Unsupervised adaptation of brain machine interface decoders

Tayfun Gürel[1,2] and Carsten Mehring[1,3]

[1]Bernstein Center Freiburg, Albert-Ludwig University of Freiburg, Germany

Current address: [2]Imperial College London, Department of Bioengineering, London, UK

[3]Imperial College London, Department of Bioengineering, London, UK

Tayfun.Guerel@bcf.uni-freiburg.de, mehring@imperial.ac.uk

ABSTRACT. The performance of neural decoders can degrade over time due to nonstationarities in the relationship between neuronal activity and behavior. In this case, brain-machine interfaces (BMI) require adaptation of their decoders to maintain high performance across time. One way to achieve this is by use of periodical calibration phases, during which the BMI system (or an external human demonstrator) instructs the user to perform certain movements or behaviors. This approach has two disadvantages: (i) calibration phases interrupt the autonomous operation of the BMI and (ii) between two calibration phases the BMI performance might not be stable but continuously decrease. A better alternative would be that the BMI decoder is able to continuously adapt in an unsupervised manner during autonomous BMI operation, i.e. without knowing the movement intentions of the user.

In the present article, we present an efficient method for such unsupervised training of BMI systems for continuous movement control. The proposed method utilizes a cost function derived from neuronal recordings, which guides a learning algorithm to evaluate the decoding parameters. We verify the performance of our adaptive method by simulating a BMI user with an optimal feedback control model and its interaction with our adaptive BMI decoder. The simulation results show that the cost function and the algorithm yield fast and precise trajectories towards targets at random orientations on a 2-dimensional computer screen. For initially unknown and nonstationary tuning parameters, our unsupervised method is still able to generate precise trajectories and to keep its performance stable in the long term. The algorithm can optionally work also with neuronal error signals instead or in conjunction with the proposed unsupervised adaptation.

1. INTRODUCTION

Brain Machine Interfaces (BMI) are systems that convey users brain signals into choices, text or movement [1, 2, 3, 4, 5]. Being still in development, BMI systems can potentially provide assistive technology to people with severe neurological disorders and spinal cord injuries, as their functioning does not depend on intact muscles. For motor control tasks, parameters of intended movements (e.g. movement direction or velocity) can be decoded from electrophysiological recordings of individual neurons [6, 7], from local field potentials inside [8, 9] and on the surface of the cerebral cortex [10, 11, 12, 13, 14] or from electrical fields on the scalp [15, 16, 17]. The decoded parameters can be used for online control of external effectors [18, 19, 20, 7].

The relation between recorded brain activity and movement is subject to change as a result of neuronal adaptation or due to changes in attention, motivation and vigilance of the user. Moreover, the neural activity-movement relationship might

1

be affected by changes in the behavioral context or changes in the recording. All these nonstationarities can decrease the accuracy of movements decoded from the brain-activity. A solution to this problem is employing adaptive decoders, i.e. decoders that learn online from measured neuronal activity during the operation of a BMI system and that track the changing tuning parameters [21, 15].

Adaptive BMI decoders can be categorized according to which signals are employed for adaptation: ***Supervised*** adaptive decoders use user's known movement intentions in conjunction with corresponding neuronal signals. During autonomous daily operation of the BMI systems, however, neither the user's precise movement intention nor his movement goal is known to the BMI decoder - otherwise one would not need a decoder. Therefore, supervised decoders can only adapt during calibration phases, where the BMI system guides the user to perform pre-specified movements. ***Unsupervised*** adaptive decoders, in contrast, track tuning changes automatically without a calibration phase. They can for example benefit from multimodal distributions of neuronal signals to perform probabilistic unsupervised clustering [22, 23, 24, 25]. Evidently much less information is available to the adaptation algorithm in the unsupervised case compared to the supervised case. Unsupervised decoders, hence, might not work for strong nonstationarities and might be less accurate and slower during adaptation. The third category, namely ***error-signal based*** adaptive decoders, do not use an informative supervision signal such as instantaneous movement velocity or target position but employ neuronal evaluation (or error) signals, which the brain generates e.g. if the current movement of the external effector is different from the intended movement or if the movement goal is not reached [26, 27, 28]. *Unsupervised* and *error-based* adaptive decoders are applicable during autonomous BMI control in contrast to *supervised* adaptive decoders.

1.1. **Related work brain machine interfaces.** In earlier work, BMI research has already addressed online adaptivity issue. For instance, Taylor et al. [21] has proposed a BMI system, where individual neuron's directional tuning changes are tracked with online adaptive linear filters. Wolpaw and McFarland have shown that intended 2-dimensional cursor movements can be estimated from EEG recordings [15]. In that study, they employed Least Mean Squares (LMS) algorithm to update the parameters of a linear filter after each trial. Later, Wolpaw et al. has also shown that a similar method can be used to decode 3-dimensional movements from EEG recordings [29]. Vidaurre et al. have proposed adaptive versions of Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis for cue-based discrete choice BMI-tasks [30, 31, 25]. These works employ supervised learning algorithms, i.e. they necessitate that the decoder knows the target of the movement or the choice in advance and adapts the decoding parameters. In other words, the employed methods know and make use of the *true* label of the recorded neural activity.

More recently, DiGiovanna et al. [32], Sanchez et al. [33] , Gage et al. [34] have proposed co-adaptive BMIs, where both subjects (rats) and decoders adapt themselves in order to perform a defined task. This task is either a discrete choice task like pushing a lever [33, 32] or a continuous estimation task such as reproducing the frequency of the cue tone by neural activity [34]. Gage et al. employ a supervised adaptive Kalman filter to update the decoder parameters that match the neural activity to cue tone frequency. DiGiovanna et al. and Sanchez et al. utilize

a reward signal to train the decoder. The reward signal is an indicator of a successful completion of the discrete choice task. The decoder adaptation follows a reinforcement learning algorithm rather than a supervised one. Whether the target has been reached, however, in contrast to a fully autonomous BMI task, is known to the decoder.

Error related activity in neural recordings [35, 36] is very interesting from a BMI perspective. In both discrete choice tasks and cursor movement tasks, EEG activity has been shown to be modulated, when subjects notice their own errors in the given tasks [16, 37]. The modulation of the neural activity is correlated with the failure of the BMI task, and hence, can be used to modify the decoder model. With reliable detection of error related activity, the requirement for the decoder to know the target location could be removed. Instead, the error signal could be utilized as an inverse reward signal [38, 39]. An unsupervised, i.e. working in complete absence of a supervision or error signal, approach has also been taken for an EEG-based BMI binary choice task. Blumberg et al. have proposed an adaptive unsupervised LDA method, where distribution parameters for each class are updated by the Expectation-Maximization algorithm [22]. More recently, unsupervised LDA has also been applied to an EEG based discrete choice task [23, 24, 25]. Unsupervised LDA, however, is limited to finite number of targets. In other words, it can not be applied to BMI tasks where possible target locations are arbitrarily many and uniformly (or unimodal) distributed. Kalman filtering methods for unsupervised adaptation after an initial supervised calibration have also been proposed for trajectory decoding tasks [40, 41, 42]. These methods adapt by maintaining consistency between a model of movement kinematics and a neuronal encoding model.

1.2. **Optimal control theory for motor behavior.** Motor behavior and associated limb trajectories is most commonly and successfully explained by optimality principles that trade off precision, smoothness or speed against energy consumption [43]. This trade off is often expressed as a motor cost function. Within the optimality based theory motor behavior, open loop and feedback optimization compose two distinct classes of motor control models. The former involves the optimization of the movement prior to its start ignoring the online sensory feedback, whereas the latter incorporates a feedback mechanism and intervenes with the average movement when intervention is sufficiently cheap. Optimal feedback control (OFC) models explain optimal strategies better than open loop models under uncertainty [44]. OFC models also provide a framework, in which high movement goals can be discounted based on online sensory input flow [43]. Optimal feedback control usually accommodates a state estimator module, e.g. a Kalman filter, and a Linear-Quadratic controller, which expresses the motor command as a linear mapping of the estimated state [45]. The state estimator uses sensory feedback as well as the afferent copy of the motor command. The motor command is a feedback rule between the sensory motor system and the environment. OFC models obey the minimal intervention principle, i.e. they utilize more effort and cost for relatively unsuccessful movements in order to correct for the errors [44, 46]. Minimal intervention principle is also very important for the current work, as substantial deviations can result from both noise and a model mismatch between the organism and the environment. The non-minimal intervention, hence, can be interpreted as a sign of a possible model mismatch between a BMI user and the decoder. Recent

evidence indicates that OFC should also model trial-by-trial and online adaptation in order to be plausible empirical evidence on motor adaptation [47, 48].

1.3. **Scope and goals of our research.** During autonomous operation of a BMI system, the BMI decoder does not know the individual movement intentions of the subject nor the goal of the movement, apart from what can be derived from the measured brain activity and from sensing the environment. Hence, the decoder has no access to an explicit supervision signal for adaptation. We, therefore, developed an algorithmic framework for adaptive decoding without supervision in which the following adaptive decoding strategies could be implemented:

(1) Unsupervised, here the adaptation works using exclusively the neuronal signals controlling the BMI movements.
(2) Error signal based, the adaptation uses binary neuronal error signals which indicate the time points where the decoded movement deviates from the intended movement more than a certain amount.
(3) Unsupervised + error signal based, the combination of the adaptive mechanisms of (i) and (ii).

With a BMI system involving those strategies, lifelong changes in brain dynamics do not have to be tracked by supervised calibration phases, where users would go under attentive training. Instead, decoder adaptation would track possible model mismatches continually. The BMI users behavior could provide a hint to the decoder even without an explicit supervision signal. It is presumable that inaccurate movements result in corrective attempts, which in turn increase control signals and control signal variability. Optimal feedback control models, which widely explain human motor behavior, support this presumption as they would generate jerky and larger control signals under mismatches between the users and the systems tuning parameters. Here, we develop a cost measure for online unsupervised decoder adaptation, which takes the amplitudes and the variations in the user's control signals into account (strategy i). Our unsupervised method incorporates a log-linear model that relates the decoding parameters to the cost via meta-parameters. Randomly selected chosen parameters are tested during also randomly chosen exploration episodes. In the rest of the time, the best decoding parameters according to the existing model (initially random) are used. The switch between these exploration and exploitation episodes is random and follows an ϵ-greedy policy (see section 2) . Harvested rewards for all episodes and associated decoding parameters compose the training data, from which meta-parameters are detected using the least squares method recursively. Note that we utilize the same algorithm for strategies (ii) and (iii). In strategy (ii), we employ the error signal as the cost instead of the derived one. In strategy (iii), a combination of both measures serves as the cost.

2. METHODS

2.1. **Simulated task.** The user's task is to move a cursor on a 2-dimensional screen from one target to the next. Each new target is located randomly on a circle of 0.2 m radius around the previous target (figure 1). If the user reaches the target within 4 seconds and stays there for 0.16 seconds, the trial is successful. After an unsuccessful trial, the users selects a new random target. Upon success, the trial *immediately* ends and the user selects a new target again. The state of the controlled
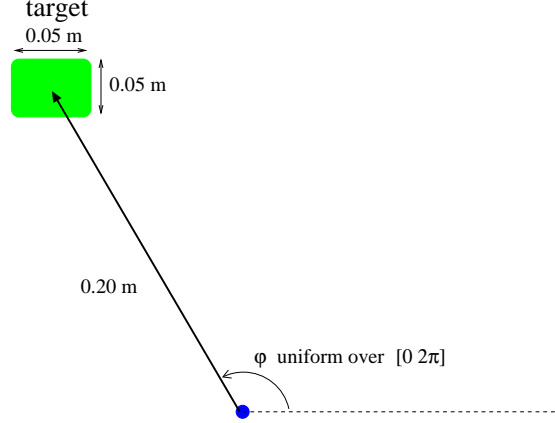
FIGURE 1. BMI task: The user has to move the computer cursor towards the target. The target is 0.2 meters away, at a random direction. The target location is decided by the user and unknown to the decoder, also for training purposes. The target has to be reached in 4 seconds and the cursor has to stay on the target for at least 0.16 seconds. Upon both success or failure, the user selects a new target.

system, i.e computer screen and cursor, at a discrete time step, t, is given by

$$x_t = (p_t^1, p_t^2, v_t^1, v_t^2, g_t^1, g_t^2)^\mathrm{T}.$$

where, v_t^1,, p_t^1 and g_t^1 are horizontal cursor velocity, cursor position and goal position, respectively. v_t^2,, p_t^2 and g_t^2 are the corresponding vertical state variables. The screen state evolves according to first order linear discrete time dynamics,

$$(1) \qquad\qquad x_{t+1} = A\,x_t \;+\; B_d\,u_t,$$

where u_t is the C-dimensional control signal and B_d is a $6 \times C$ dimensional decoder matrix. We assume that the motor command, u_t, affects only the cursor velocity directly. Therefore, B_d's first 2 and last 2 rows are 0:

$$B_d = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ b_d^{11} & b_d^{12} & & \cdots & & b_d^{1C} \\ b_d^{21} & b_d^{22} & & \cdots & & b_d^{2C} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

The state transition matrix A models the temporal evolution of the screen state. It simply performs the operation $(p_{t+1}^1, p_{t+1}^2) = (p_t^1, p_t^2) + (v_t^1, v_t^2)$,

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$
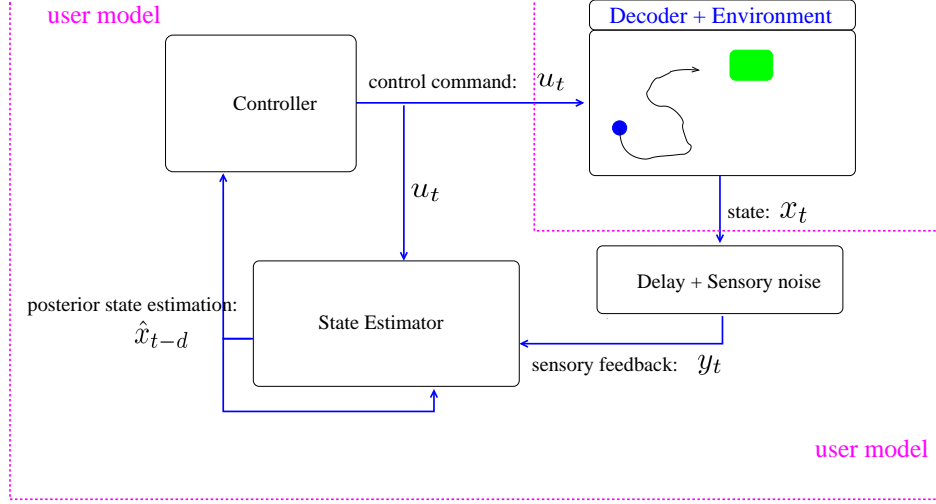
FIGURE 2. BMI user model and the environment. We model the BMI user with a stochastic optimal controller. The state estimator module is a Kalman filter that corrects the forward module estimation with sensory feedback. The controller generates the C-dimensional control command u_t, which is linearly converted to 2-dimensional cursor movement by the decoder.

Note that the goal position remains constant within a trial and it is left untouched by the linear dynamics of the screen state. Including the goal position in the state vector, however, simplifies the formulation of control signal generation by the user model (section 2.2).

2.2. **User model: stochastic optimal controller.** The BMI user is modeled as a stochastic optimal controller, who sends the C-dimensional control command u_t at discrete time step t (figure 2). The controller, i.e. the user, assumes that the screen state evolves according to a first order discrete time dynamics,

$$(2) \qquad x_{t+1} = A\,x_t \,+\, B_u\,u_t,$$

where B_u is the user's estimation of the decoder matrix B_d,

$$B_u = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ b_u^{11} & b_u^{12} & & ... & & b_u^{1C} \\ b_u^{21} & b_u^{22} & & ... & & b_u^{2C} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

The BMI user perceives the state of the cursor with sensory delay and normally distributed zero-mean noise,

$$y_t = H\,x_{t-d} \,+\, \eta_t,$$

where d is the sensory delay in time steps and η_t is the noise drawn from $\mathcal{N}(\mathbf{0}, \Omega_\eta)$. The user observes a 4-dimensional vector, y_t, which contains the velocity and position observations. H is therefore:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

In computer simulations, we use a time step of 40 ms. Sensory delay is set to 200 ms, i.e. 5 time steps. We assume that all dimensions of η_t are independently normally distributed with standard deviations of $(0.0004 \text{ m}, 0.0004 \text{ m}, 0.1 \text{ m/s}, 0.1 \text{ m/s})^{\mathrm{T}}$.

We model the control signals from the BMI-user as the output of a stochastic optimal controller. The BMI-user model aims at optimizing the cost function

(3)
$$J_u = \sum_t \left(q \, \|g_t - p_t\|^2 \; + \; r \, u_t^{\mathrm{T}} u_t \right),$$

$\|g_t - p_t\|$ stands for the euclidean distance between the 2-dimensional cursor position and the goal position vectors. q and r are constants that account for the relative weights of the two terms in the cost. The same cost expression can be written alternatively as,

$$J_u = \sum_t \left(x_t^{\mathrm{T}} Q_t \, x_t \; + \; u_t^{\mathrm{T}} R_t \, u_t \right),$$

where Q_t is a 6×6 matrix that allows for the quadratic expression of the distance cost,

$$Q_t = q \begin{pmatrix} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \end{pmatrix},$$

and $R_t = r \, I$. Q_t and R_t stay constant for all t in our cost model, $Q_t = Q$ and $R_t = R$ for all t.

Assume that the stochastic optimal controller minimizes the cost by sending the optimal control command u_t^* at every time step t. In fact, the optimal command is disturbed by noise. Here, we model the inherent noise in biological circuits with a 0-mean normally distributed additive noise vector ρ_t,

$$u_t = u_t^* + \rho_t.$$

This noise consequently presents itself also as additive at state update in equation 1

$$\begin{aligned} x_{t+1} &= A \, x_t \; + \; B_d \, u_t \\ &= A \, x_t \; + \; B_d \, u_t^* + B_d \, \rho_t \\ &= A \, x_t \; + \; B_d \, u_t^* + \omega_t, \end{aligned}$$

where $\omega_t \sim \mathcal{N}(\mathbf{0}, \Omega_\omega)$. The problem of computing u_t^* is known as Linear-Quadratic-Gaussian (LQG) control and can be recursively solved by an interconnected Linear-Quadratic-Regulator [45, 49],

$$u_t^* = -L_t \hat{x}_t$$

$$L_t = (R + B_u^{\mathrm{T}} S_{t+1} B_u)^{-1} B_u^{\mathrm{T}} S_{t+1} A$$

$$S_t = Q_t + A^{\mathrm{T}} S_{t+1}(A - B_u L_t),$$

and a state estimating Kalman Filter,

$$\text{(4)} \qquad \hat{x}_{t+1} = A\hat{x}_t + B_u u_t^* + K_t(y_t - H\hat{x}_t)$$

$$\text{(5)} \qquad K_t = A\Sigma_t H^{\mathrm{T}}(H\Sigma_t H^{\mathrm{T}} + \Omega_\eta)^{-1}$$

$$\text{(6)} \qquad \Sigma_{t+1} = \Omega_{\mathrm{FW}} + A\Sigma_t A^{\mathrm{T}} - K_t H\Sigma_t A^{\mathrm{T}}.$$

Here, Σ_t is the covariance estimate of the state vector variable x_t and $\hat{x}_t$ is estimate of its mean value posterior to noisy observation y_t. Ω_{FW} is the covariance of the noise associated with the forward model prediction. Kalman filter above is a model for the state estimator in user's motor control circuitry. B_u is the user's estimation for B_d. When B_u deviates from B_d, the user's control signals are not optimal anymore. Above equations assume that the sensory delay equals to 1 time step. Larger sensory delays, e.g. d time steps, can be realized by using an augmented state vector, $\tilde{x}_t$, which contains $d + 1$ states together [44, 48],

$$\tilde{x}_t = (x_t^{\mathrm{T}}, x_{t-1}^{\mathrm{T}}, \dots, x_{t-d}^{\mathrm{T}})^{\mathrm{T}}.$$

State transition and observation matrices are redefined for the augmented state space,

$$\tilde{A} = \begin{pmatrix} A & 0 & \cdots & & 0 \\ I & 0 & \cdots & & 0 \\ 0 & I & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I & 0 \end{pmatrix}, \quad \tilde{H} = (0, \cdots, 0, H) \text{ and } \tilde{B}_u = \begin{pmatrix} B_u \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

in order to satisfy the system dynamics, $\tilde{x}_{t+1} = \tilde{A}\tilde{x}_t + \tilde{B}_u u_t$. Kalman filter equations 4, 5 and 6 are also modified according to augmented states and system parameters: prior state and covariance estimations before the delayed observation, i.e. $y_{t+1} = H x_{t+1-d} + \eta_{t+1}$, are computed using the subject's forward model,

$$\tilde{x}_{t+1}^+ = \tilde{A}\hat{\tilde{x}}_t + \tilde{B}_u u_t^*$$

$$\tilde{\Sigma}_{t+1}^+ = \tilde{A}\hat{\tilde{\Sigma}}_t \tilde{A}^{\mathrm{T}} + \tilde{\Omega}_{\mathrm{FW}},$$

Posterior state and covariance estimates are similarly computed using the Kalman gain matrix $\tilde{K}_{t+1}$,

$$\tilde{K}_{t+1} = \tilde{\Sigma}_{t+1}^{+} \tilde{H}^{\mathrm{T}} (\tilde{H} \tilde{\Sigma}_t^{+} \tilde{H}^{\mathrm{T}} + \Omega_\eta)^{-1}$$

$$\hat{\tilde{x}}_{t+1} = \tilde{x}_{t+1}^{+} + \tilde{K}_{t+1} (y_{t+1} - \tilde{H} \tilde{x}_{t+1}^{+})$$

$$\hat{\tilde{\Sigma}}_{t+1} = (I - \tilde{K}_{t+1} \tilde{H}) \tilde{\Sigma}_{t+1}^{+}.$$

Note that in our simulations, $\tilde{\Omega}_{\mathrm{FW}}$ is set to a diagonal matrix, whose first 4 diagonal entries are the squares of the noise standard deviations, $(0.0025 \text{ m}, 0.0025 \text{ m}, 0.625 \text{ m/s}, 0.625 \text{ m/s})^{\mathrm{T}}$, and the remaining entries are 0.

2.3. **Decoder models.** The decoder is modeled by B_d, i.e. it decodes velocity information from the neuronal control signal u_t. This decoder matrix might deviate from the user's decoder matrix B_u, on the basis of which he generates his control signals. Therefore, the proposed adaptive decoders adapt their B_d according to B_u. In the current section, we describe three decoders: Our recursive least squares (RLS) based learning algorithm with *unsupervised* and *error-signal based* cost functions as well as a *supervised* RLS filter for performance comparison.

2.3.1. *Unsupervised learning algorithm.* For *unsupervised* and *error-signal based* decoder adaptation, we define a cost function and estimate B_u by optimizing the proposed cost function. In the unsupervised setting, the cost is associated with control signal,

$$(7) \qquad J_n^d = \sum_{t=n-T+1}^{n} u_t^{\mathrm{T}} u_t,$$

Here, u_t^{T} stands for the transpose of the control command vector. t and n are indices over time steps. T is the number of time steps in the control signal history for computing the cost function. Note that the decoder needs to know only the control signal, u_t, in order to compute the above cost function. This cost function reflects the control-related term of the user's cost function (equation 3). The value of the cost function is expected to be high, if the user aims at correcting the movement errors which result from a model mismatch between the user and the decoder, i.e. between B_u and B_d.

We name the cost in equation 7 *amplitude cost*, as it is based on the amplitudes of the control commands. We, however, propose a further cost function that can be utilized for decoder adaptation, namely *deviation cost*. Deviation cost uses the variances of the control signals across time instead of the summed squared norms of the control commands,

$$(8) \qquad J_{dev}^d(n) = \sum_{c=1}^{C} \sum_{t=n-T+1}^{n} (u_t^c - \bar{u}^c)^2,$$

where c is an index over control channels u_t^c is the control command at channel c at time step t. $\bar{u}^c$) is the mean value of u_t^c for channel c across the interval

$[n - T + 1, n]$. A weighted sum of the above costs can also be used as the cost function,

$$(9) \qquad J^d_{ampl+dev}(n) = \sum_{t=n-T+1}^{n} u_t^{\mathrm{T}} u_t + Z \sum_{c=1}^{C} \sum_{t=n-T+1}^{n} (u_t^c - \bar{u}^c)^2,$$

where Z is a constant for weighting the contributions from each individual cost type.

Alternatively, in case neuronal evaluation signals (i.e. error signals) are available in the recordings, we use the number of errors over a finite number of discrete time steps as cost,

$$(10) \qquad J^d_n = \sum_{t=n-T+1}^{n} err_t.$$

We simulated the neuronal error signal by assuming that neuronal error signals are generated if the deviation between intended and performed velocities exceeds a certain amount,

$$err_t = \begin{cases} 1, & \text{for } cos(v_t^*, v_t) \leq cos(20°) \\ 0, & \text{for } cos(v_t^*, v_t) > cos(20°), \end{cases}$$

where v_t^* is the intended velocity. err_t is swapped probabilistically with a probability of κ in order to reflect the reliability of error signals. Note that similar binary *movement mismatch events* are also recorded in human ECoG [28], though $20°$ in our simulation was arbitrarily choosen (see discussion).

We assume a log-linear model for the decoder cost. Let β be the parameter vector generated by the horizontal concatenation of the third and fourth rows in B_d matrix, i.e. $\beta = [B_{d3}, B_{d4}]$. The model estimates the decoder cost as,

$$(11) \qquad \hat{J}^d = \exp(-[\beta^{\mathrm{T}} \ b] \ w),$$

where b is a constant bias value concatenated to the flattened decoder parameter β and w is the column vector of the meta-parameters of this log-linear model. We denote the $-\log$ of the decoder cost by ℓ,

$$\hat{\ell}_n = -\log(\hat{J}^d_n) = [\beta_n^{\mathrm{T}} \ b] \ w_n = {\beta'_n}^{\mathrm{T}} w_n.$$

Let $[\beta^{\mathrm{T}} b] = {\beta'_n}^{\mathrm{T}}$. Here, the task is to learn w from explored β and J^d collections and to simultaneously optimize β for a given w. Note that for a given w, the cost-minimizing β would go the infinity, since $-\log$-cost linearly depends on w. Therefore, the minimization is performed on the unit circle, i.e. $|\beta| = 1$. The motivation here is to generate trajectories in the right direction rather than to optimize the speed of movement. The goal of the unsupervised as well as the error-signal based learning algorithm is to minimize the summed squared error,

$$(12) \qquad \xi_n = \sum_{k=1}^{n} \lambda^{n-k} e_k^2,$$

where $e_k = (\ell_k - \hat{\ell}_k) = (\ell_k - {\beta'_k}^{\mathrm{T}} w_n)$. n is the index of the current time step and k is an index over past time steps. λ is a constant for degrading the relative

contribution of the past time steps ($0 < \lambda \leq 1$). ξ_n can be further expressed as,

$$\xi_n = \sum_{k=1}^{n} \lambda^{n-k} (\ell_k^2 - 2w_n^{\mathrm{T}} \beta_k' \ell_k + w_n^{\mathrm{T}} \beta_k' \beta_k'^{\mathrm{T}} w_n).$$

Optimum parameters can be found by solving

$$\nabla_{w_n} \xi_n = \sum_{k=1}^{n} \lambda^{n-k} (-2\beta_k' \ell_k + 2\beta_k' \beta_k'^{\mathrm{T}} w_n) = 0.$$

Defining

$$\sum_{k=1}^{n} \lambda^{n-k} \beta_k' \ell_k = \Theta_n \text{ and } \sum_{k=1}^{n} \lambda^{n-k} \beta_k' \beta_k'^{\mathrm{T}} = \Psi_n,$$

solution to $\nabla_{w_n} \xi_n = 0$ can be found as

$$\begin{aligned} \nabla_{w_n} \xi_n &= 0 &= -2(\Theta_n - \Psi_n w_n) \\ \Rightarrow \Theta_n &= \Psi_n w_n \\ \Rightarrow \hat{w}_n &= \Psi_n^{-1} \Theta_n. \end{aligned}$$

Utilizing matrix inversion dilemma, Recursive Least Squares (RLS) [50] algorithm proposes a recursive formulation for Ψ^{-1}

$$\Psi_n^{-1} = P_n = \lambda^{-1}(P_{n-1} - k_n \beta_n'^{\mathrm{T}} P_{n-1}),$$

where

$$k_n = \frac{P_{n-1} \beta_n'}{\lambda + \beta_n'^{\mathrm{T}} P_{n-1} \beta_n'}.$$

Our method aims at simultaneous harvesting of various decoding parameters B_d and, hence, β and detecting optimum meta-parameters w. These subtasks correspond to exploration and exploitation phases of a reinforcement learning algorithm, respectively. We employ ϵ-greedy exploration policy. In other words, with a predefined probability, ϵ, the algorithm prefers exploring the parameter space, which means a new β is chosen randomly. Otherwise, i.e. with a probability of $1 - \epsilon$, the algorithm uses the best decoding parameters, i.e. the beta that minimizes the estimated decoder cost (equation 11) . Given $\hat{w}$, the current estimate of w, the optimal unit normed β is computed by finding $\arg\max_{|\beta|=1} \beta'^{\mathrm{T}} \hat{w}$. This is equivalent to maximizing the cosine between β' and $\hat{w}$ by setting $\beta' = \hat{w}$ and normalizing the corresponding β. A pseudocode for the algorithm is sketched in table 2.3.1.

2.3.2. *Adaptive supervised recursive least squares filtering.* Under the assumption that the decoder knows the intended movements of the user, B_d can be adapted to B_u by utilizing an RLS filter. Let v^{intent} be the intended velocity of the user at time step t.

$$v_t^{intent} = B_u' u_t,$$

where B_u' is the submatrix of the third and fourth rows of B_u, i.e. $B_u' = \begin{pmatrix} B_{u3} \\ B_{u4} \end{pmatrix}$.

The supervised decoder estimates the intended velocity using B_d,

$$\hat{v}_t^{intent} = B_d' u_t.$$

For the supervised decoder, it is assumed that the user's intent, v^{intent}, is known to the decoder. The supervised RLS learning algorithm infers B_u online from

**RLS based algorithm for continual *unsupervised* adaptation
of the decoding parameters**

for time step n at every T time steps do

$\qquad$ # select B
$\qquad$ **if** random $> \epsilon$

$$\beta_n \leftarrow \arg\max_{|\beta|=1} \beta'^{\mathrm{T}} \hat{w}_{n-T}$$

$\qquad$ **else**

$$\beta \rightarrow \text{random}$$

$\qquad$ **endif**

$\qquad$ # make prediction on $-log$-cost
$\qquad$ $\hat{\ell}_n = \beta_n'^{\mathrm{T}} \hat{w}_{n-T}$

$\qquad$ #observe the $-log$-cost of the last T time steps from
user's u_t
$\qquad\qquad$ $\ell_n = -\log(J_n^d) = -\log(\sum_{t=n-T+1}^{n} u_t^{\mathrm{T}} u_t)$
$\qquad\qquad$ #or alternatively according to equation 10

$\qquad$ #compute the prediction error
$\qquad$ $e_n = \ell_n - \hat{\ell}_n$

$\qquad$ #compute the innovation gain
$\qquad$ $k_n = \frac{P_{n-T} \beta_n'}{\lambda + \beta_n'^{\mathrm{T}} P_{n-T} \beta_n'}$

$\qquad$ #update meta parameters
$\qquad$ $\hat{w}_n = \hat{w}_{n-T} + k_n e_n$ [50]

$\qquad$ #update inverse of the correlation matrix
$\qquad$ $P_n = \lambda^{-1}(P_{n-T} - k_n \beta_n'^{\mathrm{T}} P_{n-T})$

$\qquad$ **endfor**

TABLE 1. A sketch of the unsupervised learning algorithm via RLS

$v_t^{intent} - \hat{v}_t^{intent}$. The supervised adaptive decoder is used to benchmark the proposed unsupervised and error-based adaptive decoders. The supervised RLS method is described in table 2.3.2. Note that P_t^u stands for the inverse of the $C \times C$ sample correlation matrix for $(u_0...u_t)$. λ_{sup} is the forgetting parameter of the supervised algorithm and is set to 1. P_0^u is set to $100\,I$.

RLS algorithm for *supervised* adaptation of the decoding parameters

for every time step t do

> \# make prediction on v_t^{intent}
> $\hat{v}_t^{intent} = B_d' u_t$
>
> \#observe user's v_t^{intent}
>
> \#compute the prediction error
> $e_t^{intent} = v_t^{intent} - \hat{v}_t^{intent}$
>
> \#compute the innovation gain
> $k_t = \frac{P_t^u u_t}{\lambda_{sup} + u_t^{\mathrm{T}} P_t^u u_t}$
>
> \#update B_d' (and hence B_d) matrix
> $B_d' \leftarrow B_d' + e_t^{intent} k_t^{\mathrm{T}}$
>
> \#update inverse of the correlation matrix
> $P_{t+1}^u = \lambda_{sup}^{-1}(P_t^u - k_t u_t^{\mathrm{T}} P_t^u)$

endfor

TABLE 2. A sketch of the supervised RLS algorithm

2.4. **Simulation Procedures.** We simulated the interaction of the optimal feedback controller with different adaptive decoders described in section 2. The behavior of the BMI user was simulated using the framework of stochastic optimal feedback control which has been shown to provide a good model for human motor behavior in various motor tasks [44, 51, 48]. The combined system of the optimal controller and the adaptive decoder was simulated at 40 ms time steps and we used a sensory delay of 200 ms. The user's task was to control a mouse cursor. The user selects a target at 0.2 m distance with a random orientation at each trial. The user has to reach the target within 4 seconds and stay at the target for at least 0.16 seconds. Upon both success or failure, the user selects a new target. The distance cost parameter q and control signal cost parameter r are both set to 0.02. We set Ω_ρ to $8 \times 10^6 I$, so that the cursor speed-noise had an average standard deviation of 0.0625 m/s over a uniform distribution of unit normed β vectors. The variance value was manually adjusted to obtain the aimed speed noise by testing on 10^4 unit normed random β vectors.

Note that the decoder does not have the information whether a trial is finished or continuing, nor does it know the target of the cursor movement. We simulated and evaluated the following adaptive decoders:

Unsupervised:: The decoder learns exclusively from continuous neuronal control signals of the user according to equation 7, without any additional
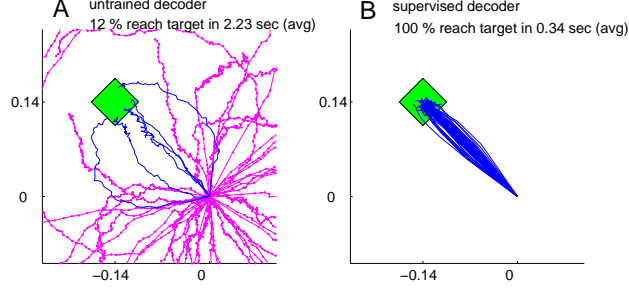
FIGURE 3. The trajectories for random decoders (A) and super-
vised adaptive decoders (B) during decoder-freeze, i.e. after de-
coder exploration and adaptation have been switched off for per-
formance evaluation. Magenta thick curves indicate the failed tra-
jectories. Each plot depicts the trajectories of 50 training simula-
tions, each at trial 1501. The 50 different targets and trajectories
at trial 1501 are rotated to the same orientation ($\frac{3\pi}{4}$) for a better
visual evaluation.

information. Note that the decoder knows neither whether the target has
been reached nor when a trial finishes.

Error signal based:: The adaptation uses binary neuronal error signals which
   indicate the time points where the decoded movement deviates from the
   intended movement more than $20°$. The reliability of the neuronal error
   signal was mainly assumed to be 80%, i.e. swapping probability, κ, was
   0.2. The effect of various κ on the decoding performance, however, was
   also investigated in section 3.3.

Unsupervised + error signal based:: The combination of the unsupervised
   and the error-signal based decoders, i.e. ℓ_n was a linear combination of the
   unsupervised ℓ_n and the error-signal based ℓ_n.

For all of the above algorithms, the current cost is computed from the last 100
time steps ($T = 100$). This corresponds to a parameter update period of 4 seconds.
λ of equation 12 was set to 1, i.e. no gradual discount of the parameter history
was performed. Exploration rate was 0.4, i.e. $\epsilon = 0.4$. We simulated 50 random
instantiations of all these unsupervised and the error signal based adaptive decod-
ing algorithms. 1501 successive trials of target reaching were simulated for each
instantiation. Note that from trial 1463 on the adaptation of the decoding algo-
rithms was frozen and the current optimal decoding parameters were used for the
last 39 trials (decoder-freeze). We evaluated their performances and compared it to
the performance of a supervised adaptive decoder where the adaptation is based on
perfect knowledge of the intended movement velocity at each time step (see sec-
tion 2.3.2). Such a supervised adaptive decoder yields the best possible adaptation,
however, it assumes knowledge that is certainly not available during autonomous
BMI operation. In addition, we also compared the performance of our adaptive
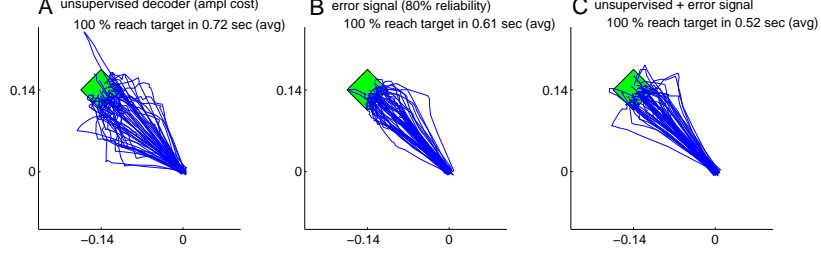decoders to the performance of a static untrained random decoder.

FIGURE 4. The trajectories for different strategies and their variations during decoder-freeze. Magenta thick curves indicate the failed trajectories. Each plot depicts the trajectories of 50 training simulations, each at trial 1501. Note that not only trial 1501 included 50 simulations, but the whole history of 1501 trials are simulated 50 times with random initial tunings. The 50 different targets and trajectories at trial 1501 are rotated to the same orientation ($\frac{3\pi}{4}$) for a better visual evaluation.

3. RESULTS

Our findings show that all the decoders described in section 2.4 can rapidly adapt to accurate cursor control from totally unknown tuning of the neuronal signals to movement velocity whereas the random decoder fails to reach the target (figure 3A). Although trajectories of the unsupervised and error-based decoders after adaptation are more jerky compared to the supervised case, they are still mainly straight and yield a high target hit rate of nearly 100% (figure 4). These results show that decoders can be trained during autonomous BMI control in the absence of any explicit supervision signal.

3.1. **Comparison of different adaptation algorithms.** As a baseline for comparisons, we implemented the supervised decoder that knows the intention of the user and fits the decoder parameters, B_d, based on this intention (see section 2.3.2). Though unrealistic, this learning scheme is obviously the most successful of the presented methods (figure 3B). In order to compare different algorithms, we utilize a measure that counts for the cumulative distance to the movement target and call it *cumulative error*,

$$\text{Cumulative Error}\,(m) = \sum_{t=1}^{M_m} \|g_t - p_t\|,$$

where g_t and p_t and are the 2-dimensional target and cursor position vectors at time step t of trial m, respectively. M_m is duration of trial m in time steps. For a more intuitive interpretation of the given error measure, we present out results in terms of *relative cumulative error*, which is the normalized cumulative error with respect
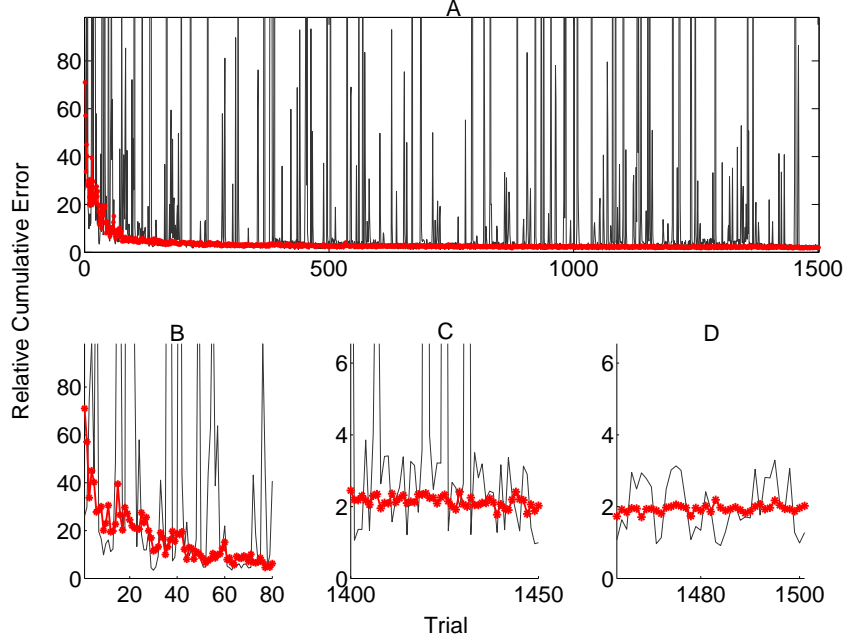
FIGURE 5. Evolution of the relative cumulative error (RCE) for
the unsupervised strategy (A). The plot shows the RCE with re-
spect to trial number for a single simulation (gray) and the Me-
dian RCE (MRCE) for 50 simulations with random initial tuning
parameters (red with marker). Zoom into RCE and MCE curves
for early (A) and late (B) learning and during decoder-freeze (C).

to average cumulative error of the supervised decoder after adaptation,

$$\text{Relative Cumulative Error} = \frac{\text{Cumulative Error}}{\text{Mean Supervised Cumulative Error}}.$$

Figure 5A depicts the evolution of the relative cumulative error for a single sim-
ulation (gray) of the *unsupervised* algorithm and median relative cumulative errors
(MRCE) for 50 randomly initialized simulations (red, *). The jumps in the gray
curve correspond to exploration periods, where random decoder matrices, B_d, are
explored and evaluated. The relative cumulative error shows different characteris-
tics for early learning (5B), late learning(5C) and decoder-freeze (5D) phases. The
early learning phase was investigated to compare the learning speeds of different
algorithms, whereas the late learning phase shows the *saturated* final performance
of the algorithms when adaptation continues. During decoder-freeze (last 39 time
steps), the adaptation (also the exploration) was stopped and the final performance
of the decoder was evaluated. No jumps in the performance are observed anymore
due to absence of exploration.

Figure 6 shows the comparison between supervised, error signal based and un-
supervised algorithms. MRCEs across 50 runs are shown for the entire simula-
tion (A), for early learning (B), for late learning (C) and during decoder-freeze
(D) (figure 6). Distribution of the relative cumulative errors for the individual
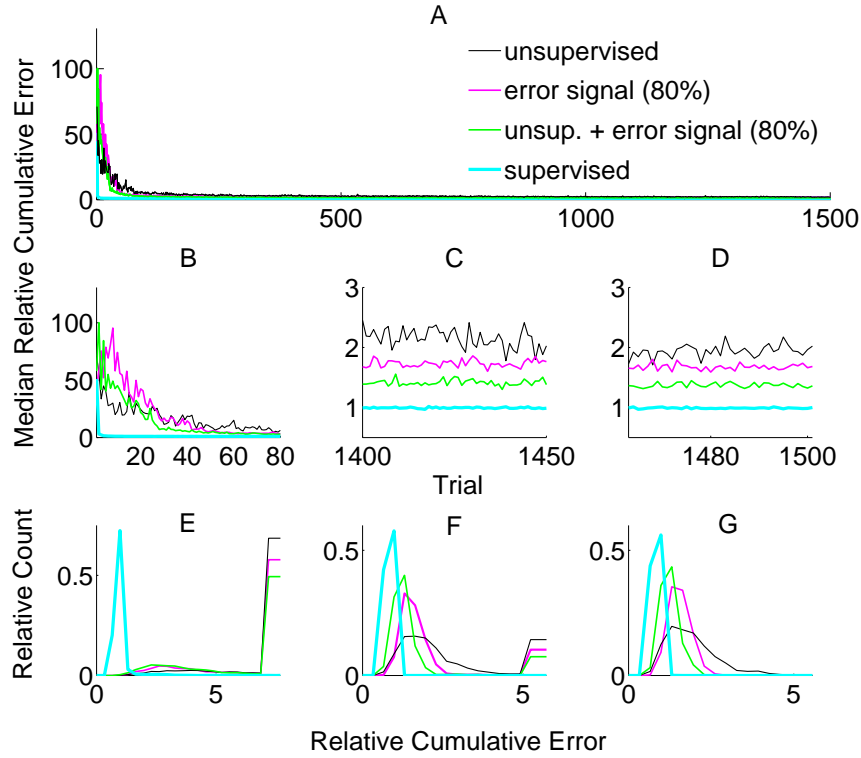phases (figure 6 E, F, G) reveal that the supervised algorithm is superior to the

FIGURE 6. Comparison of the unsupervised (black), the error signal based (80% reliability, magenta), their combination (green) and the supervised (cyan) strategies. Medians of relative cumulative errors of 50 simulations from each group for all of the trials (A). Zoomed medians for early (B) and late (C) learning and during decoder-freeze (D). The distributions of the relative cumulative errors for each of the phases (E, F, G). The rightmost values of the distribution plots denote the total relative counts of the outlier values that are greater than the associated x-axis value. The number of outlier values decreased across trials, i.e. it was the highest during early learning and zero during decoder-freeze. Outliers correspond to the failed trajectories in figures 4 and 7.

other algorithms in all phases ($p < 0.01$, Wilcoxon rank sum test). In all of the phases, the combination of the error signal based and the unsupervised strategies yielded a significantly lower cumulative error than the individual strategies alone ($p < 0.01$, Wilcoxon rank sum test). The performance of the the error-based learning was significantly better than the unsupervised strategy also for all of the phases ($p < 0.01$, Wilcoxon rank sum test). Note that the trajectories reached the targets not only during decoder-freeze (figure 4) but also mostly in the late learning phase (figure 7), where exploration can occasionally cause some trajectories to deviate strongly from a straight line towards the target.
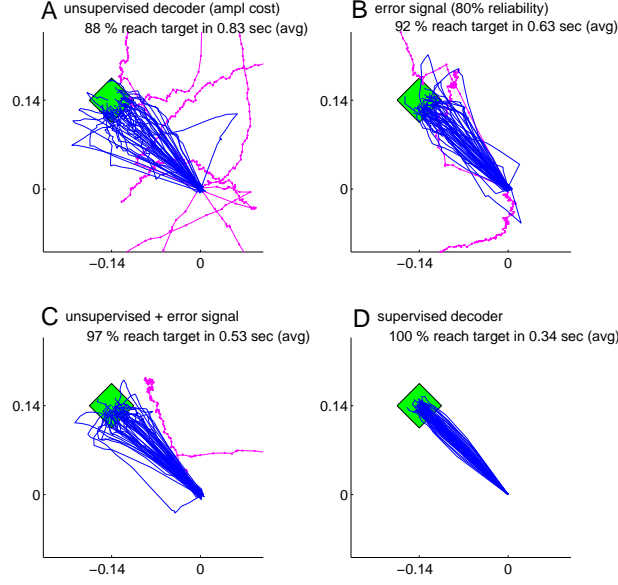
FIGURE 7. The trajectories for different strategies and their varia-
tions in the late learning phase. Magenta thick curves indicate the
failed trajectories. Each plot depicts the trajectories of 50 train-
ing simulations, each at trial 1404. Durations and target hit rates,
however, are computed from pooled trajectories of 5 consecutive
trials (1402-1406, totally 250 trajectories).

3.2. **Effect of the parameter update period.** We varied the parameter update
period, T, between 1.2 and 10 seconds in order to check the stability of the unsu-
pervised strategy with respect to this parameter (figure 8). Our results show that
for all tested update periods greater than or equal to 2.4 seconds, the performance
depended only weakly on exact value of the update period. Though the perfor-
mance for an update period of 2.4 s was significantly ($p < 0.05$, Wilcoxon rank
sum test) inferior compared to an update rate of 4 s or 10 s in both late learning
and during the freeze of the decoder, the difference was minor. Moreover, the per-
formance of the update periods 4 s and 10 s were not significantly different during
decoder-freeze ($p > 0.05$, Wilcoxon rank sum test). We, therefore, conclude that
our algorithm is robust against the update rate as long as it is high enough and used
an update period of 4 seconds for all remaining simulations.

3.3. **Effect of Error Signal Reliability.** Error signal based decoder performance
obviously depends on the reliability of the error signals. Our results so far used
an error signal with 80% reliability, i.e. $\kappa = 0.2$. Although several studies have
shown evidence on neuronal error signals [35, 36, 26, 27], conclusive quantita-
tive data on the reliability of the error signals is still missing. To compute the
dependence of the error-based adaptive decoder on κ, we varied it between 0 and
0.8. Our findings show that the reliability must be greater than 50% for successful
adaptation (figure 9A,B,C,D). Reliabilities of 80% and 100% yielded statistically
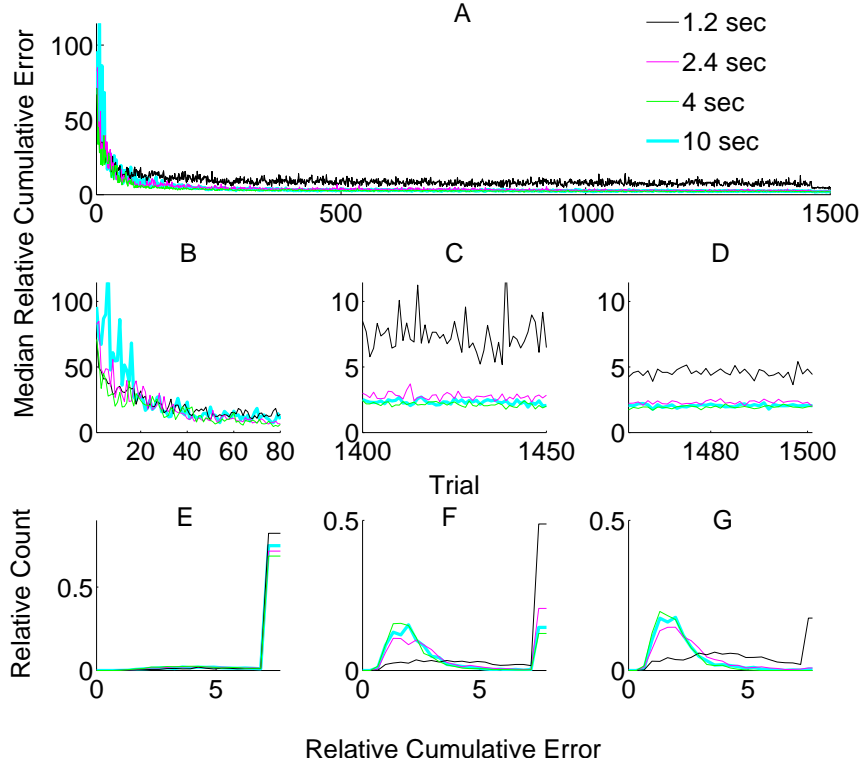indistinguishable performance during decoder-freeze ($p > 0.05$, Wilcoxon rank

FIGURE 8. Comparison of relative cumulative error measures using the unsupervised strategy for different update periods (T). The algorithm updates the decoding parameters either every 1.2 seconds (black) or 2.4 seconds (magenta) or 4 seconds (green) or 10 seconds (cyan). Medians of relative cumulative errors of 50 simulations from each group for all of the trials (A). Zoomed medians for early (B) and late (C) learning and during decoder-freeze (D). The distributions of the relative cumulative errors for each of the phases (E, F, G). The rightmost values of the distribution plots denote the total relative counts of the outlier values that are greater than the associated x-axis value. The number of outlier values decreased across trials, i.e. it was the highest during early learning and the lowest during decoder-freeze.

sum test). Though 80% was slightly yet significantly better than 100% in the late learning phase ($p = 0.049$). A decoder with a reliability of 60% yielded a significantly inferior performance in all of the phases to the decoders with 80% and 100% reliability (figure 9E, F, G, rank sum test, p-value < 0.05), its median error during late learning and freezing was only about 30% higher. Decreasing the reliability further to 50% drastically increased the median relative cumulative error.

3.4. **Adaptivity to nonstationary tuning.** We furthermore investigated, whether the unsupervised adaptive algorithm can cope with continual changes in the tuning. The velocity tuning parameters of the user, i.e. β_u, flattened third and fourth rows of B_u, were changed after each trial according to the following random walk
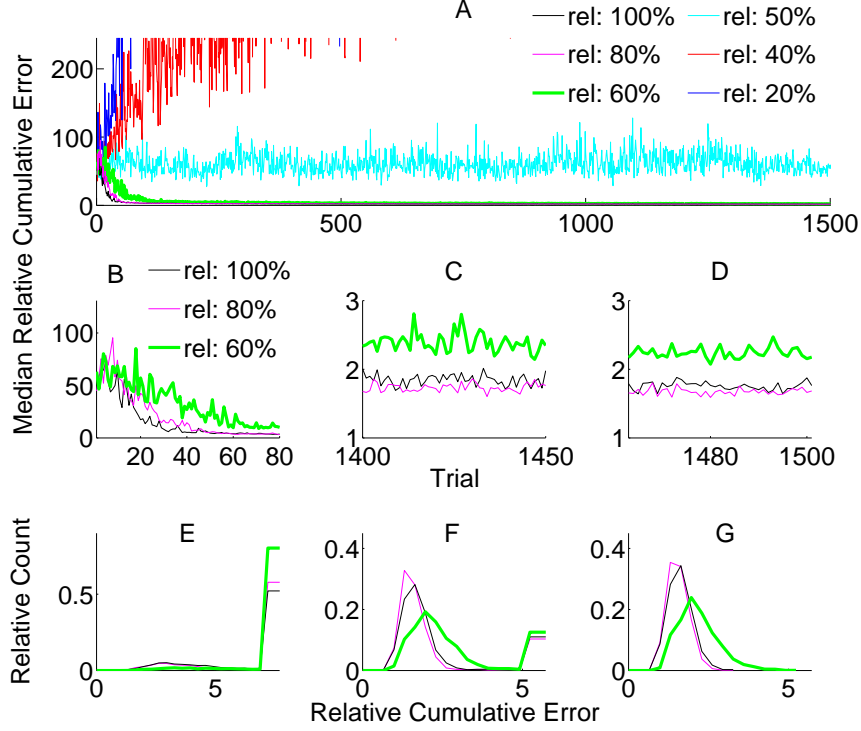
FIGURE 9. Comparison of relative cumulative error measures using the error-signal-based training strategy for different reliabilities of error signals. The error signal reliability is either 100% (black) or 80% (magenta) or 60% (green) or 50% (cyan) or 40% (red) or 20% (blue). Medians of relative cumulative errors of 50 simulations from each group for all of the trials (A). Zoomed medians for early (B) and late (C) learning and during decoder-freeze (D). The distributions of the relative cumulative errors for each of the phases (E, F, G). The rightmost values of the distribution plots denote the total relative counts of the outlier values that are greater than the associated x-axis value. The number of outlier values decreased across trials , i.e. it was the highest during early learning and the lowest during decoder-freeze. Outliers correspond to the failed trajectories in figures 4 and 7. In general, higher the reliability, better the performance. A minimum reliability of 60% is needed for successful training. 100% and 80% reliabilities are statistically equivalent during decoder-freeze (rank sum test, $p > 0.05$).

model,

$$\beta_u \leftarrow \beta_u + \varrho,$$

where ϱ is 40-dimensional row vector whose entries are randomly drawn from a normal distribution, $\varrho \sim \mathcal{N}(0, 0.007)$. We put a hard limit on the magnitude of the entries of β_u, so that they did not exceed -0.3 and 0.3. In order to investigate the performance of our algorithm under nonstationary tuning, 50 randomly initialized
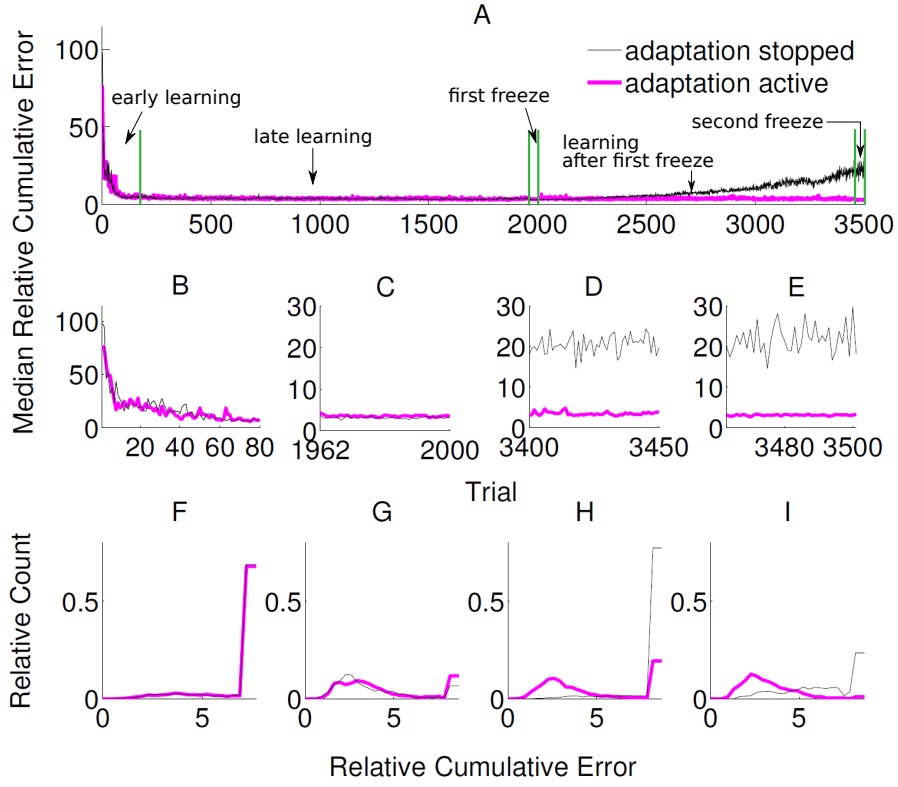
FIGURE 10. Relative cumulative errors for the unsupervised strategy under nonstationary tuning. Black curve shows the median RCE of 50 simulations, where adaptation was active between trial 1 and 1961 and stopped after that. Magenta curve shows the median RCE of 50 simulations, where adaptation was active *both* between trial 1 and 1961 and after trial 2000. In both groups, adaptation was inactive between 1962 and 2000 for comparison purposes (A). Zoomed medians during the early learning phase (B), the first decoder-freeze (C), the late learning phase after the first freeze (D) and the second decoder-freeze (E). The distributions of the relative cumulative errors for each of the phases (F, G, H, I). The rightmost values of the distribution plots denote the total relative counts of the outlier values that are greater than the associated x-axis value. Outliers correspond to the failed trajectories in figure 11.

unsupervised adaptive decoders was compared to another group of 50 randomly initialized *unsupervised* decoders, for which adaptation was stopped after a certain amount of trials. Both decoder groups were adaptive for the first 1961 trials, at the end of which they reached a stationary performance (figure 10A,B,F). Then, the adaptation of both groups was switched off during trials 1962 to 2000 (1st freeze) to compare the baseline performance of both decoder groups (figure 10C,G). As expected, both groups performed equally well during the first 2000 trials ($p > 0.1$, Wilcoxon rank sum test). For the first group, the adaptation was then switched on again for the next 1462 trials, whereas for the other group the adaptation was kept
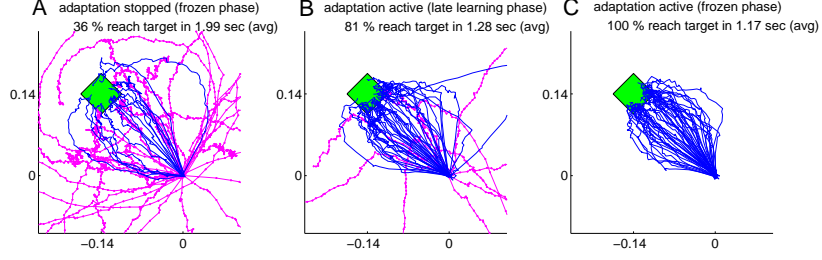
FIGURE 11. The trajectories under nonstationary tuning for un-
supervised strategy and their variations during late learning and
during decoder-freeze. Magenta thick curves indicate the failed
trajectories. Each plot depicts the trajectories of 50 training sim-
ulations. Trajectories during decoder-freeze belong to trial 3501.
The late learning trajectories were recorded at trial 3402, durations
and target hit rates are from pooled trajectories of trials 3402-3406.
The 50 different targets and trajectories at the recorded trial are ro-
tated to the same orientation ($\frac{3\pi}{4}$) for a better visual evaluation.

off. Evidently, non-adaptive decoders could not cope with the changing tuning any-
more and the performance strongly degraded (figures 10A,D,H and 11A). Adaptive
decoders, in opposite, tracked the changes in B_u well and kept the performance sta-
ble (figure 10A,D,H). After trial 3462 a 2nd freeze phase of 39 trials was used to
quantify the difference in performance between both groups for nonstationary tun-
ing parameters: adaptive decoders yielded a significantly ($p < 0.0001$ Wilcoxon
rank sum test) and about 7 times smaller error than non-adaptive decoders (fig-
ure 10A,E,I). In these simulations, we employed the unsupervised decoder cost as
in equation 7. The simulation settings are the same as described at the beginning
of section 3 except for λ. Here, we set $\lambda = 0.995$, to reduce the influence of the
earlier trials relative to the recent ones. This improves performance as recent tri-
als contain relatively more information on B_u. Trajectories of the adaptive group
reach very accurately to the target during decoder-freeze (figure 11C) and less but
also with high accuracy during the late learning phase (figure 11B).

3.5. **Different decoder costs for unsupervised adaptation.** Figure 12B shows
the trajectories obtained by 50 simulations of the unsupervised algorithm using
deviation cost (equation 8) during decoder-freeze. The trajectories were precise
and fast.

The trajectories obtained using amplitude + deviation cost (equation 9) are shown
in figure 12C. Again, straight and fast movements were obtained. A comparison
between the three unsupervised cost functions is presented in figure 13. All these
three costs yielded equal performance ($p > 0.05$ Wilcoxon rank sum test) during
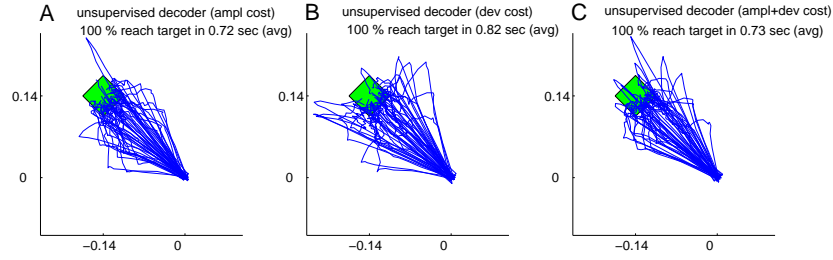all phases.

FIGURE 12. The trajectories for different cost measures and their variations during decoder-freeze. Magenta thick curves indicate the failed trajectories. Each plot depicts the trajectories of 50 training simulations, each at trial 1501. Note that not only trial 1501 included 50 simulations, but the whole history of 1501 trials are simulated 50 times with random initial tunings. The 50 different targets and trajectories at trial 1501 are rotated to the same orientation ($\frac{3\pi}{4}$) for a better visual evaluation.

4. DISCUSSION

Our results show that under realistic conditions, adaptive BMI decoding starting with random tuning parameters is feasible without an explicit supervision signal. Decoding performance gradually improves across trials and reaches a value close to the maximum possible performance as obtained by a supervised adaptive decoder, which assumes perfect knowledge of the intended movement of the BMI user. Moreover, we propose an adaptive decoder which employs neuronal error signals and show that this decoder yields a similar performance to our unsupervised adaptive decoder. Unsupervised and error-signal based decoders adapt rapidly and generate precise movement trajectories to the target. The suggested decoders do not require a supervision signal, e.g. the intended movement, and therefore can be used during autonomous BMI control. The suggested unsupervised adaptation is based on the minimization of a simple cost function, which penalizes high control signals and/or high variability of the neuronal control signals. The rationale behind these costs is, that inaccurate decoding causes corrective attempts by the BMI user, which in turn increase control signals and control signal variability. Therefore, accurate movement decoding corresponds to lower costs and the minimization of the suggested cost functions improves the accuracy of BMI movement control. Due to the generality of this approach we expect this to work in different kinds of motor tasks and not only for the reaching task considered in our simulations. Note that the cost function could be alternatively derived from only trajectories instead of control signals (e.g. deviations from straight line could be punished). An additional argument in favor of our cost functions comes from behavioral studies of human motor control: A wide range of human motor behavior can be described by optimal feedback control (OFC) models, which minimize cost functions containing
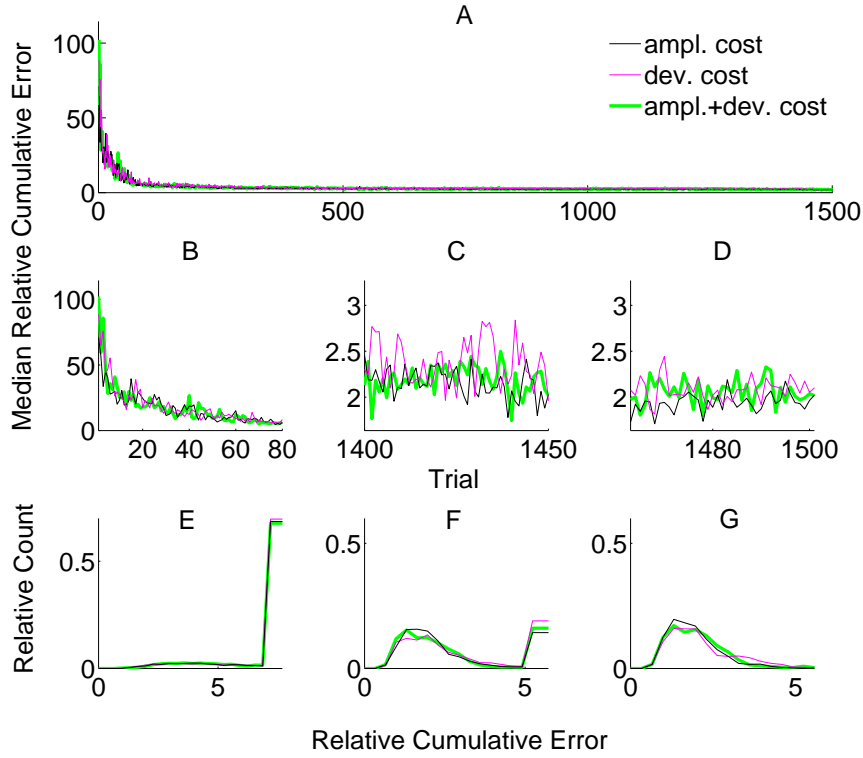
FIGURE 13. Comparison of relative cumulative errors for the un-supervised strategy using different cost measures. The algorithm employs either amplitude cost (black), or deviation cost (magenta) or their combination (green). Medians of relative cumulative errors of 50 simulations from each group for all of the trials (A). Zoomed medians for early (B) and late (C) learning and during decoder-freeze (D). The distributions of the relative cumulative errors for each of the phases (E, F, G). The rightmost values of the distribution plots denote the total relative counts of the outlier values that are greater than the associated x-axis value. The number of outlier values decreased across trials, i.e. it was the highest during early learning and zero during decoder-freeze.

the same dependence on the motor control signals as we used in our decoder cost [44, 48, 51].

Besides adaptation to unknown but static neuronal tuning to movement, we demonstrated that the proposed algorithms can also keep the decoding performance stable for nonstationary tuning. This is even possible if the tuning is not only non-stationary but also initially unknown. In these simulations, we assumed that the nonstationarity of the tuning parameters follows a random walk model and, hence, is independent of the decoder. If the decoded movement is fed back to the BMI user, the neuronal signals might adapt [52] and the learning speed as well as the final accuracy might even increase beyond the presented values.

In order to train the decoder, we assumed a log-linear model that relates the decoder parameters to cost via meta-parameters. We introduced a learning algorithm, which explores the parameter space with a ϵ-greedy policy. Our method performs least squares regression recursively to estimate the optimal values of the meta-parameters. In other words, the algorithm performs simultaneous exploration of the decoding parameters and recursive least squares (RLS) [50] regression on the decoder cost function. The same algorithm works also with neuronal error signals, where the cost is the number of error signals detected in a given time period. Error related neuronal activity has indeed been recorded from the brain via EEG [35, 36, 27], functional magnetic resonance imaging (fMRI)[26] and single-unit electrophysiology [53, 54]. Here, we assume a simple partially reliable error signal that indicates a substantial deviation from the movement intention. Neuronal activity related to this kind of movement execution errors has been found in ECoG [28] and in fMRI [26]. Milekovic et al. [28] observed neuronal responses evoked by a $180°$ degree movement mismatch during continuous joystick movement in 1-dimension. In our simulations of 2-dimensional movements, we assumed that neuronal error signals are evoked when the deviation between intended and decoded movement exceeds the somewhat arbitrary threshold of $20°$. Although it remains to be shown in future studies that neuronal error signals are indeed observable already at this threshold, we consider this a plausible assumption and expect our algorithm to be robust against the exact value of the threshold. Our results show that the overall performance of our algorithm is robust against different parameter update periods (T) and different error signal reliabilities ($> 50\%$). Arguably, the proposed algorithm has the potential to work with various types of neuronal error signals, though the computation of the cost function in terms of error signals might need adjustments to achieve high performance.

An alternative to our algorithm would be to use standard reinforcement learning algorithms and generalization methods [55] for directly training the decoding parameters without using a meta-parametric model relating cost to decoding parameters. In our practical experience, keeping a record of the previously explored parameters via P matrix of the RLS algorithm and relating the parameters to the $\log$-cost yields good performance. A comparison of our method to different reinforcement algorithms that utilize the same cost and/or other cost functions than the ones suggested here, are interesting topics for future studies. Previously, Kalman filtering methods were also applied for unsupervised adaptation during trajectory decoding [40, 41, 42, 56]. These methods adapt by maintaining consistency between a model of movement kinematics and a neuronal encoding model. They have been shown to track nonstationarities once an initial model is learned via supervised calibration [40, 41, 42]. Our unsupervised approach in this work is fundamentally different from these methods. We assume that, in the aftermath to decoding errors, the statistics of the control signals change and this change is utilized for unsupervised adaptation. In the future, it would be worthwhile to compare the performance of these different methods and their robustness against model violations in online BMI taks.

In principle, our adaptive decoding framework is independent of the type of neuronal signal that is used to control the movement. As neuronal control commands, the instantaneous firing rates of multiple single-unit or multi-unit activities could be used. Alternatively, filtered LFP, ECoG, EEG or MEG signals or the power

of LFP, ECoG, EEG and MEG signals in different frequency bands could be employed. Our algorithms assume that neuronal control signals are linearly related to movement velocity. For many different neuronal signal types, indeed, movement trajectories can be reconstructed well using this assumption ([6, 57, 21] for SUA, [14, 13] for ECoG). Linear tuning to movement position or simultaneous linear tuning to position and velocity can easily be implemented in our algorithms by straightforward modifications of the B matrices (see section 2). Future extension of our algorithmic framework might also consider nonlinear tuning. The cost measures we introduced, might need some modifications depending on the tuning of the recorded signals. For instance, if the control signal, e.g. firing rates for individual recording channels, takes an all-or-none behavior, i.e. certain channels are *on* for one direction and *off* for another direction, the norms of the command vectors might hardly vary across different movement directions. In such a case, deviation cost might be preferable over amplitude cost.

REFERENCES

[1] Miguel A L Nicolelis. Brain-machine interfaces to restore motor function and probe neural circuits. *Nat Rev Neurosci*, 4(5):417–422, May 2003.

[2] Mikhail A Lebedev and Miguel A L Nicolelis. Brain-machine interfaces: past, present and future. *Trends Neurosci*, 29(9):536–546, Sep 2006.

[3] John P Donoghue. Connecting cortex to machines: recent advances in brain interfaces. *Nat Neurosci*, 5 Suppl:1085–1088, Nov 2002.

[4] Jonathan R Wolpaw, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, and Theresa M Vaughan. Brain-computer interfaces for communication and control. *Clin Neurophysiol*, 113(6):767–791, Jun 2002.

[5] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, and H. Flor. A spelling device for the paralysed. *Nature*, 398(6725):297–298, Mar 1999.

[6] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan, and M. A. Nicolelis. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. *Nature*, 408(6810):361–365, Nov 2000.

[7] Leigh R Hochberg, Mijail D Serruya, Gerhard M Friehs, Jon A Mukand, Maryam Saleh, Abraham H Caplan, Almut Branner, David Chen, Richard D Penn, and John P Donoghue. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature*, 442(7099):164–171, Jul 2006.

[8] Carsten Mehring, Jörn Rickert, Eilon Vaadia, Simone Cardosa de Oliveira, Ad Aertsen, and Stefan Rotter. Inference of hand movements from local field potentials in monkey motor cortex. *Nat Neurosci*, 6(12):1253–1254, Dec 2003.

[9] Hansjörg Scherberger, Murray R Jarvis, and Richard A Andersen. Cortical local field potential encodes movement intentions in the posterior parietal cortex. *Neuron*, 46(2):347–354, Apr 2005.

[10] Eric C Leuthardt, Gerwin Schalk, Jonathan R Wolpaw, Jeffrey G Ojemann, and Daniel W Moran. A brain-computer interface using electrocorticographic signals in humans. *J Neural Eng*, 1(2):63–71, Jun 2004.

[11] Carsten Mehring, Martin Paul Nawrot, Simone Cardoso de Oliveira, Eilon Vaadia, Andreas Schulze-Bonhage, Ad Aertsen, and Tonio Ball. Comparing information about arm movement direction in single channels of local and epicortical field potentials from monkey and human motor cortex. *J Physiol Paris*, 98(4-6):498–506, 2004.

[12] Tonio Ball, Andreas Schulze-Bonhage, Ad Aertsen, and Carsten Mehring. Differential representation of arm movement direction in relation to cortical anatomy and function. *J Neural Eng*, 6(1):016006, Feb 2009.

[13] G. Schalk, J. Kubnek, K. J. Miller, N. R. Anderson, E. C. Leuthardt, J. G. Ojemann, D. Limbrick, D. Moran, L. A. Gerhardt, and J. R. Wolpaw. Decoding two-dimensional movement trajectories using electrocorticographic signals in humans. *J Neural Eng*, 4(3):264–275, Sep 2007.

[14] Tobias Pistohl, Tonio Ball, Andreas Schulze-Bonhage, Ad Aertsen, and Carsten Mehring. Prediction of arm movement trajectories from ecog-recordings in humans. *J Neurosci Methods*, 167(1):105–114, Jan 2008.

[15] Jonathan R Wolpaw and Dennis J McFarland. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. *Proc Natl Acad Sci U S A*, 101(51):17849–17854, Dec 2004.

[16] Benjamin Blankertz, Guido Dornhege, Christin Schfer, Roman Krepki, Jens Kohlmorgen, Klaus-Robert Müller, Volker Kunzmann, Florian Losch, and Gabriel Curio. Boosting bit rates and error detection for the classification of fast-paced motor commands based on single-trial eeg analysis. *IEEE Trans Neural Syst Rehabil Eng*, 11(2):127–131, Jun 2003.

[17] Stephan Waldert, Hubert Preissl, Evariste Demandt, Christoph Braun, Niels Birbaumer, Ad Aertsen, and Carsten Mehring. Hand movement direction decoded from meg and eeg. *J Neurosci*, 28(4):1000–1008, Jan 2008.

[18] Meel Velliste, Sagi Perel, M. Chance Spalding, Andrew S Whitford, and Andrew B Schwartz. Cortical control of a prosthetic arm for self-feeding. *Nature*, 453(7198):1098–1101, Jun 2008.

[19] Karunesh Ganguly and Jose M Carmena. Emergence of a stable cortical map for neuroprosthetic control. *PLoS Biol*, 7(7):e1000153, Jul 2009.

[20] G. Schalk, K. J. Miller, N. R. Anderson, J. A. Wilson, M. D. Smyth, J. G. Ojemann, D. W. Moran, J. R. Wolpaw, and E. C. Leuthardt. Two-dimensional movement control using electrocorticographic signals in humans. *J Neural Eng*, 5(1):75–84, Mar 2008.

[21] Dawn M Taylor, Stephen I Helms Tillery, and Andrew B Schwartz. Direct cortical control of 3d neuroprosthetic devices. *Science*, 296(5574):1829–1832, Jun 2002.

[22] Julie Blumberg, Jörn Rickert, Stephan Waldert, Andreas Schulze-Bonhage, Ad Aertsen, and Carsten Mehring. Adaptive classification for brain computer interfaces. *Conf Proc IEEE Eng Med Biol Soc*, 2007:2536–2539, 2007.

[23] C. Vidaurre, M. Kawanabe, P. von Bünau, B. Blankertz, and K. R. Müller. Toward unsupervised adaptation of lda for brain-computer interfaces. *IEEE Trans Biomed Eng*, 58(3):587–597, Mar 2011.

[24] Carmen Vidaurre, Claudia Sannelli, Klaus-Robert Mller, and Benjamin Blankertz. Co-adaptive calibration to improve bci efficiency. *J Neural Eng*, 8(2):025009, Apr 2011.

[25] Carmen Vidaurre, Claudia Sannelli, Klaus-Robert Mller, and Benjamin Blankertz. Machine-learning-based coadaptive calibration for brain-computer interfaces. *Neural Comput*, Dec 2010.

[26] Jörn Diedrichsen, Yasmin Hashambhoy, Tushar Rane, and Reza Shadmehr. Neural correlates of reach errors. *J Neurosci*, 25(43):9919–9931, Oct 2005.

[27] Olav E Krigolson, Clay B Holroyd, Geraldine Van Gyn, and Mathew Heath. Electroencephalographic correlates of target and outcome errors. *Exp Brain Res*, 190(4):401–411, Oct 2008.

[28] Tomislav Milekovic, Tonio Ball, Andreas Schulze-Bonhage, Ad Aertsen, and Carsten Mehring. Error-related electrocorticographic activity in humans during continuous movements. *J Neural Eng*, 9(2):026007, Feb 2012.

[29] Dennis J McFarland, William A Sarnacki, and Jonathan R Wolpaw. Electroencephalographic (eeg) control of three-dimensional movement. *J Neural Eng*, 7(3):036007, Jun 2010.

[30] C. Vidaurre, A. Schlogl, R. Cabeza, R. Scherer, and G. Pfurtscheller. A fully on-line adaptive bci. *IEEE Trans Biomed Eng*, 53(6):1214–1219, 2006.

[31] C. Vidaurre, A. Schlögl, R. Cabeza, R. Scherer, and G. Pfurtscheller. Study of on-line adaptive discriminant analysis for eeg-based brain computer interfaces. *IEEE Trans Biomed Eng*, 54(3):550–556, Mar 2007.

[32] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. C. Principe, and J. C. Sanchez. Coadaptive brain–machine interface via reinforcement learning. *IEEE Trans Biomed Eng*, 56(1):54–64, 2009.

[33] Justin C Sanchez, Babak Mahmoudi, Jack DiGiovanna, and Jose C Principe. Exploiting co-adaptation for the design of symbiotic neuroprosthetic assistants. *Neural Netw*, 22(3):305–315, Apr 2009.

[34] Gregory J Gage, Kip A Ludwig, Kevin J Otto, Edward L Ionides, and Daryl R Kipke. Naive coadaptive cortical control. *J Neural Eng*, 2(2):52–63, Jun 2005.

[35] M. Falkenstein, J. Hoormann, S. Christ, and J. Hohnsbein. Erp components on reaction errors and their functional significance: a tutorial. *Biol Psychol*, 51(2-3):87–107, Jan 2000.

[36] W. J. Gehring, B. Goss, M. G. H. Coles, D. E. Meyer, and E. Donchin. A neural system for error detection and compensation. *Psychological Science*, 4(6):385–390, 1993.

[37] Lucas C Parra, Clay D Spence, Adam D Gerson, and Paul Sajda. Response error correction–a demonstration of improved human-machine performance using real-time eeg monitoring. *IEEE Trans Neural Syst Rehabil Eng*, 11(2):173–177, Jun 2003.

[38] David Rotermund, Udo A Ernst, and Klaus R Pawelzik. Towards on-line adaptation of neuro-prostheses with neuronal evaluation signals. *Biol Cybern*, 95(3):243–257, Sep 2006.

[39] Babak Mahmoudi and Justin C Sanchez. A symbiotic brain-machine interface through value-based decision making. *PLoS One*, 6(3):e14760, 2011.

[40] U. Eden, W. Truccolo, M. Fellows, J. Donoghue, and E. Brown. Reconstruction of hand move-ment trajectories from a dynamic ensemble of spiking motor cortical neurons. *Conf Proc IEEE Eng Med Biol Soc*, 6:4017–4020, 2004.

[41] Uri T Eden, Loren M Frank, Riccardo Barbieri, Victor Solo, and Emery N Brown. Dynamic analysis of neural encoding by point process adaptive filtering. *Neural Comput*, 16(5):971–998, May 2004.

[42] Yiwen Wang and Jose C Principe. Tracking the non-stationary neuron tuning by dual kalman filter for brain machine interfaces decoding. *Conf Proc IEEE Eng Med Biol Soc*, 2008:1720–1723, 2008.

[43] Emanuel Todorov. Optimality principles in sensorimotor control. *Nat Neurosci*, 7(9):907–915, Sep 2004.

[44] Emanuel Todorov and Michael I Jordan. Optimal feedback control as a theory of motor coordi-nation. *Nat Neurosci*, 5(11):1226–1235, Nov 2002.

[45] Robert F. Stengel. *Optimal Control and Estimation*. Dover, New York, 1994.

[46] Emanuel Todorov and Michael I. Jordan. A minimal intervention principle for coordinated movement. In *In*, pages 27–34. MIT Press, 2002.

[47] Jun Izawa, Tushar Rane, Opher Donchin, and Reza Shadmehr. Motor adaptation as a process of reoptimization. *J Neurosci*, 28(11):2883–2891, Mar 2008.

[48] Daniel A Braun, Ad Aertsen, Daniel M Wolpert, and Carsten Mehring. Learning optimal adap-tation strategies in unpredictable motor tasks. *J Neurosci*, 29(20):6472–6478, May 2009.

[49] Emanuel Todorov. Stochastic optimal control and estimation methods adapted to the noise char-acteristics of the sensorimotor system. *Neural Comput*, 17(5):1084–1108, May 2005.

[50] B. Farhang-Boroujeny. *Adaptive Filters: Theory and Applications*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1999.

[51] Jörn Diedrichsen, Reza Shadmehr, and Richard B Ivry. The coordination of movement: optimal feedback control and beyond. *Trends Cogn Sci*, 14(1):31–39, Jan 2010.

[52] Beata Jarosiewicz, Steven M Chase, George W Fraser, Meel Velliste, Robert E Kass, and An-drew B Schwartz. Functional network reorganization during learning in a brain-computer inter-face paradigm. *Proc Natl Acad Sci U S A*, 105(49):19486–19491, Dec 2008.

[53] Shigehiko Ito, Veit Stuphorn, Joshua W Brown, and Jeffrey D Schall. Performance monitoring by the anterior cingulate cortex during saccade countermanding. *Science*, 302(5642):120–122, Oct 2003.

[54] Madoka Matsumoto, Kenji Matsumoto, Hiroshi Abe, and Keiji Tanaka. Medial prefrontal cell activity signaling prediction errors of action values. *Nat Neurosci*, 10(5):647–656, May 2007.

[55] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, March 1998.

[56] Wei Wu and Nicholas G Hatsopoulos. Real-time decoding of nonstationary neural activity in motor cortex. *IEEE Trans Neural Syst Rehabil Eng*, 16(3):213–222, Jun 2008.

[57] Mijail D Serruya, Nicholas G Hatsopoulos, Liam Paninski, Matthew R Fellows, and John P Donoghue. Instant neural control of a movement signal. *Nature*, 416(6877):141–142, Mar 2002.