B. Yu, S. Ryu, and K. Shenoy, "High-performance continuous neural cursor control enabled by feedback control perspective," in *Front. Neurosci.: Comp. Syst. Neurosci. Conf.*, 2010.
[19] L. Shpigelman, H. Lalazar, and E. Vaadia, "Kernel-arma for hand tracking and brain-machine interfacing during 3d motor control," in *Proc. Neural Inf. Process. Syst.*, 2008, pp. 1489–1496.
[20] S. Koyama, S. Chase, A. Whitford, M. Velliste, A. Schwartz, and R. Kass, "Comparison of braincomputer interface decoding algorithms in open-loop and closed-loop control," *J. Comp. Neurosci.*, vol. 29, pp. 73–87, 2010.
[21] K. Ganguly and J. M. Carmena, "Neural correlates of skill acquisition with a cortical brain-machine interface," *J. Motor Behavior*, vol. 42, no. 6, pp. 355–360, 2010.
[22] G. J. Gage, K. A. Ludwig, K. J. Otto, E. L. Ionides, and D. R. Kipke, "Naïve coadaptive cortical control," *J. Neural. Eng.*, vol. 2, pp. 52–63, 2005.
[23] B. Mahmoudi and J. C. Sanchez, "A symbiotic brain-machine interface through value-based decision making," *PLoS ONE*, vol. 6, no. 3, p. e14760, 2011.
[24] A. L. Orsborn, S. Dangi, H. G. Moorman, and J. M. Carmena, "Exploring time-scales of closed-loop decoder adaptation in brain-machine interfaces," in *Proc. IEEE EMBS 33rd Annu. Int. Conf.*, Boston, MA, 2011, pp. 5436–5439.
[25] G. Paxinos, X. F. Huang, and A. W. Toga, *The Rhesus Monkey Brain in Stereotaxic Coordinates*. San Diego, CA: Academic, 2000.
[26] S. Haykin, *Adaptive Filter Theory*, 4th ed. Upper Saddle River, NJ: Prentice Hall, 2001.
[27] W. Wu *et al.*, , S. Becker, S. Thrun, and K. Obermayer, Eds., "Neural decoding of cursor motion using a kalman filter," in *Advances in Neural Information Processing Systems 15*. Cambridge: MIT Press, 2003, pp. 117–124.
[28] R. Héliot, K. Ganguly, J. Jimenez, and J. M. Carmena, "Learning in closed-loop brain-machine interfaces: Modeling and experimental validation," *IEEE Trans. Syst. Man Cyber. Part B*, vol. 40, no. 5, pp. 1387–1397, Oct. 2010.
[29] I. S. Mackenzie, T. Kauppinen, and M. Silfverberg, "Accuracy measures for evaluating computer pointing devices," in *Proc. SIGCHI Conf. Human Factors Comp. Syst.*, Seattle, WA, 2001, pp. 9–16.
[30] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding of movement direction," *Science*, vol. 233, no. 4771, pp. 1416–1419, 1986.
[31] P. Nuyujukian, J. M. Fan, V. Gilja, P. S. Kalanithi, C. A. Chestek, and K. V. Shenoy, "Monkey models for brain-machine interfaces: The need for maintaining diversity," in *Proc. IEEE EMBS 33rd Annu. Int. Conf.*, Boston, MA, 2011, pp. 1301–1305.
[32] S. Dangi, S. Gowda, R. Heliot, and J. M. Carmena, "Adaptive kalman filtering for closed-loop brain-machine interface systems," in *Proc. 5th Int. IEEE EMBS Conf. Neural Eng.*, Cancun, Mexico, 2011, pp. 609–612.
[33] D. Tkach, J. Reimer, and N. G. Hatsopoulos, "Congruent activity during action and action observation in motor cortex," *J. Neurosci.*, vol. 27, no. 48, pp. 13241–13250, 2007.
[34] W. Truccolo, G. 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, 2008.
[35] A. J. Suminski, D. C. Tkach, and N. G. Hatsopoulos, "Exploiting multiple sensory modalities in brain-machine interfaces," *Neural Netw.*, vol. 22, pp. 1224–1234, 2009.
[36] K. Ganguly *et al.*, "Cortical representations of ipsilateral arm movements in monkey and man," *J. Neurosci.*, vol. 29, no. 41, pp. 12948–12956, 2009.
[37] B. Blankertz *et al.*, "Neurophysiological predictor of SMR-based BCI performance," *Neuroimage*, vol. 51, pp. 1303–1309, 2010.

Amy L. Orsborn received the B.S. degree in engineering physics from Case Western Reserve University, Cleveland, OH, in 2007. She is currently working towards the Ph.D. degree at the University of California, Berkeley—University of California, San Francisco graduate group in bioengineering.

She is currently working with the Brain–Machine Interface Systems Laboratory, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley. Her research interests include brain–machine interfaces, closed-loop systems, limb dynamics control, and neural representations of movement.

Siddharth Dangi received the B.S. and M.S. degrees in electrical engineering and computer science from the University of California, Berkeley, in 2010 and 2011, respectively. He is currently working toward the Ph.D. degree in the Brain–Machine Interface Systems Laboratory, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley.

His research interests include brain–machine interfaces and closed-loop decoder adaptation algorithms.

Helene G. Moorman received the B.S. degree in brain and cognitive science from the Massachusetts Institute of Technology, Cambridge, in 2008. She is currently working toward the Ph.D. degree in neuroscience at the University of California, Berkeley.

A member of the Brain Machine Interface Systems Laboratory, her research interests include motor prosthetics, motor control, and learning and plasticity in the motor system.

Ms. Moorman is the recipient of a National Science Foundation Graduate Fellowship.

Jose M. Carmena (S'97–M'99–SM'09) received the B.S. and M.S. degrees in electrical engineering from the Polytechnic University of Valencia, Valencia, Spain, in 1995, and the University of Valencia, Valencia, Spain, in 1997. Following those he received the M.S. degree in artificial intelligence and the Ph.D. degree in robotics both from the University of Edinburgh, Edinburgh, U.K., in 1998 and 2002, respectively. From 2002 to 2005 he was a Postdoctoral Fellow at the Department of Neurobiology and the Center for Neuroengineering at Duke University, Durham, NC.

He is an Associate Professor of Electrical Engineering, Cognitive Science, and Neuroscience at the University of California (UC)-Berkeley, and Co-Director of the Center for Neural Engineering and Prostheses at UC Berkeley and UC San Francisco. His research program in neural engineering and systems neuroscience is aimed at understanding the neural basis of sensorimotor learning and control, and at building the science and engineering base that will allow the creation of reliable neuroprosthetic systems for the severely disabled.

Dr. Carmena is senior member of the Society for Neuroscience and the Neural Control of Movement Society. He has been the recipient of the IEEE Engineering in Medicine and Biology Society Early Career Achievement Award (2011), the Aspen Brain Forum Prize in Neurotechnology (2010), the National Science Foundation CAREER Award (2010), the Alfred P. Sloan Research Fellowship (2009), the Okawa Foundation Research Grant Award (2007), the UC Berkeley Hellman Faculty Award (2007), and the Christopher Reeve Paralysis Foundation Postdoctoral Fellowship (2003).