Development/Plasticity/Repair

A Reward-Modulated Hebbian Learning Rule Can Explain Experimentally Observed Network Reorganization in a Brain Control Task

Robert Legenstein,[1] Steven M. Chase,[2,3,4] Andrew B. Schwartz,[2,3] and Wolfgang Maass[1]

[1]Institute for Theoretical Computer Science, Graz University of Technology, 8010 Graz, Austria, [2]Department of Neurobiology, University of Pittsburgh, Pittsburgh, Pennsylvania 15213, [3]Center for the Neural Basis of Cognition, University of Pittsburgh and Carnegie Mellon University, and [4]Department of Statistics, Carnegie Mellon University, Pittsburgh, PA 15213

It has recently been shown in a brain–computer interface experiment that motor cortical neurons change their tuning properties selectively to compensate for errors induced by displaced decoding parameters. In particular, it was shown that the three-dimensional tuning curves of neurons whose decoding parameters were reassigned changed more than those of neurons whose decoding parameters had not been reassigned. In this article, we propose a simple learning rule that can reproduce this effect. Our learning rule uses Hebbian weight updates driven by a global reward signal and neuronal noise. In contrast to most previously proposed learning rules, this approach does not require extrinsic information to separate noise from signal. The learning rule is able to optimize the performance of a model system within biologically realistic periods of time under high noise levels. Furthermore, when the model parameters are matched to data recorded during the brain–computer interface learning experiments described above, the model produces learning effects strikingly similar to those found in the experiments.

Introduction

Recent advances in microelectrode recording technology make it possible to sample the neural network generating behavioral output with brain–computer interfaces (BCIs). Monkeys using BCIs to control cursors or robotic arms improve with practice (Taylor et al., 2002; Carmena et al., 2003; Musallam et al., 2004; Schwartz, 2007; Ganguly and Carmena, 2009), indicating that learning-related changes are funneling through the set of neurons being recorded. In a recent report (Jarosiewicz et al., 2008), adaptation-related changes in neural firing rates were systematically studied in a series of BCI experiments. In that work, monkeys used motor cortical activity to control a cursor in a three-dimensional (3D) virtual reality environment during a center-out movement task. When the activity of a subset of neurons was decoded incorrectly, to produce cursor movement at an angle to the intended movement, the tuning properties of that subset changed significantly more than for the subset of neurons for which activity was decoded correctly. This experiment demonstrated that motor cortical neurons may be able to solve the "credit assignment" problem: using only the global feedback of cursor movement, the subset of cells contributing more to cursor error underwent larger tuning changes. This adaptation strategy is quite surprising, since it is not clear how a learning mechanism is able to determine which subset of neurons needs to be changed.

In this article, we propose a simple biologically plausible reinforcement learning rule and apply it to a simulated 3D reaching task similar to the task in the study by Jarosiewicz et al. (2008). This learning rule is reward-modulated Hebbian: weight changes at synapses are driven by the correlation between a global reward signal, presynaptic activity, and the difference of the postsynaptic potential from its recent mean (Loewenstein and Seung, 2006). An important feature of the learning rule proposed in this article is that noisy neuronal output is used for exploration to improve performance. We show that large amounts of noise are beneficial for the adaptation process but not problematic for the readout system. In contrast to most other proposed reward-modulated learning rules, the version of the reward-modulated Hebbian learning rule that we propose does not require any external information to differentiate internal noise from synaptic input. We demonstrate that this learning rule is capable of optimizing performance in a neural network engaging in a simulated 3D reaching task. Furthermore, when compared to the results of Jarosiewicz et al. (2008), the simulation matches the differential-learning effects they report. Thus, this study shows that noise-driven learning can explain detailed experimental results about neuronal tuning changes in a motor control task and suggests that the corresponding reward modulation of the learning process acts on specific subpopulations as an essential cortical mechanism for the acquisition of goal-directed behavior.

We consider several of the sections within Materials and Methods, below, to be crucial to understanding the results.

Received Aug. 14, 2009; revised Dec. 4, 2009; accepted Dec. 7, 2009.

This work was supported by the Austrian Science Fund FWF (S9102-N13, to R.L. and W.M.), the European Union [FP6-015879 (FACETS), FP7-506778 (PASCAL2), FP7–231267 (ORGANIC) to R.L. and W.M.], and the National Institutes of Health (R01-NS050256, EB005847, to A.B.S.).

Correspondence should be addressed to Robert Legenstein, Institute for Theoretical Computer Science, Graz University of Technology, Inffeldgasse 16b, 8010 Graz, Austria. E-mail: legi@igi.tugraz.at.

DOI:10.1523/JNEUROSCI.4284-09.2010

 0270-6474/10/308400-11$15.00/0

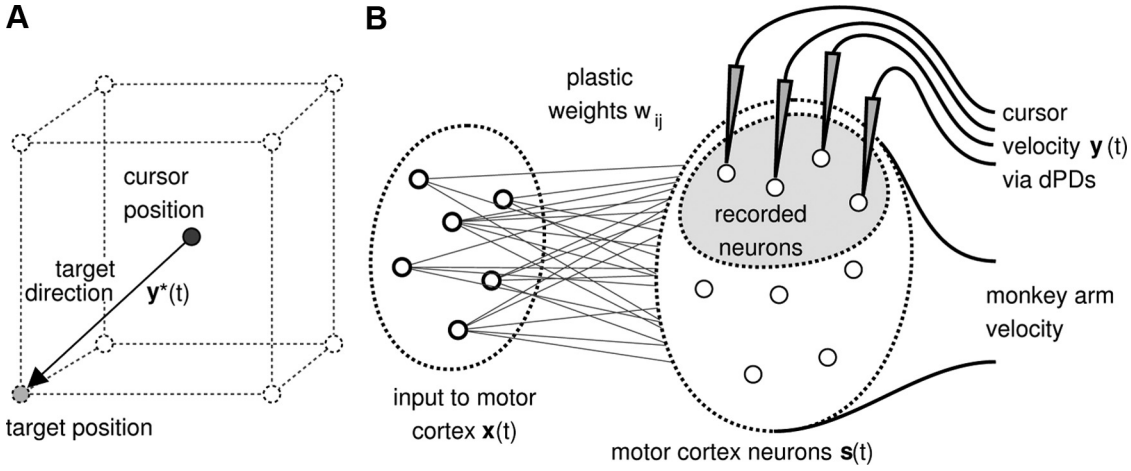

Figure 1. Description of the 3D cursor control task and network model for cursor control. ***A***, The task was to move the cursor from the center of an imaginary cube to one of its eight corners. The target direction $\mathbf{y}^*(t)$ was given by the direction of the straight line from the current cursor position to the target position. ***B***, Schematic of the network model used for the cursor control task. A set of m neurons project to n_{total} noisy neurons in motor cortex. The monkey arm movement was modeled by a fixed linear mapping from the activities of the modeled motor cortex neurons to the 3D velocity vector of the monkey arm. A subset of n neurons in the simulated motor cortex was recorded for cursor control. The velocity of the cursor movement at time t was given by the population vector, which is the vector sum of decoding PDs of recorded neurons weighted by their normalized activities.

Materials and Methods

The Materials and Methods are structured as follows. The experiments of Jarosiewicz et al. (2008) are briefly summarized in the following section, Experiment: learning effects in monkey motor cortex. Simulation methods that we consider to be crucial to understanding the Results are given in the section Simulation. Simulation details that are needed for completeness, but not necessary for all readers, are given in the section Simulation details.

Experiment: learning effects in monkey motor cortex

This section briefly describes the experiments of Jarosiewicz et al. (2008); a more complete description can be found in the original work. To extract intended movement from recorded neuronal activity in motor cortex, the firing rate of each neuron was fit as a function of movement direction using a cosine tuning curve (Georgopoulos et al., 1986; Schwartz, 2007). The preferred direction (PD) of the neuron was defined as the direction in which the cosine fit to its firing rate was maximal, and the modulation depth was defined as the difference in firing rate between the maximum of the cosine fit and the baseline (mean). The monkey's intended movement velocity was extracted from the firing rates of a group of recorded units by computing the weighted sum of their PDs, where each weight was the unit's normalized firing rate, i.e., by the population vector algorithm (Georgopoulos et al., 1988). (Note that units represented either well isolated single neurons or a small number of neurons that could not be reliably distinguished, but were nevertheless tuned to movement as a group.)

In the learning experiments, the monkey controlled a cursor in a 3D virtual reality environment. The task for the monkey was to move the cursor from the center of an imaginary cube to a target appearing at one of its corners. Each of the experiments consisted of a sequence of four brain control sessions: calibration, control, perturbation, and washout. The tuning functions of an average of 40 recorded units were first obtained in the calibration session, where an iterative procedure was used to obtain data for the linear regressions. These initial estimates of the PDs were later used for decoding neural trajectories into cursor movements. To distinguish between measured PDs and PDs used for decoding, we refer to the latter as "decoding PDs" (dPDs). In the control, perturbation, and washout sessions, the monkey had to perform a cursor control task in a 3D virtual reality environment (Fig. 1*A*). The cursor was initially positioned in the center of an imaginary cube; a target position on one of the corners of the cube was then randomly selected and made visible. When the monkey managed to hit the target position with the cursor (success), or a 3 s time period expired (failure), the cursor position was reset to the origin and a new target position was randomly selected from the eight corners of the imaginary cube. In the control session, the PDs measured during the calibration session were used as dPDs for cursor control. In the perturbation session, the dPDs of a randomly selected subset of units (25% or 50% of the recorded units) were altered from their control values by rotating them 90° around one of the x, y, or z axes (all PDs were rotated around a common axis in each experiment). In this article, we term these units "rotated" units. The other dPDs remained the same as in the control session. We term these units "nonrotated" units. In the subsequent washout session, the measured PDs were again used for cursor control.

In the perturbation session, the firing behavior of the recorded units changed to compensate for the altered dPDs. The authors observed differential effects of learning between the nonrotated and rotated groups of units. Rotated units tended to shift their PDs in the direction of dPD rotation, hence they compensated for the perturbation. The change of the PDs of nonrotated units was weaker and significantly less strongly biased toward the direction of rotation than the PDs of rotated units. We refer to this differential behavior of rotated and nonrotated units as the "credit assignment effect."

Simulation

Network model. Our aim was to explain the experimentally observed learning effects in the simplest possible model. This network model consisted of two populations of neurons connected in a feedforward manner (Fig. 1*B*). The first population modeled those neurons that provide input to the neurons in motor cortex. It consisted of $m = 100$ neurons with activities $x_1(t), \ldots, x_m(t) \in \mathbb{R}$. The second population modeled neurons in motor cortex that receive inputs from the input population. It consisted of $n_{\text{total}} = 340$ neurons with activities $s_1(t), \ldots, s_{n_{\text{total}}}(t)$. The distinction between these two layers is purely functional: input neurons may be situated in extracortical areas, in other cortical areas, or even in motor cortex itself. The important functional feature of these two populations in our model is that learning takes place solely in the synapses of projections between these populations. In principle, the same learning is applicable to multilayer networks. All of the modeled motor cortical neurons were used to determine the monkey arm movement in our model; however, only $n = 40$ of these (the "recorded" subset) were used for cursor control. The activities of this recorded subset are denoted in the following as $s_1(t), \ldots, s_n(t)$. The arm movement, based on the total population of modeled motor cortex neurons, was used to determine the PDs of modeled recorded neurons.

In monkeys, the transformation from motor cortical activity to arm movements involves a complicated system of several synaptic stages. In our model, we treated this transformation as a black box. Experimental

findings suggest that monkey arm movements can be predicted quite well by a linear model based on the activities of a small number of motor cortex neurons (Georgopoulos et al., 1989; Velliste et al., 2008). We therefore assumed that the direction of the monkey arm movement $\boldsymbol{y}^{\text{arm}}(t)$ at time t could be modeled in a linear way, using the activities $s_1(t), \ldots, s_{n_{\text{total}}}(t)$ of the total population of n_{total} cortical neurons and a fixed linear mapping:

$$\boldsymbol{y}^{\text{arm}}(t) \propto \sum_{i=1}^{n\ \text{total}} s_i(t)\boldsymbol{q}_i, \tag{1}$$

where $\boldsymbol{q}_i \in \mathbb{R}^3$ is the direction in which neuron i contributes to the movement. The vectors $\boldsymbol{q}_i$ were chosen randomly from a uniform distribution on the unit sphere (see below, Simulation details, Determination of input activities).

With the transformation from motor cortical neurons to monkey arm movements being defined, the input to the network for a given desired movement direction $\boldsymbol{y}^\star$ should be chosen such that motor cortical neurons produce a monkey arm movement close to $\boldsymbol{y}^\star$. We therefore calculated from the desired movement direction suitable input activities by a linear transformation (see Simulation details, Determination of input activities below). This transformation from desired directions to input neuron activities was defined initially and held fixed during each simulation because learning in response to perturbations took place in the single synaptic stage from neurons of the input population to neurons in the motor cortex population in our model; the coding of desired directions did not change in the input population.

Neuron model for motor cortex neurons. The total synaptic input $a_i(t)$ to neuron i at time t was modeled as a noisy weighted linear sum of its inputs:

$$a_i(t) = \sum_{j=1}^{m} w_{ij}x_j(t) + \xi_i(t),\ \xi_i(t) \text{ drawn from distribution } D(\nu), \tag{2}$$

where w_{ij} is the synaptic efficacy from input neuron j to neuron i. These weights were set randomly at the beginning of each simulation, drawn from a uniform distribution over $[-0.5, 0.5]$. $\xi_i(t)$ models some additional signal that is used for exploration (i.e., to explore possibly better network behaviors). In cortical neurons, this exploratory signal $\xi_i(t)$ could, for example, result from internal noise sources; it could be input from other brain areas; or it could be spontaneous activity of the neuron. At each time step, an independent sample from the zero mean distribution $D(\nu)$ was drawn as the exploratory signal $\xi_i(t)$. The parameter ν determines the variance of the distribution and hence the amount of noise in the neuron. We term the parameter ν the exploration level.

The activity $s_i(t)$ of neuron i at time t was modeled as a nonlinear function of the total synaptic input:

$$s_i(t) = \sigma(a_i(t)), \tag{3}$$

where σ: $\mathbb{R} \to \mathbb{R}$ is the threshold linear activation function that assures non-negative activities:

$$\sigma(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}. \tag{4}$$

Task model. We modeled the cursor control task as shown in Figure 1*A*. Eight possible cursor target positions were located at the corners of a unit cube in 3D space with its center at the origin of the coordinate system. We simulated the closed-loop situation where the cursor moves according to the network output during simulated sessions and weights are adapted online. Before the simulation of a cursor control session, we determined the preferred directions $\boldsymbol{p}_i$ of simulated recorded neurons ($i = 1, \ldots, n$) as described below, in Simulation details, Computation of preferred directions. In the simulated perturbation sessions, the decoding preferred directions $\boldsymbol{p}_i'$ of a randomly chosen subset of 25% or 50% of the modeled recorded neurons were rotated around one of the x-, y-, or z-axes (all PDs were rotated around a common axis in each experiment) as in the study by Jarosiewicz et al. (2008). The dPDs of the nonrotated neurons were left the same as their measured PDs. After the simulation of a perturbation session, preferred directions of recorded neurons were reestimated and compared to the original PDs.

Each simulated session consisted of a sequence of movements from the center to a target position at one of the corners of the imaginary cube, with online weight updates during the movements. To start a trial, the cursor position was initialized at the origin of the coordinate system and a target location was drawn randomly and uniformly from the corners of a cube with unit side length. The target location was held constant until the cursor hit the target, at which point the cursor was reset to the origin, a new target location was drawn, and another trial was simulated.

Each trial was simulated in the following way. At each time step t, we performed a series of six computations. (1) The desired direction of cursor movement $\boldsymbol{y}^\star(t)$ was computed as the difference between the target position $\boldsymbol{l}^\star(t)$ and the current cursor position $\boldsymbol{l}(t)$. By convention, the desired direction $\boldsymbol{y}^\star(t)$ had unit Euclidean norm. (2) From the desired movement direction $\boldsymbol{y}^\star(t)$, the activities $x_1(t), \ldots, x_m(t)$ of the neurons that provide input to the motor cortex neurons were computed via a fixed linear mapping. Details on how this mapping was determined are given below in Simulation details, Determination of the input activities. (3) These input activities $\boldsymbol{x}$ were then used to calculate the total synaptic activities $a_1(t), \ldots, a_{n_{\text{total}}}(t)$ and the resultant motor unit activities $s_1(t), \ldots, s_{n_{\text{total}}}(t)$ via Equations 2 and 3 above. (4) The activities $s_1(t), \ldots, s_n(t)$ of the subset of modeled recorded neurons were used to determine the cursor velocity via their population activity vector, described in Equation 9 below in Simulation details, Generating cursor movements from neural activity. (5) The synaptic weights w_{ij} defined in Equation 2 were updated according to a learning rule, defined by Equation 16 below in Results. (6) Finally, if the new cursor location was close to the target (i.e., if $\|\boldsymbol{l}(t) - \boldsymbol{l}^\star(t)\| < 0.05$), we deemed it a hit, and the trial ended. Otherwise, we simulated another time step and returned to computation step 1. In summary, every trial was simulated as follows:

(0) Initialize cursor to origin and pick target.
(1) Compute desired direction $\boldsymbol{y}^\star(t)$.
(2) Determine input activities $x_1(t), \ldots, x_m(t)$.
(3) Determine motor cortical activities $s_1(t), \ldots, s_{n_{\text{total}}}(t)$.
(4) Determine new cursor location $\boldsymbol{l}(t)$.
(5) Update synaptic weights w_{ij}.
(6) If target is not hit, set t to $t + \Delta t$ and return to step 1.

Simulation details

Determination of input activities. To draw the contributions of simulated motor-cortical neurons to monkey arm movement $\boldsymbol{q}_i = (q_{i1}, q_{i2}, q_{i3})^T$ for $i = 1, \ldots, n_{\text{total}}$ (see Eq. 1) randomly on the unit sphere, we adopted the following procedure. First, an angle φ_i was chosen randomly from the uniform distribution on $[0, 2\pi]$. Then, q_{i3} was chosen randomly from a uniform distribution on $[-1, 1]$. Finally, q_{i1} and q_{i2} were computed as follows:

$$q_{i1} = \sqrt{1 - q_{i3}^2}\cos(\varphi_i), \quad q_{i2} = \sqrt{1 - q_{i3}^2}\sin(\varphi_i). \tag{5}$$

The activities of the neurons in the input population $\boldsymbol{x}(t) = (x_1(t), \ldots, x_m(t))^T$ were determined such that the arm movement $\boldsymbol{y}^{\text{arm}}(t)$ approximated the target direction $\boldsymbol{y}^\star(t)$. We describe in this section how this was achieved. Let $\mathbf{Q}$ be the $3 \times n_{\text{total}}$ matrix where column i is given by $\boldsymbol{q}_i$ from Equation 1 for $i = 1, \ldots, n_{\text{total}}$. First we computed the vector $\tilde{\boldsymbol{s}} = \mathbf{Q}^\dagger \boldsymbol{y}^\star$, where $\mathbf{Q}^\dagger$ denotes the pseudoinverse of $\mathbf{Q}$. When the activities of motor cortex neurons are given by this vector $\tilde{\boldsymbol{s}}$, they produce an arm movement very close to $\boldsymbol{y}^\star$. Let $\mathbf{W}^{\text{total}}$ denote the matrix of weights w_{ij} before learning, i.e., the element of $\mathbf{W}^{\text{total}}$ in row i and column j is the weight from input neuron j to neuron i in the simulated motor cortex before learning. Since $\tilde{\boldsymbol{s}} \approx \mathbf{W}^{\text{total}}\boldsymbol{x}$, an input activity of $x(t) = (\mathbf{W}^{\text{total}})^\dagger \tilde{\boldsymbol{s}}$ approximately produces an arm movement in the desired direction $\boldsymbol{y}^\star$. The activities of the input neurons were thus directly given by the following:

$$\boldsymbol{x}(t) = c_{\text{rate}}(\boldsymbol{W}^{\text{total}})^\dagger \boldsymbol{Q}^\dagger \boldsymbol{y}^\star(t), \tag{6}$$

where we used the scaling factor c_{rate} to scale the input activity such that the activities of the neurons in the simulated motor cortex could directly

be interpreted as rates in hertz, i.e., such that their outputs were in the range between 0 and 120. c_{rate} was determined in the first time step of the simulation and then kept constant for all later time steps. By using the above equation, we neglected the nonlinearity of the non-negative linear activation function. This simple mapping can be calculated efficiently, and the error induced by this simplification is small. In general, we have tried different types of mappings and found that the choice of the input coding does not have a significant impact on the learning results.

Note that this mapping was defined initially and kept fixed during each simulation. Thus, when $\mathbf{W}^{\text{total}}$ was adapted by some learning rule, we still used the initial weights in the computation of the inputs, since we assumed that the coding of desired directions did not change in the input coding.

Computation of preferred directions. As described above, a subset of the motor cortex population was chosen to model the recorded neurons that were used for cursor control. For each modeled recorded neuron $i \in \{1, \ldots, n\}$, we determined the preferred direction $\boldsymbol{p}_i \in \mathbb{R}^3$, baseline activity β_i, and modulation depth α_i as follows. We defined eight unit norm target directions $\boldsymbol{y}^\star(1), \ldots, \boldsymbol{y}^\star(8)$ as the eight directions from the origin to the eight corners of an imaginary cube centered at the origin. The activations $s_i(1), \ldots, s_i(8)$ of neuron i for these target directions were computed without internal neural noise through Equations 2, 3, and 6 above. The fitting was then done by linear regression, i.e., minimizing the objective

$$\sum_{j=1}^{8} (s_i(j) - \boldsymbol{v}_i^T \boldsymbol{y}^\star(j) - \beta_i)^2 \tag{7}$$

with respect to the vector $\boldsymbol{v}_i = (v_{i1}, v_{i2}, v_{i3})^T$ and β_i. This is the fitting of a cosine tuning curve, since $\boldsymbol{v}_i^T \boldsymbol{y}^\star(j)$ is the cosine of the angle between $\boldsymbol{v}_i$ and $\boldsymbol{y}^\star(j)$ scaled by the L_2 norm of $\boldsymbol{v}_i$. We thus obtained the baseline firing rate β_i, the modulation depth $\alpha_i = \|\boldsymbol{v}_i\|$, and the preferred direction $\boldsymbol{p}_i = \boldsymbol{v}_i/\alpha_i$. Neuron i was thus approximately cosine tuned to the fitted preferred direction:

$$s_i(j) \approx \beta_i + \alpha_i \frac{\boldsymbol{y}^\star(j)^T \boldsymbol{p}_i}{\|\boldsymbol{p}_i\|}, \quad \text{for all } j. \tag{8}$$

After training, we reestimated the PDs and analyzed how they changed due to learning.

Generating cursor movements from neural activity. In the simulated perturbation session, the decoding preferred directions $\boldsymbol{p}_i'$ of a randomly chosen subset of 50% of the modeled recorded neurons were rotated around one of the *x*-, *y*-, or *z*-axes (all PDs were rotated around a common axis in each experiment). The dPDs of the nonrotated neurons were left the same as their measured PDs. The dPDs were then used to determine the movement velocity of the cursor as in the study by Jarosiewicz et al. (2008) by the population vector algorithm (Georgopoulos et al.,1988): The cursor velocity was computed as the vector sum of the dPDs weighted by the corresponding normalized activities:

$$y(t) = k_s \frac{d}{n} \sum_{i=1}^{n} \frac{s_i(t) - \beta_i}{\alpha_i} \boldsymbol{p}_i', \tag{9}$$

where d is the movement dimensionality (in our case 3), n is the number of recorded neurons, and the constant k_s converts the magnitude of the population vector to speed. To set this speed factor in accordance with the experimental setup, we had to take the following temporal and geometrical considerations into account. In the study by Jarosiewicz et al. (2008), the cursor position was updated every 30 Hz. Hence, a time step in our simulation corresponded to 1/30 s in biological time. The imaginary cube in the study by Jarosiewicz et al. (2008) had a side length of 11 cm, whereas we used a cube with unit side length in our simulations. We therefore used a speed factor of $k_s = 0.03$, which corresponds to a factor 100 mm/s used by Jarosiewicz et al. (2008).

Finally, this velocity signal was integrated to obtain the cursor position $\boldsymbol{l}(t)$:

$$\boldsymbol{l}(t + \Delta t) = \boldsymbol{l}(t) + \Delta t \boldsymbol{y}(t), \tag{10}$$

where $\Delta t = 1$ in our simulations.

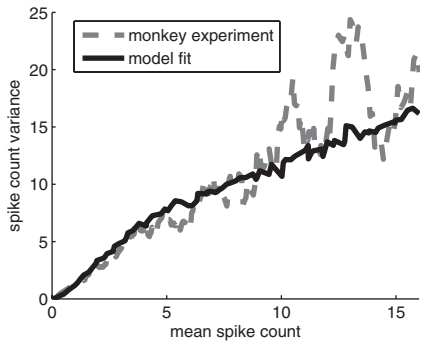

Figure 2. Experimentally observed variances of the spike count in a 200 ms window for motor cortex neurons and noise in the neuron model. For a given target direction, the variance of the spike count for a unit scales approximately linearly with the mean activity of the unit in that direction with a bend at ~5 spikes (20 Hz) in a monkey experiment (gray dashed line; data smoothed, see Materials and Methods). Similar behavior can be obtained by an appropriate noise distribution in our neuron model with non-negative linear activation function (black line).

Fitting of neuronal noise and learning rate to experimental data. To simulate the experiments as closely as possible, we fit the noise in our model to the experimental data. To obtain quantitative estimates for the variability of neuronal responses in a cursor control task, we analyzed the 12 cursor control experiments in the study by Jarosiewicz et al. (2008), in which 50% of the neurons were perturbed (990 presented targets in total). We calculated for each recorded neuron the mean and variance of its firing rate over all successful trajectories with a common target. The firing rate was computed in a 200 ms window halfway to the target. This resulted in a total of 3592 unit-target location pairs. To smooth the data, running averages were taken of the sorted mean activities r and variances v:

$$\bar{r}_i = \frac{1}{\tau + 1} \sum_{k=i}^{i+\tau} r_k \tag{11}$$

$$\bar{v}_i = \frac{1}{\tau + 1} \sum_{k=i}^{i+\tau} v_k. \tag{12}$$

We used a smoothing window of $\tau = 10$. Mean rates varied between 0 and 120 Hz with a roughly exponential distribution such that mean rates of >60 Hz were very rare. In Figure 2, the smoothed variances $\bar{v}$ were plotted as a function of the smoothed means $\bar{r}$. Since some recorded units can represent the activity of several neurons, this procedure may overestimate the amount of variability. This analysis was done on data from trained monkeys that have fairly stable movement trajectories. We thus obtained an estimate of neuronal firing rate variability for a given target direction.

The variance of the spike counts scaled approximately linearly with the mean spike count of a neuron for a given target location. This behavior can be obtained in our neuron model with noise that is a mixture of an activation-independent noise source and a noise source where the variance scales linearly with the noiseless activity of the neuron. In particular, the noise term $\xi_i(t)$ of neuron i was drawn from the uniform distribution in $[-\nu_i(x(t)), \nu_i(x(t))]$ with an exploration level ν_i in hertz that was given by the following:

$$\nu_i(x(t)) = \upsilon \left(1 + \sqrt{\kappa \sigma \left(\sum_{j=1}^{m} w_{ij} x_j(t) \right)} \right). \tag{13}$$

Recall that the input activities $x_j(t)$ were scaled in such a way that the output of the neuron at time t could be interpreted directly as its firing rate. This noisy neuron model fits the observed variability of activity in motor cortex neurons well for constants $\nu = 10$ Hz and $\kappa = 0.0784$ s (Fig. 2).

Having estimated the variability of neuronal response, the learning rate η (see Eq. 16 in Results) remained the last free parameter of the

model. No direct experimental evidence for the value of η exists. We have therefore chosen the value of η such that after 320 target presentations, the performance in the 25% perturbation task approximately matched the monkey performance. In the study by Jarosiewicz et al. (2008), the performance was measured as the deviation of the cursor trajectory from the ideal straight line measured when the trajectory was halfway to the target. In the 25% perturbation experiment, the monkey performance after 320 target presentations was ~3.2 mm. By constraining this parameter according to the experimental data, we ensured that the model did not depend on any free parameter. We note that with this learning rate, the performance of the model was superior to monkey performance in the 50% perturbation experiment (see below).

Determination of trajectory deviation. To compute the deviation of the trajectory from the ideal one, trajectories were first rotated into a common frame of reference as in the study by Jarosiewicz et al. (2008). In this reference frame, the target is positioned at $(1, 0, 0)^T$, movement along the x-axis represents movement toward the target, movement along the y-axis represents deviation in the direction of the applied perturbation, and movement along the z-axis represents deviation orthogonal to the applied perturbation. See Jarosiewicz et al. (2008) for the detailed transformation. The trajectory deviation in the perturbation direction halfway to the target is then given in this reference frame by the y-value of the rotated trajectory when its x-value crosses 0.5. We scaled the results to a cube of 11 cm side length to be able to compare the results directly to the results of Jarosiewicz et al. (2008).

Results

Adaptation with the EH learning rule

We model the learning effects observed by Jarosiewicz et al. (2008) through adaptation at a single synaptic stage, from a set of hypothesized input neurons to our motor cortical neurons. Adaptation of these synaptic efficacies w_{ij} will be necessary if the actual decoding PDs $\boldsymbol{p}_i'$ do not produce efficient cursor trajectories. To make this more clear, assume that suboptimal dPDs $\boldsymbol{p}_1', \ldots, \boldsymbol{p}_n'$ are used for decoding. Then for some input $\boldsymbol{x}(t)$, the movement of the cursor is not in the desired direction $\boldsymbol{y}^\star(t)$. The weights w_{ij} should therefore be adapted such that at every time step t, the direction of movement $\boldsymbol{y}(t)$ is close to the desired direction $\boldsymbol{y}^\star(t)$. We can quantify the angular match $R_{\text{ang}}(t)$ at time t by the cosine of the angle between movement direction $\boldsymbol{y}(t)$ and desired direction $\boldsymbol{y}^\star(t)$:

$$R_{\text{ang}}(t) = \frac{\boldsymbol{y}(t)^T \boldsymbol{y}^\star(t)}{\|\boldsymbol{y}(t)\|}. \quad (14)$$

This measure has a value of 1 if the cursor moves exactly in the desired direction, it is 0 if the cursor moves perpendicular to the desired direction, and it is -1 if the cursor movement is in the opposite direction. The angular match $R_{\text{ang}}(t)$ will be used as the reward signal for adaptation below. For desired directions $\boldsymbol{y}^\star(1), \ldots, \boldsymbol{y}^\star(T)$ and corresponding inputs $\boldsymbol{x}(1), \ldots, \boldsymbol{x}(T)$, the goal of learning is hence to find weights w_{ij} such that

$$R_{\text{batch}} = \frac{1}{T}\sum_{t=1}^{T} R_{\text{ang}}(t) \quad (15)$$

is maximized.

The plasticity model used in this article is based on the assumption that learning in motor cortex neurons has to rely on a single global scalar neuromodulatory signal that carries information about system performance. One way for a neuromodulatory signal to influence synaptic weight changes is by gating local plasticity. In the study by Loewenstein and Seung (2006), this idea was implemented by learning rules where the weight changes were proportional to the covariance between the reward signal R and some measure of neuronal activity N at the synapse, where N could correspond to the presynaptic activity, the postsynaptic activity, or the product of both. The authors showed that such learning rules can explain a phenomenon called Herrnstein's matching law. Interestingly, for the analysis of Loewenstein and Seung (2006), the specific implementation of this correlation-based adaptation mechanism is not important. From this general class, we investigate in this article the following learning rule:

$$\text{EH-rule: } \Delta w_{ij}(t) = \eta x_j(t)[a_i(t) - \bar{a}_i(t)][R(t) - \bar{R}(t)], \quad (16)$$

where $\bar{z}(t)$ denotes the low-pass filtered version of some variable z with an exponential kernel; we used $\bar{z}(t) = 0.8\bar{z}(t-1) + 0.2z(t)$. We call this rule the exploratory Hebb rule (EH rule). The important feature of this learning rule is that apart from variables that are locally available for each neuron ($x_j(t)$, $a_i(t)$, $\bar{a}_i(t)$), only a single scalar signal, the reward signal $R(t)$, is needed to evaluate performance (we also explored a rule where the activation a_i is replaced by the output s_i and obtained very similar results). This reward signal is provided by some neural circuit that evaluates performance of the system. In our simulations, we simply use the angular match $R_{\text{ang}}(t)$, corresponding to the deviation of the instantaneous trajectory from its ideal path to the target, as this reward signal. The rule measures correlations between deviations of the reward signal $R(t)$ from its mean and deviations of the activation $a_i(t)$ from the mean activation and adjusts weights such that rewards above mean are reinforced. The EH rule approximates gradient ascent on the reward signal by exploring alternatives to the actual behavior with the help of some exploratory signal $\xi(t)$. The exploratory signal could, for example, be interpreted as spontaneous activity, internal noise, or input from some other brain area. The deviation of the activation from the recent mean $a_i(t) - \bar{a}_i(t)$ is an estimate of the exploratory term $\xi_i(t)$ at time t if the mean $\bar{a}_i(t)$ is based on neuron activations $\sum_j w_{ij}x_j(t')$, which are similar to the activation $\sum_j w_{ij}x_j(t)$ at time t. Here we make use of (1) the fact that weights are changing very slowly and (2) the continuity of the task (inputs x at successive time points are similar). If conditions 1 and 2 hold, the EH rule can be seen as an approximation of the following:

$$\Delta w_{ij}(t) = \eta x_j(t)\xi_i(t)[R(t) - \bar{R}(t)]. \quad (17)$$

This rule is a typical node-perturbation learning rule (Mazzoni et al., 1991; Williams, 1992; Baxter and Bartlett, 2001; Fiete and Seung, 2006) (see also the Discussion) that can be shown to approximate gradient ascent (see, e.g., Fiete and Seung, 2006). A simple derivation that shows the link between the EH rule and gradient ascent is given in the Appendix.

The EH learning rule is different from other node-perturbation rules in one important aspect. In standard node-perturbation learning rules, the noise needs to be accessible to the learning mechanism separately from the output signal. For example, in the studies by Mazzoni et al. (1991) and Williams (1992), binary neurons were used and the noise appears in the learning rule in the form of the probability of the neuron to output 1. In the study by Fiete and Seung (2006), the noise term is directly incorporated in the learning rule. The EH rule instead does not directly need the noise signal, but a temporally filtered version of the activation of the neuron, which is an estimate of the noise signal. Obviously, this estimate is only sufficiently accurate if the structure of the task is appropriate, i.e., if the input to the neuron is temporally stable on small timescales. We note that the filtering of postsynaptic activity makes the Hebbian part of the EH rule

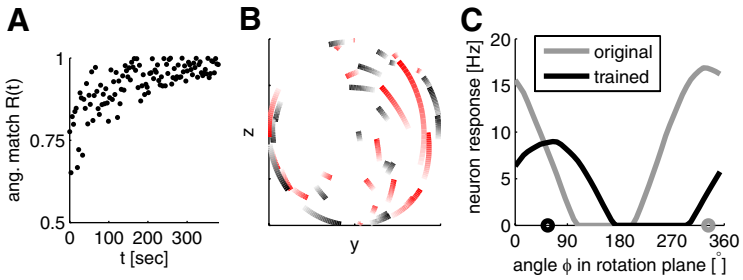

Figure 3. One example simulation of the 50% perturbation experiment with the EH rule and data-derived network parameters. ***A***, Angular match R_{ang} as a function of learning time. Every 100th time point is plotted. ***B***, PD shifts projected onto the rotation plane (the rotation axis points toward the reader) for rotated (red) and nonrotated (black) neurons from their initial values (light color) to their values after training (intense color, these PDs are connected by the shortest path on the unit sphere; axes in arbitrary units). The PDs of rotated neurons are consistently rotated counter-clockwise to compensate for the perturbation. ***C***, Tuning of an example rotated neuron to target directions of angle Φ in the rotation plane (y–z-plane) before (gray) and after (black) training. The target direction for a given Φ was defined as $\mathbf{y}^*(\Phi) = (1/\sqrt{2})(1, \cos(\Phi), \sin(\Phi))^T$. Circles on the x-axis indicate projected preferred directions of the neuron before (gray) and after (black) training.

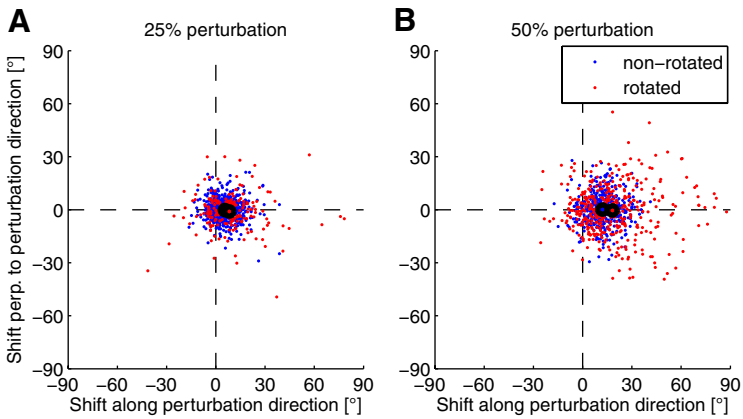

Figure 4. PD shifts in simulated perturbation sessions are in good agreement with experimental data [compare to Jarosiewicz et al. (2008), their Fig. 3 *A*, *B*]. Shift in the PDs measured after simulated perturbation sessions relative to initial PDs for all units in 20 simulated experiments where 25% (***A***) or 50% (***B***) of the units were rotated. Dots represent individual data points and black circled dots represent the means of the rotated (red) and nonrotated (blue) units.

reminiscent of a linearized BCM rule (Bienenstock et al., 1982). The postsynaptic activity is compared with a threshold to decide whether the synapse is potentiated or depressed.

Comparison with experimentally observed learning effects

We simulated the two types of perturbation experiments reported by Jarosiewicz et al. (2008) in our model network with 40 recorded neurons. In the first set of simulations, we chose 25% of the recorded neurons to be rotated neurons, and in the second set of simulations, we chose 50% of the recorded neurons to be rotated. In each simulation, 320 targets were presented to the model, which is similar to the number of target presentations in the study by Jarosiewicz et al. (2008). The performance improvement and PD shifts for one example run are shown in Figure 3. To simulate the experiments as closely as possible, we fit the noise and the learning rate in our model to the experimental data (see Materials and Methods). All neurons showed a tendency to compensate the perturbation by a shift of their PDs in the direction of the perturbation rotation. This tendency is stronger for rotated neurons. The training-induced shifts in PDs of the recorded neurons were compiled from 20 independent simulated experiments, and analyzed separately for rotated and nonrotated neurons. The results are in good agreement with the experimental data (Fig. 4). In the simulated 25% perturbation experiment, the mean shift of the PD for rotated neurons was 8.2 ± 4.8°, whereas for nonrotated neurons, it was 5.5 ± 1.6°. This is a relatively small effect, similar to the effect observed by Jarosiewicz et al. (2008), where the PD shifts were 9.86° for rotated units and 5.25° for nonrotated units. A stronger effect can be found in the 50% perturbation experiment (see below). We also compared the deviation of the trajectory from the ideal straight line in rotation direction halfway to the target (see Materials and Methods) from early trials to the deviation of late trials. In early trials, the trajectory deviation was 9.2 ± 8.8 mm, which was reduced by learning to 2.4 ± 4.9 mm. In the simulated 50% perturbation experiment, the mean shift of the PD for rotated neurons was 18.1 ± 4.2°, whereas for nonrotated neurons, it was 12.1 ± 2.6°. Again, the PD shifts are very similar to those in the monkey experiments: 21.7° for rotated units and 16.11° for nonrotated units. The trajectory deviation was 23.1 ± 7.5 mm in early trials, and 4.8 ± 5.1 mm in late trials. Here, the early deviation was stronger than in the monkey experiment, while the late deviation was smaller.

The EH rule falls into the general class of learning rules where the weight change is proportional to the covariance of the reward signal and some measure of neuronal activity (Loewenstein and Seung, 2006). Interestingly, the specific implementation of this idea influences the learning effects observed in our model. We performed the same experiment with slightly different correlation-based rules:

$$\Delta w_{ij}(t) = \eta x_j(t) a_i(t)[R(t) - \bar{R}(t)] \tag{18}$$

and

$$\Delta w_{ij}(t) = \eta x_j(t)[a_i(t) - \bar{a}_i(t)]R(t), \tag{19}$$

where the filtered postsynaptic activation or the filtered reward was not taken into account. Compare these to the EH rule (Eq. 16). These rules also converge with performance similar to the EH rule. However, no credit assignment effect can be observed with these rules. In the simulated 50% perturbation experiment, the mean shift of the PD of rotated neurons (nonrotated neurons) was 25.5 ± 4.0° (26.8 ± 2.8°) for the rule given by Equation 18 and 12.8 ± 3.6° (12.0 ± 2.4°) for the rule given by Equation 19 (Fig. 5). Only when deviations of the reward from its local mean and deviations of the activation from its local mean are both taken into account do we observe differential changes in the two populations of cells.

In the monkey experiment, training in the perturbation session also resulted in a decrease of the modulation depth of rotated neurons, which led to a relative decrease of the contribution of these neurons to the cursor movement. A qualitatively similar result could be observed in our simulations. In the 25% perturbation

simulation, modulation depths of rotated neurons changed on average by −2.7 ± 4.3 Hz, whereas modulation depths of nonrotated neurons changed on average by 2.2 ± 3.9 Hz (average over 20 independent simulations; a negative change indicates a decreased modulation depth in the perturbation session relative to the control session). In the 50% perturbation simulation, the changes in modulation depths were on average −3.6 ± 5.5 Hz for rotated neurons and 5.4 ± 6.0 Hz for nonrotated neurons (when comparing these results to experimental results, one has to take into account that modulation depths in monkey experiments were around 10 Hz, whereas in the simulations, they were ~25 Hz). Thus, the relative contribution of rotated neurons on cursor movement decreased during the perturbation session.

It was reported by Jarosiewicz et al. (2008) that after the perturbation session, PDs returned to their original values in a subsequent washout session where the original PDs were used as decoding PDs. We simulated such washout sessions after our simulated perturbation sessions in the model and found a similar effect (Fig. 6*A*,*B*). However, the retuning in our simulation is slower than observed in the monkey experiments. In the experiments, it took about 160 target presentations until mean PD shifts relative to PDs in the control session were around zero. This fast unlearning is consistent with the observation that adaptation and deadaptation in motor cortex can occur at substantially different rates, likely reflecting two separate processes (Davidson and Wolpert, 2004). We did not model such separate processes; thus, the timescales for adaptation and deadaptation are the same in the simulations. In a simulated washout session with a larger learning rate, we found faster convergence of PDs to original values (Fig. 6*C*,*D*).

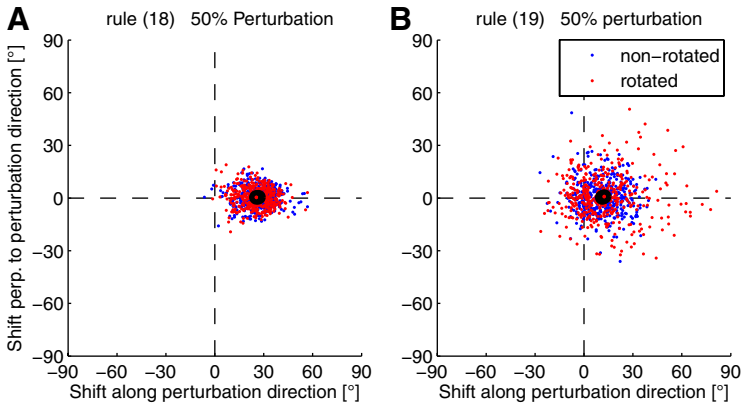

Figure 5. PD shifts in simulated 50% perturbation sessions with the learning rules in Equations 18 (***A***) and 19 (***B***). Dots represent individual data points and black circled dots represent the means of the rotated (red) and nonrotated (blue) units. No credit assignment effect can be observed for these rules.

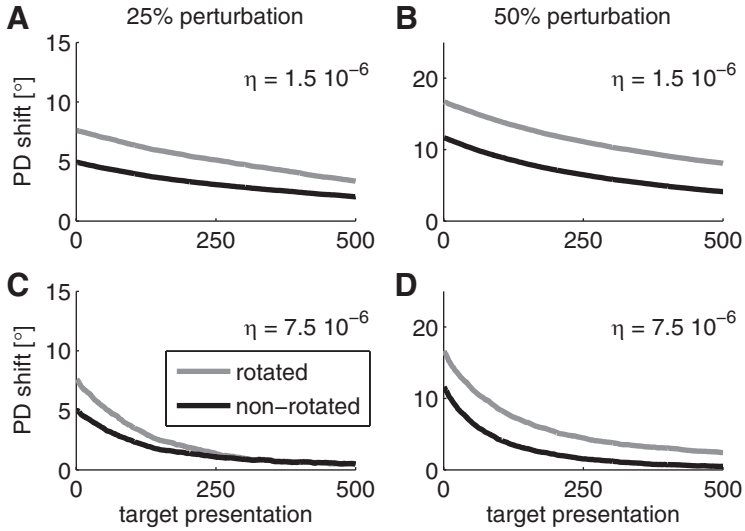

Figure 6. PDs shifts in simulated washout sessions. ***A–D***, Shift in the PDs (mean over 20 trials) for rotated neurons (gray) and nonrotated neurons (black) relative to PDs of the control session as a function of the number of targets presented for 25% perturbation (***A***, ***C***) and 50% perturbation (***B***, ***D***). ***A***, ***B***, Simulations with the same learning rate as in the simulated perturbation session. ***C***, ***D***, Simulations with a five times larger learning rate.

The performance of the system before and after learning is shown in Figure 7. The neurons in the network after training are subject to the same amount of noise as the neurons in the network before training, but the angular match after training shows much less fluctuation than before training. We therefore conjectured that the network automatically suppresses jitter in the trajectory in the presence of high exploration levels v. We quantified this conjecture by computing the mean angle between the cursor velocity vector with and without noise for 50 randomly drawn noise samples. In the mean over the 20 simulations and 50 randomly drawn target directions, this angle was 10 ± 2.7° (mean ± SD) before learning and 9.6 ± 2.5° after learning. Although only a slight reduction, it was highly significant when the mean angles before and after learning were compared for identical target directions and noise realizations ($p < 0.0002$, paired t test). This is not an effect of increased network weights, because weights increased only slightly and the same test where weights were normalized to their initial L_2 norm after training produced the same significance value.

Psychophysical studies in humans (Imamizu et al., 1995) and monkeys (Paz and Vaadia, 2004) showed that the learning of a new sensorimotor mapping generalizes poorly to untrained directions with better generalization for movements in directions close to the trained one. It was argued by Imamizu et al. (1995) that this is evidence for a neural network-like model of sensorimotor mappings. The model studied in this article exhibits similar generalization behavior. When training is constrained to a single target location, performance is optimized in this direction, while the performance clearly decreased as target direction increased from the trained angle (Fig. 8).

Tuning changes depend on the exploration level

When we compare the results obtained by our simulations to those of monkey experiments [compare Fig. 4 to Jarosiewicz et al. (2008), their Fig. 3], it is interesting that quantitatively similar effects were obtained with noise levels that were measured in the experiments. We therefore explored whether the fitting of pa-

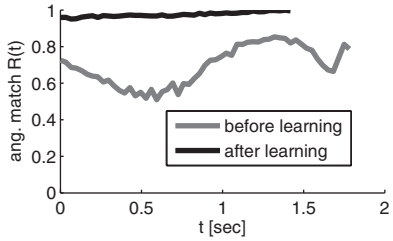

Figure 7. Comparison of network performance before and after learning for 50% perturbation. Angular match $R_{ang}(t)$ of the cursor movements in one reaching trial before (gray) and after (black) learning as a function of the time since the target was first made visible. The black curve ends prematurely because the target is reached faster. Note the reduced temporal jitter of the performance after learning, indicating reduced sensitivity to the noise signal.

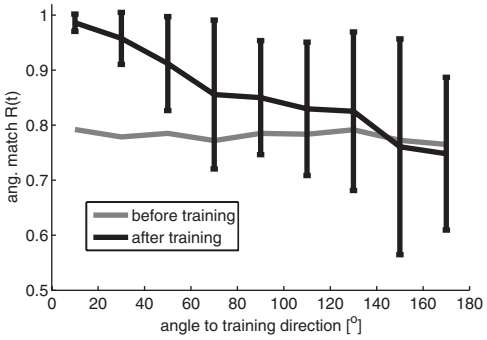

Figure 8. Generalization of network performance for a 50% perturbation experiment with cursor movements to a single target location during training. Twenty independent simulations with randomly drawn target positions (from the corners of the unit cube) and rotation axes (either the *x*-, *y*-, or *z*-axis) were performed. In each simulation the network model was first tested on 100 random target directions, then trained for 320 trials, and then tested again on 100 random target directions. Angular matches of the test trials before (gray) and after (black) training are plotted against the angle between the target direction vector in the test and the vector from the origin to the training target location. Shown is the mean and SD of the angular match R_{ang} over movements with an angle to the training direction in [0, 20]°, (20, 40]°, (40, 60]°, etc. For clarity, the SD for movements before learning is not shown. It was quite constant over all angles being 0.15 in the mean.

rameters to values extracted from experimental data was important by exploring the effect of different exploration levels and learning rates on performance and PD shifts.

The amount of noise was controlled by modifying the exploration level ν (see Eq. 13). For some extreme parameter settings, the EH rule can lead to large weights. We therefore implemented a homeostatic mechanism by normalizing the weight vector of each neuron after each update, i.e., the weight after the tth update step is given by the following:

$$w_{ij}(t+1) = \sqrt{\sum_k w_{ik}(t)^2}\,\frac{w_{ij}(t) + \Delta w_{ij}(t)}{\sqrt{\sum_k (w_{ik}(t) + \Delta w_{ik}(t))^2}}. \quad (20)$$

Employing the EH learning rule, the network converged to weight settings with good performance for most parameter settings, except for large learning rates and very large noise levels. Note that good performance is achieved even for large exploration levels of $\nu \approx 60$ Hz (Fig. 9*A*). The good performance of the system shows that already a very small network can use large amounts of noise for learning, while this noise does not interfere with performance.

We investigated the influence of learning on the PDs of circuit neurons. The amount of exploration and the learning rate η both turned out be important parameters. The tuning changes reported in neurons of monkeys subsumed under the term "credit assignment effect" were qualitatively met by our model networks for most parameter settings (Fig. 9), except for very large learning rates (when learning does not work) and very small learning rates (compare panels *B* and *C*). Quantitatively, the amount of PD shift especially for rotated neurons strongly depends on the exploration level, with shifts close to 50° for large exploration levels.

To summarize, for small levels of exploration, PDs change only slightly and the difference in PD change between rotated and nonrotated neurons is small, while for large noise levels, PD change differences can be quite drastic. Also the learning rate η influences the amount of PD shifts. This shows that the learning rule guarantees good performance and a qualitative match to experimentally observed PD shifts for a wide range of parameters. However, for the quantitative fit found in our simulations, the parameters extracted from experimental data turned out to be important.

Discussion

By implementing a learning rule that uses neuronal noise as an exploratory signal for parameter adaptation, we have successfully simulated experimental results showing selective learning within a population of cortical neurons (Jarosiewicz et al., 2008). This learning rule implements synaptic weight updates based on the instantaneous correlation between the deviation of a global error from its recent mean and the deviation of the neural activity from its recent mean; all of these parameters would be readily accessible in a biological system. Strikingly, it turns out that the use of noise levels similar to those that had been measured in experiments was essential to reproduce the learning effects found in the monkey experiments.

Jarosiewicz et al. (2008) discussed three possible strategies that could be used to compensate for the errors caused by the perturbations: reaiming, reweighting, and remapping. With reaiming, the monkey would compensate for perturbations by aiming for a virtual target located in the direction that offsets the visuomotor rotation. The authors identified a global change in the measured PDs of all neurons, indicating that monkeys used a reaiming strategy. Reweighting would reduce the errors by selectively suppressing the use of rotated units, i.e., a reduction of their modulation depths relative to the modulation depths of nonrotated units. The same reduction was found in the firing rates of the rotated neurons in the data. A remapping strategy would selectively change the directional tunings of rotated units. As discussed above, rotated neurons shifted their PDs more than the nonrotated population. Hence, the authors found elements of all three strategies in their data. We identified in our model all three elements of neuronal adaptation, i.e., a global change in activity of neurons (all neurons changed their tuning properties; reaiming), a reduction of modulation depths for rotated neurons (reweighting), and a selective change of the directional tunings of rotated units (remapping). This modeling study therefore suggests that all three elements could be explained by a single learning mechanism. Furthermore, the credit assignment phenomenon observed by Jarosiewicz et al. (2008) (reweighting and remapping) is an emergent feature of our learning rule.

Although the match of simulation results to experimental results is quite good, systematic differences exist. The change in simulated modulation depth was approximately twice that found in the experiments. It also turned out that the model produced smaller trajectory deviations after learning in the 50% deviation experiment. Such quantitative discrepancies could be attributed to the simplicity of the model. However, another

factor that could systematically contribute to all of the stronger effects could be the accurate reward signal modeled at the synapse. We did not incorporate noisy reward signals in our model, however, because this would introduce a free parameter with no available evidence for its value. Instead, the parameters of the presented model were strongly constrained: the noise level was estimated from the data, and the learning rate was chosen such that the average trajectory error in the 25% perturbation experiment was comparable to that in experiments after a given number of trials.

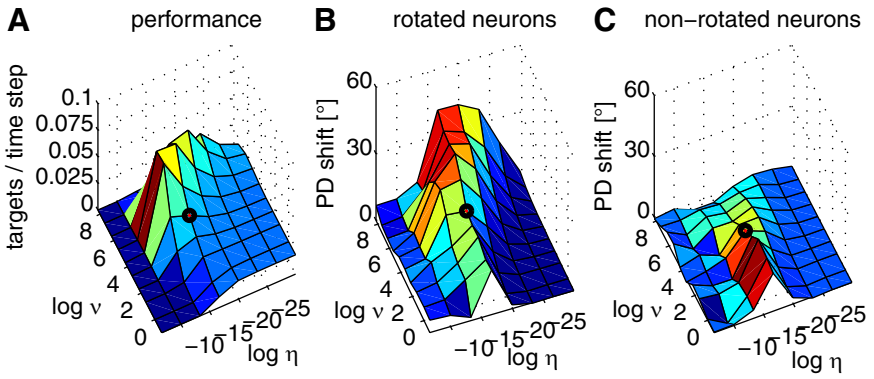

Figure 9. Behavior of the EH rule in simulated perturbation sessions (50% perturbed neurons) for different parameter settings. All plotted values are means over 10 independent simulations. Logarithms are to the basis of 2. The black circle indicates the parameter setting used in Results. ***A***, Dependence of network performance measured as the mean number of targets reached per time step on learning rate η and exploration level υ. Performance deteriorates for high learning rate and exploration levels. ***B***, Mean PD shifts in rotation direction for rotated neurons. ***C***, Mean PD shifts in rotation direction for nonrotated neurons. In comparison to rotated neurons, PD shifts of nonrotated neurons are small, especially for larger exploration levels.

Comparison of the EH rule with other learning models

Several reward-modulated Hebbian learning rules have been studied, both in the context of rate-based (Barto et al., 1983; Mazzoni et al., 1991; Williams, 1992; Baxter and Bartlett, 1999; Loewenstein and Seung, 2006) and spiking-based (Xie and Seung, 2004; Fiete and Seung, 2006; Pfister et al., 2006; Baras and Meir, 2007; Farries and Fairhall, 2007; Florian, 2007; Izhikevich, 2007; Legenstein et al., 2008) models. They turn out to be viable learning mechanisms in many contexts and constitute a biologically plausible alternative to the backpropagation-based mechanisms preferentially used in artificial neural networks. Such three-factor learning rules are well studied in corticostriatal synapses where the three factors are presynaptic and postsynaptic activity and dopamine (see, e.g., Reynolds and Wickens, 2002). The current conclusion drawn from the experimental literature is that presynaptic and postsynaptic activity is needed for plasticity induction. Depression is induced at low dopamine levels, and potentiation is induced at high dopamine levels. The EH rule is in principle consistent with these observations, although it introduces an additional dependency on the recent postsynaptic rate and reward, which has not been rigorously tested experimentally.

Reinforcement learning takes place when an agent learns to choose optimal actions based on some measure of performance. To improve performance, the agent has to explore different behaviors. In neuronal reinforcement learning systems, exploration is often implemented by some noise source that perturbs the operation to explore whether parameter settings should be adjusted to increase performance. In songbirds, syllable variability results in part from variations in the motor command, i.e., the variability of neuronal activity (Sober et al., 2008). It has been hypothesized that this motor variability reflects meaningful motor exploration that can support continuous learning (Tumer and Brainard, 2007). Two general classes of perturbation algorithms can be found in the literature. Either the tunable parameters of the system (weights) are perturbed (Jabri and Flower, 1992; Cauwenberghs, 1993; Seung, 2003) or the output of nodes in the network are perturbed (Mazzoni et al., 1991; Williams, 1992; Baxter and Bartlett, 2001; Fiete and Seung, 2006). The latter have the advantage that the perturbation search space is smaller and that the biological interpretation of the perturbation as an internal neural noise is more natural. Another interesting idea is the postulation of an "experimenter," that is, a system that injects noisy current into trained neurons. Some evidence for an experimenter exists in the song-learning system of zebra finches (Fiete et al., 2007). For the EH learning rule, the origin of the exploratory signal is not critical, as long as the trained neurons are noisy. The EH learning rule is in its structure similar to the rule proposed by Fiete and Seung (2006). However, while it had to be assumed by Fiete and Seung (2006) that the experimenter signal [$\xi_i(t)$ in our notation] is explicitly available and distinguishable from the membrane potential at the synapse, the EH rule does not rely on this separation. Instead it exploits the temporal continuity of the task, estimating $\xi_i(t)$ from activation history.

Often perturbation algorithms use eligibility traces to link perturbations at time t to rewards delivered at some later point in time $t' > t$. In fact, movement evaluation may be slow, and the release/effect of neuromodulators may add to the delay in response imparted to neurons in the trained area. For simplicity, we did not use eligibility traces and assumed that evaluation by the critic can be done quite fast.

The EH rule falls into the general class of learning rules where the weight change is proportional to the covariance of the reward signal and some measure of neuronal activity (Loewenstein and Seung, 2006). Interestingly, the specific implementation of this idea influences the learning effects observed in our model. In particular, we found that the implementations given by the rules in Equations 18 and 19 do not exhibit the reported credit assignment effect.

The results of this modeling paper also support the hypotheses introduced by Rokni et al. (2007). The authors presented data suggesting that neural representations change randomly (background changes) even without obvious learning, while systematic task-correlated representational changes occur within a learning task. They proposed a theory based on three assumptions: (1) representations in motor cortex are redundant, (2) sensory feedback is translated to synaptic changes in a task, and (3) the plasticity mechanism is noisy. These assumptions are also met in our model of motor cortex learning. The authors also provided a simple neural network model where the stochasticity of plasticity was modeled directly by random weight changes. In our model, such stochasticity arises from the firing rate noise of the model neurons, and it is necessary for task-dependent learning. This neuronal behavior together with the EH rule also leads to background synaptic changes in the absence of obvious learning (i.e., when performance is perfect or near-perfect).

Conclusion

Reward-modulated learning rules capture many of the empirical characteristics of local synaptic changes thought to generate goal-directed behavior based on global performance signals. The EH rule is one particularly simple instance of such a rule that emphasizes an exploration signal, a signal that would show up as "noise" in neuronal recordings. We showed that large exploration levels are beneficial for the learning mechanism without interfering with baseline performance, because of readout pooling effects. The study therefore provides a hypothesis about the role of "noise" or ongoing activity in cortical circuits as a source for exploration used by local learning rules. The data from Jarosiewicz et al. (2008) suggest that the level of noise in motor cortex is quite high. Under such realistic noise conditions, our model produces effects strikingly similar to those found in the monkey experiments, which suggests that this noise is essential for cortical plasticity. Obviously, these learning mechanisms are important for neural prosthetics, since they allow closed-loop corrections for poor extractions of movement intention. In addition, these learning mechanisms may be a general feature used for the acquisition of goal-directed behavior.

Appendix: Theoretical Link between the EH Rule and Gradient Ascent

In the following, we give a simple derivation that shows that the EH rule performs gradient ascent on the reward signal $R(t)$. The weights should change in the direction of the gradient of the reward signal, which is given by the chain rule as follows:

$$\frac{\partial R(t)}{\partial w_{ij}} = \frac{\partial R(t)}{\partial a_i(t)} \frac{\partial a_i(t)}{\partial w_{ij}} = \frac{\partial R(t)}{\partial a_i(t)} x_j(t), \tag{A1}$$

where $a_j(t)$ is the total synaptic input to neuron j at time t (see Eq. 2 in the main text). We assume that the noise ξ is independently drawn at each time and for every neuron with zero mean and variance μ^2, hence we have $\langle \xi_i(t) \rangle = 0$, and $\langle \xi_i(t) \xi_j(t') \rangle = \mu^2 \delta_{ij} \delta(t - t')$, where δ_{ij} denotes the Kronecker delta, $\delta(t - t')$ denotes the Dirac delta, and $\langle \cdot \rangle$ denotes an average over trials. Let $R_0(t)$ denote the reward at time t that would be delivered for a network response without noise. The deviation of the reward $R(t)$ from $R_0(t)$ can be approximated to be linear in the noise for small noise:

$$R(t) - R_0(t) \approx \sum_k \frac{\partial R(t)}{\partial a_k(t)} \xi_k(t). \tag{A2}$$

Multiplying this equation with $\xi_i(t)$ and averaging over different realizations of the noise, we obtain the correlation between the reward at time t and the noise signal at neuron i:

$$\langle [R(t) - R_0(t)] \xi_i(t) \rangle \approx \sum_k \frac{\partial R(t)}{\partial a_k(t)} \langle \xi_k(t) \xi_i(t) \rangle = \mu^2 \frac{\partial R(t)}{\partial a_i(t)}. \tag{A3}$$

The last equality follows from the assumption that the noise signal is temporally and spatially uncorrelated. Hence, the derivative of the reward signal with respect to the activation of neuron i is as follows:

$$\frac{\partial R(t)}{\partial a_i(t)} \approx \frac{1}{\mu^2} \langle [R(t) - R_0(t)] \xi_i(t) \rangle. \tag{A4}$$

Since $\langle \xi_i(t) \rangle = 0$, we find the following:

$$a_i(t) - \langle a_i(t) \rangle = \sum_{j=1}^{m} w_{ij} x_j(t) + \xi_i(t) - \sum_{j=1}^{m} w_{ij} x_j(t) - \langle \xi_i(t) \rangle = \xi_i(t), \tag{A5}$$

and we can write Equation A4 as follows:

$$\frac{\partial R(t)}{\partial a_i(t)} \approx \frac{1}{\mu^2} \langle [R(t) - R_0(t)][a_i(t) - \langle a_i(t) \rangle] \rangle. \tag{A6}$$

We note that the following is true:

$$\langle [R(t) - R_0(t)][a_i(t) - \langle a_i(t) \rangle] \rangle = \langle R(t) a_i(t) \rangle - \langle R(t) \rangle \langle a_i(t) \rangle = \langle [R(t) - \langle R(t) \rangle][a_i(t) - \langle a_i(t) \rangle] \rangle. \tag{A7}$$

Using this result in Equation A1, we obtain the following:

$$\frac{\partial R(t)}{\partial w_{ij}} \approx \frac{1}{\mu^2} \langle [R(t) - \langle R(t) \rangle][a_i(t) - \langle a_i(t) \rangle] \rangle x_j(t). \tag{A8}$$

In our implementation, the EH learning rule estimates $\langle a_i(t) \rangle$ (that is, the neuron activation averaged over different realizations of the noise for a given input) and $\langle R_i(t) \rangle$ by temporal averages $\bar{a}_i(t)$ and $\bar{R}_i(t)$. With these temporal averages, the EH rule approximates gradient ascent on $R(t)$ if the noise signal can be estimated from $a_i(t) - \bar{a}_i(t)$ (i.e., if the input changes slowly compared to the noise signal). We further note that for a small learning rate and if the input changes slowly compared to the noise signal, the weight vector is self-averaging, and we can neglect the outer average in Equation A8.

References

Baras D, Meir R (2007) Reinforcement learning, spike-time-dependent plasticity, and the BCM rule. Neural Comput 19:2245–2279.

Barto AG, Sutton RS, Anderson CW (1983) Neuronlike adaptive elements that can solve difficult learning and control problems. IEEE Trans Syst Man Cybern 13:835–846.

Baxter J, Bartlett PL (1999) Direct gradient-based reinforcement learning: I. Gradient estimation algorithms. Canberra, Australia: Research School of Information Sciences and Engineering, Australian National University.

Baxter J, Bartlett PL (2001) Infinite-horizon policy-gradient estimation. J Artif Intell Res 15:319–350.

Bienenstock EL, Cooper LN, Munro PW (1982) Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. J Neurosci 2:32–48.

Carmena JM, Lebedev MA, Crist RE, O'Doherty JE, Santucci DM, Dimitrov DF, Patil PG, Henriquez CS, Nicolelis MA (2003) Learning to control a brain-machine interface for reaching and grasping by primates. PLoS Biol 1:E42.

Cauwenberghs G (1993) A fast stochastic error-descent algorithm for supervised learning and optimization. In: Advances in neural information processing systems (Hanson SJ, Cowan JD, Giles CL, eds), pp 244–251. San Mateo, CA: Morgan Kaufmann.

Davidson PR, Wolpert DM (2004) Scaling down motor memories: de-adaptation after motor learning. Neurosci Lett 370:102–107.

Farries MA, Fairhall AL (2007) Reinforcement learning with modulated spike timing-dependent synaptic plasticity. J Neurophysiol 98:3648–3665.

Fiete IR, Seung HS (2006) Gradient learning in spiking neural networks by dynamic perturbation of conductances. Phys Rev Lett 97:048104.

Fiete IR, Fee MS, Seung HS (2007) Model of birdsong learning based on gradient estimation by dynamic perturbation of neural conductances. J Neurophysiol 98:2038–2057.

Florian RV (2007) Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. Neural Comput 19:1468–1502.

Ganguly K, Carmena JM (2009) Emergence of a stable cortical map for neuroprosthetic control. PLoS Biol 7:e1000153.

Georgopoulos AP, Schwartz AB, Kettner RE (1986) Neuronal population coding of movement direction. Science 233:1416–1419.

Georgopoulos AP, Kettner RE, Schwartz AB (1988) Primate motor cortex and free arm movements to visual targets in three-dimensional space. II. Coding of the direction of movement by a neuronal population. J Neurosci 8:2928–2937.

Georgopoulos AP, Crutcher MD, Schwartz AB (1989) Cognitive spatial-motor processes. 3. Motor cortical prediction of movement direction during an instructed delay period. Exp Brain Res 75:183–194.

Imamizu H, Uno Y, Kawato M (1995) Internal representations of the motor apparatus: implications from generalization in visuomotor learning. J Exp Psychol Hum Percept Perform 21:1174–1198.

Izhikevich EM (2007) Solving the distal reward problem through linkage of STDP and dopamine signaling. Cereb Cortex 17:2443–2452.

Jabri M, Flower B (1992) Weight perturbation: an optimal architecture and learning technique for analog VLSI feedforward and recurrent multilayer networks. IEEE Trans Neural Netw 3:154–157.

Jarosiewicz B, Chase SM, Fraser GW, Velliste M, Kass RE, Schwartz AB (2008) Functional network reorganization during learning in a brain-computer interface paradigm. Proc Natl Acad Sci U S A 105:19486–19491.

Legenstein R, Pecevski D, Maass W (2008) A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. PLoS Comput Biol 4:e1000180.

Loewenstein Y, Seung HS (2006) Operant matching is a generic outcome of synaptic plasticity based on the covariance between reward and neural activity. Proc Natl Acad Sci U S A 103:15224–15229.

Mazzoni P, Andersen RA, Jordan MI (1991) A more biologically plausible learning rule for neural networks. Proc Natl Acad Sci U S A 88:4433–4437.

Musallam S, Corneil BD, Greger B, Scherberger H, Andersen RA (2004) Cognitive control signals for neural prosthetics. Science 305:258–262.

Paz R, Vaadia E (2004) Specificity of sensorimotor learning and the neural code: neuronal representations in the primary motor cortex. J Physiol Paris 98:331–348.

Pfister J-P, Toyoizumi T, Barber D, Gerstner W (2006) Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. Neural Comput 18:1318–1348.

Reynolds JN, Wickens JR (2002) Dopamine-dependent plasticity of corticostriatal synapses. Neural Netw 15:507–521.

Rokni U, Richardson AG, Bizzi E, Seung HS (2007) Motor learning with unstable neural representations. Neuron 54:653–666.

Schwartz AB (2007) Useful signals from motor cortex. J Physiology 579:581–601.

Seung HS (2003) Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. Neuron 40:1063–1073.

Sober SJ, Wohlgemuth MJ, Brainard MS (2008) Central contributions to acoustic variation in birdsong. J Neurosci 28:10370–10379.

Taylor DM, Tillery SI, Schwartz AB (2002) Direct cortical control of 3D neuroprosthetic devices. Science 296:1829–1832.

Tumer EC, Brainard MS (2007) Performance variability enables adaptive plasticity of 'crystallized' adult birdsong. Nature 450:1240–1244.

Velliste M, Perel S, Spalding MC, Whitford AS, Schwartz AB (2008) Cortical control of a prosthetic arm for self-feeding. Nature 453:1098–1101.

Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach Learn 8:229–256.

Xie X, Seung HS (2004) Learning in neural networks by reinforcement of irregular spiking. Phys Rev E Stat Nonlin Soft Matter Phys 69:041909.