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

Learning to use a Brain-Machine Interface: Model, Simulation and Analysis

Jessica Jimenez, *Member, IEEE*, Rodolphe Heliot, *Member, IEEE*,
and Jose M. Carmena, *Senior Member, IEEE*

Abstract— **This paper presents a model of the learning process occurring during operation of a closed-loop brain-machine interface. The model consists of a population of simulated cortical neurons, a decoder that transforms neural activity into motor output, a feedback controller whose role is to reduce the error based on an error-descent algorithm, and an open-loop controller whose parameters are updated based on the corrections made by the feedback controller. We present evidence of the convergence of the internal model to the decoder's inverse model and use global sensitivity analysis to study the convergence's dependence on the parameters of the overall learning model. This model can be used as a simulation tool that predicts the outcome of closed-loop BMI experiments.**

I. Introduction

BRAIN-MACHINE interfaces (BMIs) are direct communication pathways between the brain and an external device, such as a computer cursor or a robot. A decoder transforms the recorded neural signals to motor commands that are used to control the external device, and information about the performance is given back to subject in some form of feedback, usually visual. Recent demonstrations in non-human primates have successfully achieved closed-loop BMI control of a mechanical arm in 2 and 3-dimensional space [1], [2].

In closed-loop BMI systems the decoder is usually obtained by recording neural signals of the subject during a manual task and then using these signals in conjunction with the recorded kinematics to solve a statistical inference problem, It has been shown that simple decoders have good predictive power in offline (i.e. open loop) simulations [7] although this performance is not guaranteed when switching to closed-loop BMI mode. This is because the brain changes its neural mapping while learning to control the BMI [4] and other factors out of the scope of this paper.

Here we present a model of the learning process that occurs while learning to use a closed-loop BMI system. The model represents the brain as a dynamical system that adapts using the feedback information. Evidence that supports the control and learning capabilities of the system is presented.

Our goal is to study the underlying properties of the learning process, and see how the knowledge of these properties can be used to increase the performance of closed-loop BMI systems.

II. Learning Model

A. Global description

Several studies investigating the mechanisms used in motor learning suggest that the brain builds an inverse model of the system it is trying to control [5]-[8]. Based on these studies, it is reasonable to think that the brain encodes a model of the decoder and plant when it is learning to use a closed-loop BMI system.

In the case considered in this paper the decoder of the closed-loop BMI system transforms the firing rate of an ensemble of neurons from motor cortex, $\vec{f}$, into the position of a computer cursor $Y = D(\vec{f})$. Here $\vec{f} = (f_1, f_2, \dots f_N)^T$, where f_i is the firing rate of cell i and N is the total number of neurons used by the decoder. These firing rates are generated by the brain based on its inputs: the desired position and the sensory feedback information, i.e. actuator's position. Therefore,

$$\vec{f} = B_\lambda(T, Y), \tag{1}$$

where λ is a set of parameters that change as the brain is learning to control the BMI system.

The internal model B_λ is equivalent to a feedforward controller, and is coupled with a feedback controller as shown in Fig. 1. The internal model generates the ensemble firing rates that will lead to the targeted position whereas the feedback controller compensates for errors in the internal model. The motivation and description of each part of this model is included in the next two subsections.

B. Feedback Controller

We can think of the BMI learning process as an iterative optimization algorithm that minimize the error distance e, i.e. the distance between the target and the output position, which is given by:

$$e = \|Y - T\|^2 \tag{2}$$

The classical way of solving this problem is by updating the parameters of the system based on the gradient of the error distance with respect to the parameters. Unfortunately, we cannot model the brain's learning process in such a way

Manuscript received April 23, 2009. This work was supported in part by the French "Délégation générale pour l'armement" under Grant N° PDE 07C0067 to RH, and the Christopher and Dana Reeve foundation to JMC.

J. Jiménez is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA.

R. Héliot is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA.

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

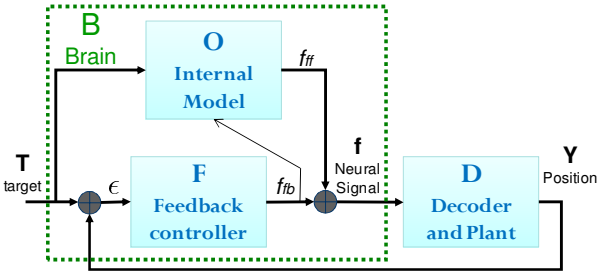

Fig. 1. The brain is modeled as an adaptive system with two components: a feedback controller and a feedforward controller that represents the inverse internal model of the external device.

because that will imply that the brain knows the external transformation D. An alternative method is to use an error-descent optimization algorithm like the one presented in [9]. This error descent algorithm uses the output's change produced by small random changes in the parameters to estimate the gradient. This approach seems suitable for this problem since the neuronal firing rate is a noisy process.

We used the error-descent algorithm as a feedback controller that reduces the error distance and compensates for the errors in the internal model. This algorithm works as follows: a small and random perturbation $\vec{g}$ is added to the feedback signal $\vec{f}_{fb}$ and the error distance (2) is calculated before and after the perturbation $\vec{g}$ is added. These error distances are denoted by e_f and e_{fg} respectively. As shown in [9], we update the feedback signal using the update rule given by:

$$\begin{aligned} \vec{f}_{fb} &= \vec{f}_{fb} + \vec{\Delta}_f \\ \vec{\Delta}_f &= -\mu\,(e_{fg} - e_f)\vec{g} \end{aligned} \tag{3}$$

where μ is a small positive constant. Since numerous cells are involved in the model, but a single error measurement is available, the update rule (3) does not guarantee that the change in feedback goes in the right direction for a given cell at a given update step. However, the algorithm statistically leads to error descent over the whole population. Although, this feedback controller is capable of reducing the error, by itself it does not provide a long term learning mechanism. For this reason an adaptive feedforward model is necessary.

C. Adaptive Internal Model

As mentioned before, the internal model predicts the firing rates that should lead to the desired position. Therefore the firing rates produced by the internal model depend on the target position T and are given by the transformation:

$$\vec{f}_{ff} = O_\lambda(T), \tag{4}$$

where λ is a set of parameters of the internal model. In theory, any transformation O_λ can be used as the internal model. In this paper, as in [10], we choose an affine transformation given by:

$$\vec{f}_{ff} = A\,T + \vec{b}. \tag{5}$$

Here $\lambda = \{A, \vec{b}\}$, where $\vec{b}$ is a vector whose components are the baseline firing rates of each neuron and A is a matrix whose rows are the weights used for modulation of the firing rates of each individual neuron.

The parameters adapt as the brain learns the inverse model of the external system. As first suggested by Kawato [6], the feedback signal is used to update the internal model as it is representative of the internal model's error. We update the parameters in such a way that the feed-forward contribution of the firing rates, $\vec{f}_{ff}$, change in the same direction as the feedback update. Indeed, this direction is the one for which the error is reduced. An update rule that satisfies this condition is given by:

$$\begin{aligned} A &= A + v\,\vec{\Delta}_f\,\frac{T'}{||T||} \\ \vec{b} &= \vec{b} + v\,\vec{\Delta}_f \end{aligned} \tag{6}$$

where v is the learning speed. Besides reducing the error, if a perfect inverse model is reached, this update leads to zero error hence zero feedback, and zero update of the internal model's parameters. Step by step, the inverse model is built by small increments in the parameter space resulting in decreasing error. It should be noted that although we chose to use a linear internal model, the same scheme can be applied to virtually any internal model.

III. Simulation

Closed-loop BMI experiments were simulated using the system previously described. We used a linear decoder for the simulation, i.e.:

$$Y = D\vec{f} \tag{7}$$

where D is a 2-by-N matrix converting the firing rates into the 2-dimensional position of the BMI-controlled cursor. In our simulations, a number $N = 10$ cells was used. Initial parameters A and $\vec{b}$ of simulated cells were chosen randomly. The task to be performed by the simulated closed-loop BMI involved reaches to peripheral targets from a center position, then returning to the center. Targets were held for at most M time samples. A trial was successful if the target was reached before this maximal duration, in which case a new target was presented. Otherwise, a new target was presented after M samples. These simulations are used for the numerical analysis portion of this paper. For further information on the learning model and its validation see [11].

IV. Analysis

In this section we analyze the BMI learning model introduced in the previous sections. As mentioned in [7], it is important to study stability and convergence of the neural model in order to verify that its properties match the properties of the neural system under study. To study the

stability and convergence, we study input-output properties, as well as evidence that supports the convergence of the state to an inverse model. However, the state convergence properties of the model depend on a series of intrinsic parameters. In the last subsection, global sensitivity analysis is used to study this dependence.

A. *Input-Output Behavior*

Cauwenberghs in [9] proved that the error-descent optimization algorithm decreases the error, given by (2), to the optimal solution, provided that the perturbation magnitude $|\vec{g}|$ is small compared to the absolute value of the gradient error function that we are trying to minimize and that μ is positive and small. He also showed that the algorithm performs gradient descent on average with convergence rate $\mu_t = \mu\,\sigma^2$, given that components of the perturbation vector $\vec{g}$ are uncorrelated and with variance σ^2. This means that the error-descent algorithm, viewed as a feedback controller, is capable of reaching any input target. The required reaching time depends on the initial conditions and parameters of the system.

Since the feedback controller in our BMI learning model is coupled with an internal model, Cauwenberghs' global convergence result, mentioned above, cannot be directly applied to our model. Fortunately, it is easy to show that for our internal model and learning rule the internal model contribution to the firing rates equals $\vec{f}_{ff} = \vec{f}_{ff} + 2v\,\vec{\Delta}_f$. This implies that the overall neural signal update rule is given by:

$$\begin{aligned} \vec{f} &= \vec{f} + \vec{f}_\Delta \\ \vec{f}_\Delta &= (1+2v)\vec{\Delta}_f \\ &= -(1+2v)\mu\big(e_{fg} - e_f\big)\vec{g}. \end{aligned} \tag{8}$$

It follows that as with the original error-descent algorithm, the model performs gradient descent with respect to $\vec{f}$ on average.

This result implies that output $Y^{(k)} = D(\vec{f}^{(k)})$ tends to T as k tends to infinity, where k is a learning epoch. However, this does not imply that the state of the system, i.e. A, converges to a stable solution. In particular we want to know if the state $A^{(k)}$ converges to an inverse of D as k tends to infinity. This would mean that our learning model indeed builds an inverse model of the decoder.

B. *Convergence of the State to an Inverse Model*

If the open loop controller is capable of learning the inverse model of the decoder, then $DA^{(k)} \to I$ as $k \to \infty$, provided that the input targets span the two dimensional space. To prove convergence in quadratic mean we need to show that $E^{(k)} = E\big[\|DA^{(k)} - I\|\big] \to 0$ as $k \to \infty$. Ideally, we would like to find a closed-form expression of $E^{(k)}$ and prove its convergence under certain conditions on the parameters. However, it is not straightforward to derive this closed-form expression. Therefore, we must resort to numerical analysis.

Monte Carlo analysis was used to find estimates for $E^{(k)}$ given the configuration parameters $\theta = \{\mu, v, \sigma^2\}$, of the model, where σ^2 is the variance of the perturbation components. Note that $|\vec{g}|$ depends directly on σ^2 since the components of $\vec{g}$ are independent, zero-mean, and uniformly distributed random variables. Fig. 2(a) shows the results of the analysis for the configuration $\theta = \{0.8, 0.8, 0.01\}$. The blue line is the estimation and the red lines mark the confidence interval with coverage of 0.99. For this particular configuration, and many others, $E^{(k)}$ converges to zero. In addition the variance of the sequences tends to zero as k increases, as shown by the confidence interval converging with the estimation in Fig. 2 (a). To verify that the state stabilizes, we estimate the expected value of $A^{(k)}$, shown in Fig. 2 (b), and verify that it reaches steady state. This proves that the model is capable of converging to one of the decoder's inverses.

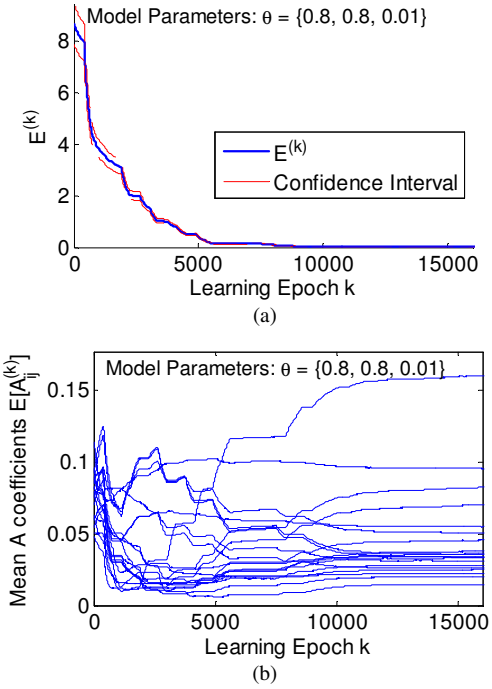

Fig. 2. Monte Carlo analysis of the learning model with parameters $\theta = \{0.8, 0.8, 0.01\}$. We used 25 replicas of the process, with random initial conditions and random input sequences. (a)Estimation of $E^{(k)}$ and confidence interval with a 0.99 coverage. (b) Mean A coefficients.

C. *Factor Mapping: Regionalized Sensitivity Analysis*

In sensitivity analysis, the factor mapping setting is associated with mapping portions of the output space to the factor (i.e. parameter) space. Here we apply this setting to study how the convergent behavior maps to the model's parameters μ, v and σ^2. The methods used in this section can be found in [12].

We used Monte Carlo filtering (MCF) to separate the convergent and divergent portions of our sample space. The parameters where sampled randomly with a uniform distribution with range $[0,10]$ for μ and v and $[0,0.07]$ for σ^2. These ranges were chosen in a way that the unconditional probability of convergence p is near 0.5, in

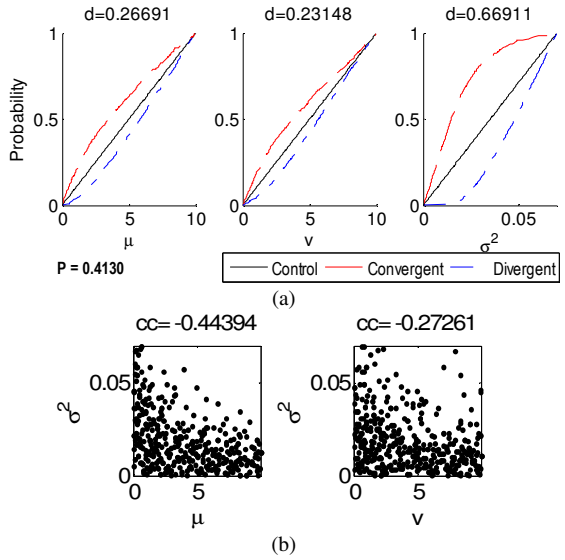

Fig. 3. Monte Carlo Filtering (a) CDF of the parameters conditioned to convergence, d is the Smirnov statistic; (b) Bidimensional projection of the different pairs of parameters (cc is the correlation coefficient).

this case $p = 0.413$. This was done to have enough samples in both the convergent and divergent subsets, and in that way avoid lack of statistical power. After MCF, Regionalized Sensitivity Analysis (RSA) is used to identify what parameters are more relevant to the state convergence.

First we computed the probability distribution of each parameter conditioned on the convergent and divergent subsets. We compared the conditional cumulative distributions functions (CDF) obtained for each parameter graphically and by computing the Smirnov test statistic $d = sup\|F(x|C) - F(x|\bar{C})\|$, where x is the parameter and C and $\bar{C}$ denote the convergent and divergent subsets respectively. These comparisons are shown in Fig. 3(a). High values of d indicate that the parameter is important for the state convergence. From the Smirnov statistic we see that while all the parameters have a significant effect on the convergence, σ^2 has a greater impact (i.e. much greater coefficient). From the curve of the convergent set's CDF, we can find the regions of the parameter space that have a greater probability of convergence. Convergence is mostly obtained for small values of the parameters, indicated by the steep slope of the convergent set's CDF in small regions.

We also studied the parameters' interaction mechanisms that produce state convergence. To do that, we found the correlation coefficient for the different pairs and plot the bidimensional projection of the pairs with high correlation, Fig 3(b). For the pair (μ, v) the correlation coefficient was not significantly high. For the other pairs, the curves have a negative correlation suggesting that the overall interaction mechanism between two parameters is a sum or a product. Since the curves resemble hyperboles it is plausible that the condition for state convergence has the form of $x_1 x_2 < constant$, for parameters $(x_1, x_2) \in \{(\mu, \sigma^2), (v, \sigma^2)\}$.

V. Discussion

This paper presented a model for simulating the learning process in a closed-loop BMI experiment. The adaptive scheme presented uses random perturbations of the firing rates to update an internal model. This makes it a biologically plausible model since the neuronal firing rates are inherently noisy. The analysis showed that the model is capable of learning an internal inverse model and that the ability of doing so depends greatly on the magnitude of the random changes. These results agree with the properties of real BMI systems. First, BMI performance is believed to depend on the ability of changing the neural mapping, which in our model is equivalent to adapting the internal model. Second, as mentioned in [13], fluctuations of the neural activity can be one of the mechanisms used to learn a new mapping and in our model the ability of having detailed neural fluctuation depends directly on the magnitude of the random changes, making it a very influential parameter. Our learning model could be improved with a more realistic internal model transformation, and by adding the delay in the feedback loop to account for delays in sensory afferents.

The model presented is useful to study BMI and predict its closed-loop behavior. This may help us to design new "optimal" decoders that would maximize the learning speed rather than the offline prediction power.

References

[1] Carmena, J.M., Lebedev, M.A., Crist, R.E., O'Doherty, J.E., Santucci, D.M., Dimitrov, D., Patil, P.G., Henriquez, C.S., and Nicolelis, M.A.L. "Learning to control brain-machine interface for reaching, grasping by primates," PloS Biol., vol. 1, pp. 193–208, 2003.

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

[3] Kim S.-P., Sanchez J.C., Rao Y.N., Erdogmus D., Carmena J.M., Lebedev M.A., Nicolelis M.A.L., and Principe J.C. A comparison of optimal MIMO linear and nonlinear models for brain-machine interfaces. Journal of Neural Engineering 3, pp. 145-161., 2006

[4] Eberhard E. Fetz, "Volitional control of neural activity: implications for brain–computer interfaces", J Physiol March 2007 579:571-579

[5] Doya, K. Kimura, H. Kawato, M., "Neural mechanisms of learning and control". Control Systems Magazine, IEEE, Vol. 21, Issue: 4 2001

[6] M. Kawato, K. Furukawa and R. Suzuki, "A hierarchical neural network model for control and learning of voluntary movement", Biol. Cybern., vol. 57, pp. 169-185, 1987

[7] Tin C., Poon C.S., Internal models in sensorimotor integration: perspectives from adaptive control theory. J Neural Eng 2 2005

[8] D. Wolpert, R. Miall, and M. Kawato, "Internal models in the cerebellum," Trends in Cognitive Science, vol. 2, pp. 338–347, 1998.

[9] G. Cauwenberghs, "A fast stochastic error-descent algorithm for supervised learning and optimization," in Adv. Neural Information Processing Systems (NIPS*92), vol. vol. 5, 1993, pp. pp. 244–251.

[10] U. Rokni, A. Richardson, E. Bizzi, and H. Seung, "Motor learning with unstable neural representations," Neuron, vol. 54, no. 4, pp. 653–666, 2007.

[11] Heliot R., Ganguly K., Carmena J.M. "Modeling and experimental validation of the learning process during closed-loop BMI operation", International Conference on Machine Learning and Cybernetics (ICMLC), Hebei, China, 2009 (to appear)

[12] A. Saltelli, et al., "Global Sensitivity Analysis. The Primer," John Wiley and Sons., 2008, chap. 5.

[13] X. Xie and H. S. Seung, "Learning in neural networks by reinforcement of irregular spiking," Physical Review E, vol. 69, no. 4, pp. 1–10, 2004.