Network: Computation in Neural Systems
September 2006; 17(3): 235–252

informa
healthcare

Dimensional reduction for reward-based learning

CHRISTIAN D. SWINEHART[1] & L. F. ABBOTT[2]

[1] *Volen Center for Complex Systems, Department of Biology, Brandeis University, Waltham, MA 02454-9110, USA*
[2] *Center for Neurobiology and Behavior, Department of Physiology and Cellular Biophysics, Columbia University College of Physicians and Surgeons, New York, NY 10032-2695, USA*

(Received 22 December 2005; accepted 26 April 2006)

Abstract
Reward-based learning in neural systems is challenging because a large number of parameters that affect network function must be optimized solely on the basis of a reward signal that indicates improved performance. Searching the parameter space for an optimal solution is particularly difficult if the network is large. We show that Hebbian forms of synaptic plasticity applied to synapses between a supervisor circuit and the network it is controlling can effectively reduce the dimension of the space of parameters being searched to support efficient reinforcement-based learning in large networks. The critical element is that the connections between the supervisor units and the network must be reciprocal. Once the appropriate connections have been set up by Hebbian plasticity, a reinforcement-based learning procedure leads to rapid learning in a function approximation task. Hebbian plasticity within the network being supervised ultimately allows the network to perform the task without input from the supervisor.

Keywords: *PCA, neural networks, functional approximation*

Introduction

Learning often takes place solely through the reinforcement of improved performance. Reinforcement-based learning is challenging because no information is provided to indicate how a task should be done or to suggest how performance can be improved. Instead, strategies must be generated internally and evaluated solely on the basis of the reinforcement they generate.

Faced with such a dearth of information, models of learning often rely on a random-search approach in which reinforcement guides an otherwise random walk in the space of parameters controlling task performance (Barto et al., 1983; Mazzoni et al., 1991; Jabri and Flower, 1992; Williams, 1992; Cauwnberghs, 1993; Doya and Sejnowski, 1995; O'Reilly, 1996; Xie and Seung, 2003; Seung, 2003). In a neural network, such a scheme typically involves randomly changing synaptic strengths or neuronal excitabilities and keeping or rejecting those changes on the basis of reward. For example, in the scheme we use, which is based on bacterial chemotaxis, changes are made by moving along a straight line in the space of parameters as long as performance improves and reward is provided. If at some point reward is denied, indicating worsening performance, the system starts moving in a new, randomly chosen direction.

Correspondence: L. F. Abbott, Center for Neurobiology and Behavior, Kolb Research Annex, Columbia University College of Physicians and Surgeons, New York, NY 10032-2695, USA.

ISSN: 0954-898X print / ISSN 1361-6536 online
DOI: 10.1080/09548980600773215

Reward-based learning strategies of the type described in the previous paragraph typically converge to a set of parameters that optimizes task performance if they are applied for a sufficiently long time. Because the parameter search is random, this time can be very long, and it typically scales badly (i.e. increases dramatically) as the size of the network performing the task increases. In most cases, the space of network parameters is enormous, and the system can easily get lost when the only guide is reinforcement.

Here we explore the idea that the convergence time for reinforcement-based, random-walk learning schemes and its scaling with network size can be improved dramatically by reducing the effective dimension of the parameter space. This is a rather obvious strategy, and we use a standard dimensional reduction method, principal component analysis (PCA) to implement it. The novelty is that we implement both the dimensional reduction scheme and the mechanism by which neuronal excitabilities and synaptic strengths are modified through well-known, local synaptic modification rules acting along biologically plausible pathways.

Most reinforcement learning schemes are based on a form of synaptic plasticity that is modulated by the reward. Although synaptic plasticity can depend on activation of metabotropic glutamate receptors, there is no convincing experimental evidence for such reward-based modulation of synaptic plasticity. Therefore, we take a different approach in which learning arises from the modulatory effects of reward on a network generating ionotropic synaptic input rather than on synaptic plasticity. Our goal is not to outperform standard reinforcement-based learning algorithms, but to introduce a scheme based on non-synaptic targets of reward that we feel is more plausible in light of experimental data.

The task and the network

To explore the role of dimensional reduction in reinforcement learning, we chose a well-defined task of obvious behavioral and cognitive relevance: function approximation (Poggio, 1990). In the network we consider, N input units respond to inputs that are tuned to the value of a particular stimulus parameter (in our case, an angle θ), and they drive an output unit so that its firing rate follows a specified function of the stimulus value. The network can easily be extended to include more than one output unit, but we will not need to do this for our purposes. The learning task consists of adjusting network parameters so that the firing rate of the output unit matches the specified target function. Applying reinforcement learning to this task allows us to illustrate clearly the features and limitations of the scheme we are studying.

The architecture of the network is shown in figure 1. Each input unit is characterized by a firing rate, r_i, with $i = 1, 2, \ldots, N$, that is given by a sigmoidal function of the total synaptic current it recieves. The synaptic current is divided into two terms: a stimulus current $I_i(\theta)$ that depends on the stimulus angle θ, and a bias current $\mathcal{J}_i$ that is independent of the stimulus and represents synaptic currents arising from the supervisor circuit shown in figure 1. Thus,

$$r_i = \frac{1}{1 + \exp\left(-g(I_i(\theta) + \mathcal{J}_i - s)\right)} . \tag{2.1}$$

The parameters s and g control the shape of the sigmoid. The shift parameter, s, which is set to 0.9 for all the simulations shown, determines the location at which the firing rate reaches its half-maximal value, and the gain parameter, g, set to 5 in all cases shown, specifies the slope at that half-maximal point. The firing rate is normalized so that its maximum value is 1.

The stimulus current $I_i(\theta)$ that appears in equation 2.1 is constructed from Gaussian functions of the difference between the stimulus variable θ and the preferred stimulus value for unit i, θ_i,

$$I_i(\theta) = G(\theta - \theta_i) + G(\theta - \theta_i - 2\pi) + G(\theta - \theta_i + 2\pi)\,, \tag{2.2}$$

where

$$G(\theta) = 1.5 \exp\left(-\frac{\theta^2}{2}\right) - 0.5. \tag{2.3}$$

The three terms appearing in equation 2.2 impose an approximate periodicity on the network to match the fact that θ is an angle. This stimulus current makes the input units selective for different values of θ, and we can write their firing rates as $r_i(\theta)$. The preferred stimulus values, θ_i for $i = 1, 2, \ldots, N$, are uniformly distributed over the range from 0 to 2π (see figure 1), so the input units collectively represent stimulus values over the entire range of angles.

The output unit is driven by the input units through a set of modifiable synapses, so its firing rate R is given by a weighted sum of their rates,

$$R(\theta) = \sum_{i=0}^{N} w_i r_i(\theta)\,. \tag{2.4}$$

The weight w_i represents the strength of the synapse from input unit i to the output unit. The task for learning in this network is to make $R(\theta)$ match, as closely as possible, a specified

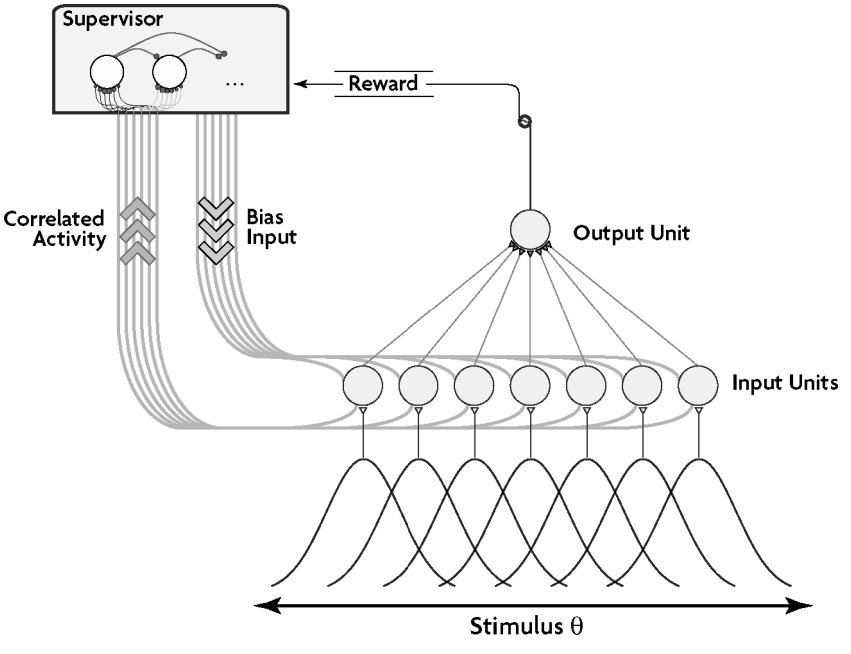

Figure 1. Network for function approximation. Input units (lower circles) are driven by input current that is a Gaussian function of the difference between a stimulus angle θ and a preferred stimulus value for each unit (tuning curves at bottom). The input units send projections to both the output unit (upper circle) and units within the supervisory circuit. Supervisor units synapse onto each input unit. The supervisor receives limited information about network performance in the form of reward which it uses to direct its influence on the input units.

target function $F(\theta)$. In our case, this is done by presenting random values of θ to the network on learning trials and providing feedback in the form of a reward or reinforcement signal that indicates improved performance.

The task of making the output unit match the target function, i.e. setting $R(\theta) = F(\theta)$, is assigned to the supervisor circuit in figure 1. The standard way of doing this is to have the supervisor adjust the synaptic weights between the input and output units of the network using a delta learning rule (Widrow and Hoff, 1960; Widrow and Stearns, 1985). We do not follow this procedure for two reasons. First, the delta rule requires that the supervisor has knowledge of the target function and can determine the error that the network makes on each trial. In reinforcement learning this information is not available. Second, delta-rule learning assumes that the supervisor can control synaptic modification, but there is little evidence for such control of synaptic plasticity in biological circuits. Instead, we assume that the supervisor circuit interacts with the input units solely through conventional excitatory and inhibitory synapses arising from the pathway from the supervisor to the input units depicted in figure 1. The input from the supervisor circuit to input unit i is represented by the bias current $\mathcal{J}_i$. Thus, in our biologically realizable scheme, the supervisor modifies the responses of the input units on the basis of a reward signal through ordinary excitatory and inhibitory synapses.

We have previously studied this form of learning in a supervised rather than reward-based scheme (Swinehart and Abbott, 2005). Two results from that study are relevant to the current work:

- It is possible to get the firing rate of the output unit $R(\theta)$ to approximate the target function $F(\theta)$ solely by having the supervisor adjust the bias currents of the input units to appropriate values. This allows the network to perform the function approximation task as long as inputs from the supervisor units are maintained at proper levels.
- If a Hebbian form of synaptic modification is implemented at the synapses between the input and output units, the information about how to perform the function approximation task transfers spontaneously from the bias currents to the synapses. When this transfer is complete, the network can perform the task without input from the supervisor.

Thus, in this scheme, learning is a two-component process. The excitatory and inhibitory input from the supervisor to the network adjusts the bias currents of the input units to improve network performance. At the same time, the synapses from the input units to the output unit are modified by Hebbian plasticity. This transfers the improvements induced by supervisory input into permanent changes within the network that ultimately allow it to perform the task successfully, even without supervisory input.

A challenge to this two-component scheme is that reinforcement-based supervision and Hebbian plasticity occur together. If reinforcement learning is too slow or spends too long in the wrong parts of parameter space, the Hebbian component will "lock in" modifications that are detrimental to task performance, which can destroy convergence. Thus, it essential for us to find an efficient reinforcement-based scheme that established the correct biases in the input units rapidly enough to keep the unsupervised Hebbian plasticity mechanism on track. For this reason, we begin our study of reinforcement learning by focusing solely on reinforcement-based modification of input unit bias currents by the supervisor circuit, leaving the analysis of the additional Hebbian component until the end.

The reinforcement-based supervisor

The supervisor circuit shown in figure 1 is responsible for making adjustments to the network on the basis of reward reinforcement to improve performance. To do this, the supervisor must explore the space of network parameters by generating different patterns of excitation and inhibition to the input units.

In a direct application of these ideas to the network of figure 1, the supervisor circuit would consist of N units with activities v_i, for $i = 1, 2, \ldots, N$, one for each of the N input units (we allow v_i to be either positive or negative to represent excitatory or inhibitory inputs provided by the supervisor even though the firing rates of the individual supervisory units would, of course, be positive). All biases start out at zero, but during learning these are changed by an amount $\Delta \mathcal{J}_i = \epsilon v_i$, where ϵ is a learning-rate constant and v_i is the ith component of a vector describing the pattern of activity in the supervisor circuit. This pattern of activity is similar to the sustained activity seen in models of short-term memory (Compte et al., 2000; Seung et al., 2000), and it could easily be generated by such a model (although we will not do this here).

The direct reinforcement-based random walk strategy can now be specific in terms of the effects of reward on the components v_i that describe the supervisory activities and, in turn, determine the bias currents for the input units. As long as reward is obtained, all these components, v_i for $i = 1, 2, \ldots, N$ remain fixed. This could be due to the effects of a reward-induced modulator. If reward is not obtained, the vector describing the pattern of supervisory activity is changed by choosing a new set of components v_i randomly. This could be realized by having a second modulator or noisy input temporarily disrupt the stability of the self-sustained activity in the supervisory circuit. These two actions, and the rule for changing biases, $\Delta \mathcal{J}_i = \epsilon v_i$, completely describe the learning strategy we use. Thus far, we have discussed it and its implementation within the context of a direct approach, not the dimensionally reduced scheme we propose.

We have shown previously that the direct scheme can work if N is small enough, but it is slow and gets slower as N increases (Swinehart and Abbott, 2005). The basic feature we exploit to get around the limitations of the direct approach is the fact that the firing of different input units is correlated. Two units with highly overlapping input tunings tend to fire together at similar rates. If the function being approximated varies slowly on the scale of the separation between preferred stimulus values (i.e. if $F(\theta_i) \approx F(\theta_{i+1})$, it does not make sense to vary the two bias currents $\mathcal{J}_i$ and $\mathcal{J}_{i+1}$ independently. The best strategy is to vary those combinations of bias currents that have the biggest impact on network output. These combinations can be determined by performing principal component analysis (PCA) on the correlation matrix of the input units. The key result, present in the following section, is that this can be accomplished by applying appropriate forms of synaptic plasticity to the synapses between the input units and the units of the supervisor circuit.

Dimensional reduction

The learning strategy based on dimensional reduction is similar to the direct reward-guided random-walk strategy discussed above, except that the dimensionality of the space being searched is much smaller. In this case, on time steps when learning is applied, the modification of the bias current is controlled by only $n \ll N$ supervisor units through the equation

$$\Delta \mathcal{J}_i = \epsilon \sum_{a=1}^{n} w_{ia} v_a \,, \tag{3.5}$$

where v_a for $a = 1, 2, \ldots n$ are the activities of the n supervisor units, and w_{ia} represents the strength of the synapse from supervisor unit a to input unit i. The implementation of reinforcement learning in this dimensionally reduced scheme proceeds exactly as described above for the direct approach. After rewarded trials all the components v_a for $a = 1, 2, \ldots, n$ remain the same, and after non-rewarded trials they are reset to randomly chosen values.

The key to making the modifications described by equation 3.5 as effective as possible is to make sure that the synaptic weight w_{ia} is proportional to the ith component of the ath principal component of the input-unit correlation matrix. In other words, we want $w_{ia} \propto \xi_i^a$, where ξ_i^a is the ith component of the eigenvector of the input-unit correlation matrix with the ath largest eigenvalue. To achieve this, we exploit a virtually universal property of neural circuits, the reciprocal nature of inter-connections. The supervisor can obtain information about the correlations between the input units through the projections from the input units to the supervisor depicted in figure 1. We refer to these projections as the ascending pathway to the supervisor, and call the projections from the supervisor to the input units the descending pathway. Methods exist for setting the ascending synaptic weights proportional to the principal component eigenvectors of the input-unit correlation matrix (Oja, 1989; Sanger, 1989), but this is the wrong set of synapses for our purposes. We need to set the descending synaptic weights proportional to these eigenvectors. We now show that this can be achieved by applying an additional ordinary Hebbian plasticity to the descending synapses.

Initially, the supervisor units are driven by the input units. We denote the strength of the ascending synapse from input unit i to supervisor unit a by w'_{ai}, so the firing rate of supervisor unit a is given by

$$v_a = \sum_{i=1}^{N} w'_{ai} r_i \, . \tag{3.6}$$

The first step in setting the descending synaptic weights is to apply a Sanger rule (Sanger, 1989) to the ascending synapses. This is a form of synaptic plasticity that is essentially Hebbian, but with the added wrinkle of subtracting out contributions already accounted for by other supervisor units. The resulting modification amounts to the replacement

$$w'_{ai} \rightarrow w'_{ai} + \eta' v_a \left(r_i - \sum_{b=1}^{a} v_b w'_{bi} \right) \tag{3.7}$$

on every time step of the simulation, where η' sets the learning rate and is set to the value 0.25, and v_a and r_i are the firing rates of supervisor unit a and input unit i, respectively. For the $a = 1$ supervisor unit, this rule is identical to the standard Oja rule which sets $w'_{1i} \propto \xi_i^1$ (Oja, 1982; see below). For the other supervisor units, the summed term in equation 3.7 assures that $w'_{ai} \propto \xi_i^a$ for $a = 2, 3, \ldots n$. The Sanger rule thus sets the connection strengths in such a way that the weight vector from the input units to the $a = 1$ supervisor unit is proportional to the eigenvector with the largest eigenvalue, the $a = 2$ unit to the eigenvector with the next largest eigenvalue, and so on.

The second step in setting the descending synapses properly is to transfer the weights from the ascending to the descending pathway. Surprisingly, this can be done by having ordinary, multiplicatively constrained Hebbian plasticity act on the descending synapses. Specifically, we apply the Oja plasticity rule (Oja, 1982) to the descending synapses,

$$w_{ia} \rightarrow w_{ia} + \eta r_i \left(v_a - r_i w_{ia} \right) \, , \tag{3.8}$$

on every time step of the simulation, where the parameter η controls the learning rate and is set to 0.005. The weight decay term, $r_i w_{ia}$, in this rule ensures that the weight vector converges toward unit length, preventing the runaway potentiation that would arise from a purely Hebbian rule.

We now consider what happens if, after the ascending weights have been set to the principal component eigenvectors by the Sanger rule (equation 3.7), the Oja rule (equation 3.8) acts on the descending synapses while the supervisor units are driven by the input units according to equation 3.6. The key result is that the descending synaptic weights are set to $w_{ia} \propto \xi_i^a$ for $i = 1, 2, \ldots, N$ and $a = 1, 2, \ldots n$. To see how this arises, we note that, when plasticity comes to equilibrum, the Oja rule (equation 3.8) sets the synaptic weights proportional to the pre-postsynaptic correlation matrix (if η is small enough),

$$w_{ia} = c_a \langle r_i v_a \rangle \,, \tag{3.9}$$

where the angle brackets denote an average over time and c_a is a constant that may be different for different values of a. Combing this result with equation 3.6, we find that

$$w_{ia} = c_a \sum_{j=1}^{N} \langle r_i r_j \rangle w'_{aj} = c_a \lambda_a w'_{ai} \propto \xi_i^a . \tag{3.10}$$

The second and third equalities follow from the fact that w'_{ai} is proportional to ξ_i^a and is thus an eigenvector of the correlation matrix with eigenvalue λ_a,

$$\sum_{j=1}^{N} \langle r_i r_j \rangle w'_{aj} = \lambda_a w'_{ai} \,. \tag{3.11}$$

The above derivation shows that, when the ascending synaptic weights are proportional to eigenvectors of the input-unit correlation matrix, the Oja rule sets the descending weights proportional to the ascending weights, and thus proportional to those same eigenvectors. Thus, the mixture of a Sanger rule on the ascending synapses and a Oja rule on the descending synapses accomplishes the goal of setting the descending weights equal to the values needed for the dimensional reduction of reinforcement-based learning. The fact that we only required that equation 3.9 be satisfied by the learning rule for the descending synapses indicates that any correlation-based form of plasticity, not only the Oja rule, can be used for this purpose.

Figure 2 illustrates how the ascending and descending weights become proportional to the principal eigenvectors through synaptic plasticity. In this example, the input units were driven by randomly chosen angles activating the input currents of equation 2.2 while the Sanger and Oja rules were applied to the ascending and descending synapses, respectively. Because the tuning curves of the input units were placed uniformly over the range of stimulus angle values, assuring translational invariance ($\theta \to \theta +$ constant is a symmetry), the eigenvectors of the input correlation matrix are sine and cosine functions of the input unit preferred angles θ_i and the principal components are ordered by wavelength. In other words, PCA is equivalent in this case to Fourier analysis. This makes it easy to see that the appropriate synaptic assignments have been made.

The development of the ascending synaptic weights from the input units onto seven supervisor units on the basis of the Sanger rule can be seen in the left panels of figure 2 and the progression of the corresponding descending weights according to the Oja rule is shown in the right panels. From random initial values (top row), the weights to and from the $a = 1$ supervisor unit become equal to the constant Fourier component by 128 trials (the second row of plots in figure 2). As the plasticity proceeds (rows 3–5 in figure 2 representing the results after 256, 1472, and 23,552 trials, respectively), the additional weights to and from

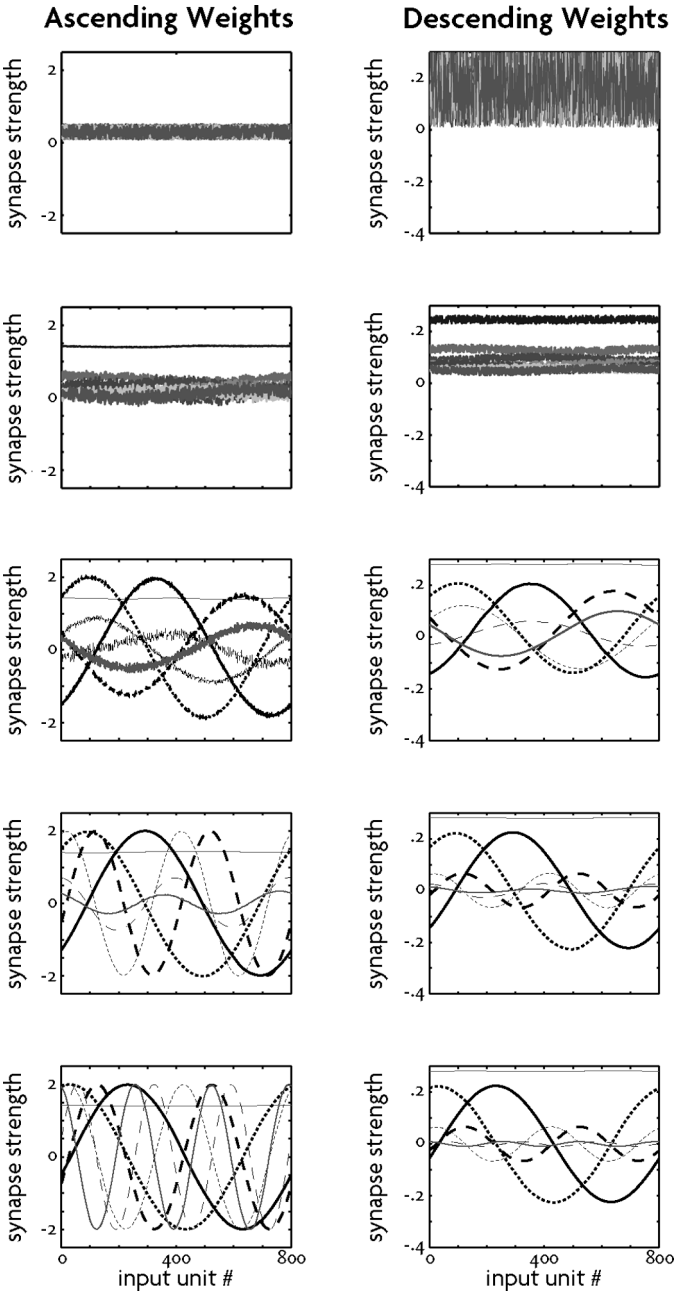

Figure 2. Development of ascending and descending synaptic weights. Each line represents the set of synaptic weights, with $N = 800$, to and from one supervisor unit plotted against input unit number. The left column depicts the ascending connections from the input units to the supervisor, with synapses modified over time, from top to bottom, using the Sanger rule. The right column shows the same thing for descending synapses from the supervisor to the input units, which are controlled by an Oja rule. In the top row, before the plasticity changes them, the weights have random values. Over time (rows below the top show results after 256, 1472, and 23,552 trials), the Sanger rule extracts the principal component eigenvectors from the input correlations and sets the ascending weights proportional to them. Simultaneously, the descending weights are modified by the Oja rule and become proportional to the eigenvectors as well.

the $a = 2, 3, \ldots 7$ supervisor units take sine and cosine forms. During this process, the development of the descending synapses tends to lag behind that of the ascending synapses because the Sanger rule pulls out the principle components while the Oja rule transfers these components to the descending weights. By the lowest panels, when the system has equilibrated, the weights of the synapses received by each supervisor unit are paired with weights of the synapses they send to the input units, both being proportional to the same principle component eigenvector. The amplitudes of the descending weights are not the same as their ascending partners. These amplitudes are related to the corresponding eigenvalues. The unevenness of the amplitudes has no effect on the reinforcement learning strategy we study next.

Dimensionally reduced reinforcement learning

The dimensional reduction process described in the previous section reduces the problem of learning to setting appropriate supervisor firing rates v_a for $a = 1, 2, \ldots, n$ to introduce the appropriate biases into the input units to generate the desired responses in the output unit. This is a much simpler problem than trying to set the bias currents independently, and the use of descending weights proportional to the principal component eigenvectors of the input-unit correlation matrix assures that a small number ($n = 7$ in the all of our examples except for figure 7) of supervisor units has the maximum impact on the activity of the output unit. The price of lowering the dimensionality of the control process from N, which is typically in the thousands in our examples, down to $n = 7$ will be addressed after we show how the basic scheme works.

Recall that the dimensionally reduced reinforcement learning process for the supervisor unit activities consists of keeping them constant if reward is given and changing them randomly if reward is denied. When the network is in the optimal configuration for task performance, any changes will result in an increased error. To address this type of end-stage thrashing, the learning rate parameter ϵ is decreased as the overall size of the errors made by the network goes down.

Before showing the results of dimensionally reduced reinforcement learning, we need to discuss how reward is delivered to the supervisor to determine whether or not its pattern of activity is changed.

The reward procedure

Reward criteria are determined by external factors not by neural circuits, but neural circuits determine how reward is interpreted and what results follow from receiving reward. There is no unique model of such a reward procedure. The reinforcement learning scheme we propose will work with any reasonable reward scheme, where reasonable means that reward is based on some cumulative assessment of improvement integrated over a long enough time to assess whether or not a particular set of parameter changes is beneficial. We use a simple scheme to demonstrate that reinforcement learning can work. Undoubtedly, performance could be improved over what we show by using a more elaborate and clever reward schedule, but our purpose is to study the actions of the supervisory circuit, not the reward system.

A trial in our learning scheme consists of the presentation of a randomly chosen stimulus angle θ, which produces a network output $R(\theta)$ that is suppose to match a target function $F(\theta)$. The reward procedure is based on whether the function approximation error, $(R(\theta) - F(\theta))^2$, shows an increasing or decreasing tendency. To assess this, we computed it over a number of trials with different θ values. The reward after any trial is based on errors

accumulated over the previous 140 trials. These 140 trials are divided into two blocks of 70. The errors for the most recent block of 70 trials are summed, as are the errors for the next most recent block of 70 trials (the 70 trials prior to the most recent block). If the summed error for the most recent block of trials is less than the summed error for the next most recent block, reward is given. If the summed error for the most recent block of trials is larger than that of the next most recent block, reward is denied. This procedure is simply a way of provided reward when trial-averaged performance has improved.

Demonstration of reinforcement learning

Dimensionally reduced reinforcement learning is illustrated in figure 3. The left column in this figure shows the response of the output unit as a function of stimulus angle and the target function that is to be matched. The right column shows the individual contributions to the total bias current provided to each input unit by each of the 7 supervisory units.

In this simulation, all the biases start at zero (figure 3a right panel), which causes all input units to respond identically to their preferred stimuli. This, combined with the fact that synaptic weights between the input units and the output unit are set equal results in the output unit response being independent of the stimulus (figure 3, top-left panel).

After 1600 trials, (second row of figure 3), the supervisor units have biased the input units, resulting in modulation of the flatline response of the initial state. The right panel indicates that this response arises primarily from a baseline shift and a single-cycle cosine modulated bias current that is not well chosen for the target function. However, the components that ultimately will be responsible for a successful function approximation also start to develop at this point. After 5632 trials (third row of figure 3) a single-cycle sine has become the dominant factor contributing to the approximation. After 11,008 trials (fourth row of figure 3) the general shape of the function has been captured by the uniform shift combined with this sine wave, but higher frequency components need to be recruited to account for the detailed shape of the target function. The final state of the bias currents in the bottom row shows that two higher frequency sinusoids now contribute, resulting in a successful approximation. Note that the supervisor units with the highest frequency components were not needed for this particular target function.

Non-uniform input sampling

The connectivity patterns we have seen thus far are equivalent to Fourier modes, but the scheme we propose is not limited to this translationally invariant case. To show this, we break the translational invariance by making the distribution of preferred stimulus values for the input units non-uniform. Two examples of this are shown in figure 4.

For the left column of figure 4, a bimodal distribution was used in which the peaks had a sampling density triple that of the low-density regions. The ascending weights (middle panel at left) that result when the synaptic weights have equilibrated are decidedly non-Fourier, and are well adapted to the activity correlations among the input units that arise from the bimodal distribution of preferred angles. Rather than being flat, the first component is bimodal with troughs aligned with the centers of the regions where input coverage is less dense. The next two components are also non-sinusoidal with dimpled peaks aligned with the central low-density region. Nevertheless, there is a doubling of frequency between the second and third components similar to what is seen in the Fourier case. Another frequency doubling can be seen between the fourth and fifth components. The bottom left panel of figure 4 shows that

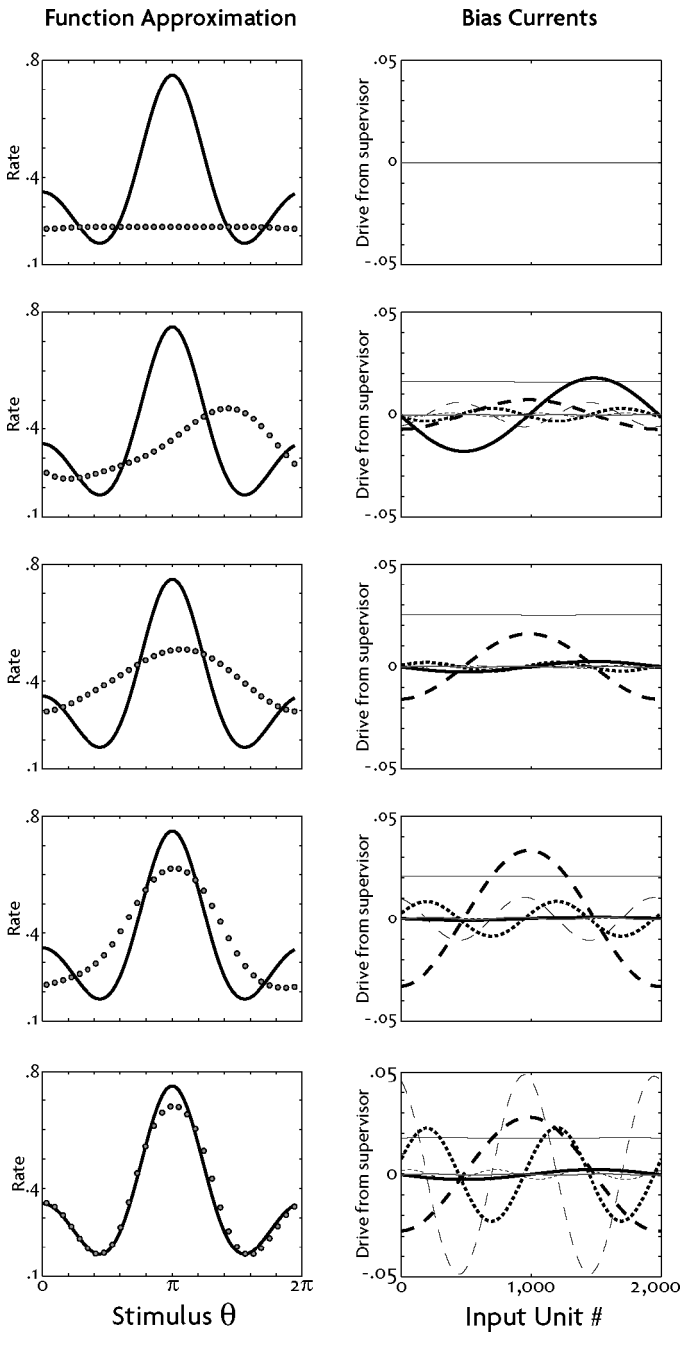

Figure 3. Dimensionally reduced reinforcement learning. The left column shows the target function (solid line) and the output of an $N = 2000$ network for different values of the stimulus angle (open circles). The right column depicts the bias currents sent to the network by the supervisor as a function of the input unit number, with each supervisor unit represented by a separate line. The total bias current received by each input unit can be determined by summing theses curves. Rows show results after 0, 1600, 5632, 11,008, and 310,400 trials. As the bias currents develop through the reinforcement-based random walk procedure, the initially poor approximation (top row) changes into a fairly accurate approximation of the target function (bottom row).

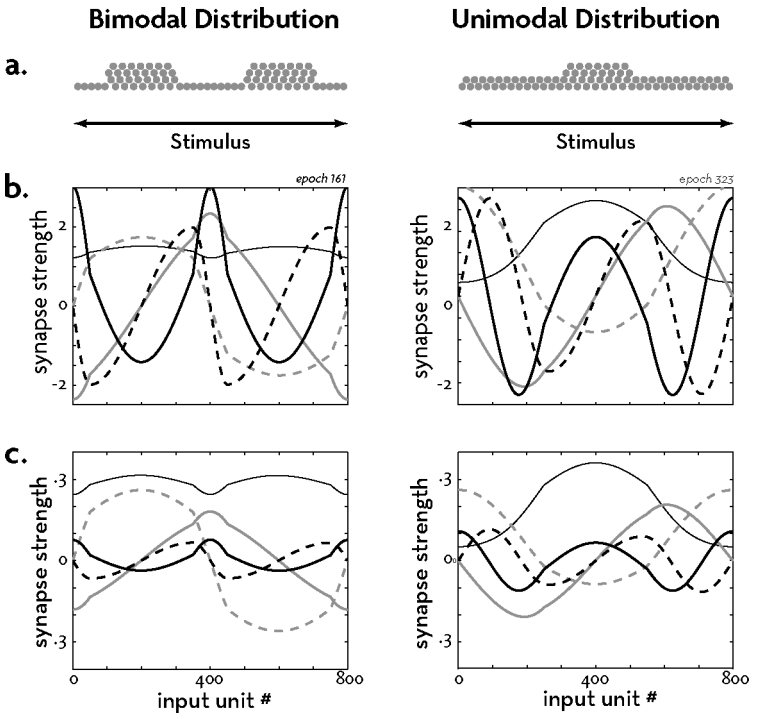

Figure 4. Ascending and descending weights for nonuniform distributions of preferred stimulus angles. a) The two alternative distributions used. When the preferred stimulus angles of the input units are not uniform, the components extracted for the ascending units by the Sanger rule (b) and transferred to the descending weights by the Oja rule (c) are non-Fourier. The results are for $N = 800$, the left panels are after 10,304 trials, and the right panels are after 20,672 trials.

the Oja rule sets the descending weights proportional to the ascending weights in this case as well.

Similar results are seen in the right column of figure 4 for a unimodal distribution of preferred stimulus angles, with the central region having a density twice that of the surrounding region. Again the baseline shift of the first component is distorted, reflecting the non-uniformity in the input population. The second and fourth components are centered around the high-density peak and are frequency-doubled versions of each other, whereas the third and fifth components are more sine-like but also show frequency-doubling.

Figure 5a shows the initial state for a network with the unimodal sampling distribution seen in figure 4. Although the input units all respond to their preferred stimuli identically and the synaptic weights from the input units to the output unit all take the same value as in figure 3, the output unit response depends on the stimulus angle. The response is largest for stimuli near the middle of the range because that is where the largest number of input units respond. As a result of this, the initial approximation is significantly worse than it would be in the uniform case, and the supervisor must overcome this to fit the target function. Nevertheless, as can be seen in Figure 5b, once the reward-guided random walk sets the activities of the superisor units properly, the network successfully approximates the target function. Thus, the synaptic strength distributions extracted by the Sanger process and communicated to the network through Oja plasticity provide a general solution for constructing an efficient supervisor strategy regardless of the specific input population statistics.

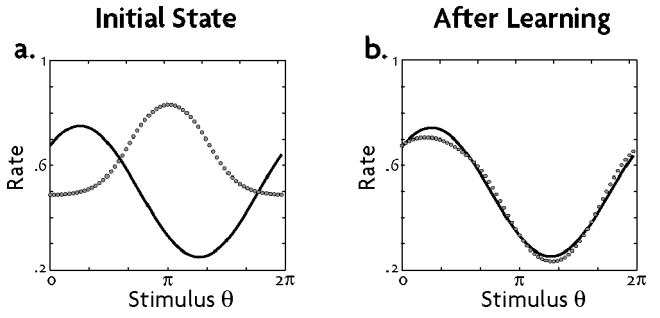

Figure 5. Function approximation with a non-uniform stimulus sampling. The solid line is the target function and the open circles indicate the output unit response. $N = 800$. a) When the sampling density of the input units is not constant, the initial response of the output unit depends on the stimulus angle; a higher density of input units near $\theta = \pi$ causes a larger output response in that region. b) A reward-guided random walk procedure, dimensionally reduced with the appropriate principal components, nevertheless produces a successful approximation of the target function.

Effects of input population size

A critical advantage of the dimensional reduction implemented by the combination of Sanger and Oja rules is that the number of supervisor units is independent of the number of input units. This means that the poor scaling behavior of the direct reward-based random walk approach as a function of network size is replaced by a scheme in which convergence time is essentially independent of network size. This is shown in figure 6.

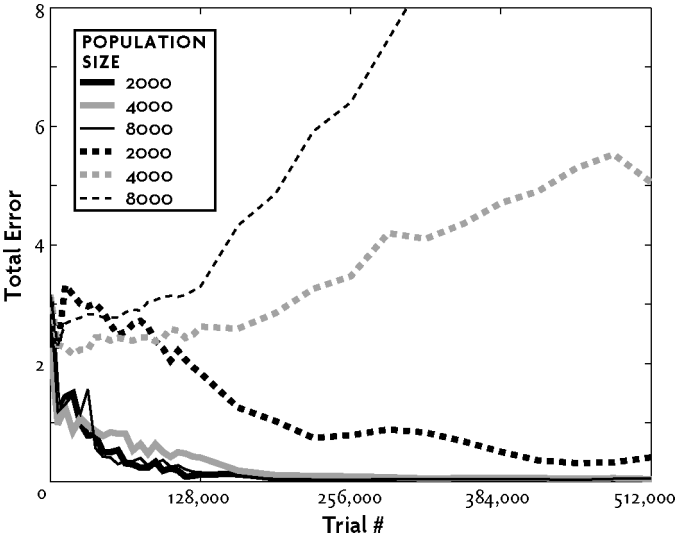

Figure 6. Scaling of dimensionally reduced and direct reward-based random walk learning strategies with network size. The total function approximation error is plotted over the course of learning trials for small, medium, and large networks ($N = 2000$, 4000 and 8000). The function being approximated here is the same as in figure 3. Dotted lines show errors for the direct random walk strategy, which converges slowly for $N = 2000$ but fails to converge over the time period shown for $N = 4000$ and 8000. Solid lines correspond to the dimensionally reduced scheme and are fairly invariant with respect to population size.

Error values after different numbers of trials for the direct learning process are plotted as dotted lines in figure 6. The direct scheme performs acceptably for moderate input population sizes (e.g. 2,000 units), but once there are 4,000 or more input units, the direct approach fails to converge over the number of trials shown. Although the random walk algorithm ensures that the supervisor will change any strategy that increases its error, a series of changes that continue to be detrimental may result before a better strategy is found, causing a prolonged increase in error magnitude as seen in figure 6 for the 4,000- and 8,000-unit cases.

The PCA approach, on the other hand, is effectively invariant with respect to the input population size (solid lines in figure 6). In this scheme, the burden of dealing with more input units is handled by the Sanger/Oja learning procedure on the connections between the input and supervisor units prior to any function learning. Though there seems to be slightly more noise in the early stages of learning when the population size is larger, overall the solid traces in figure 6 are virtually overlapping. This is not surprising because the dimension of the parameter space being searched is independent of network size in the dimensionally reduced case, and the principal components have the same shape over the input distribution independent of its size. Thus the PCA supervisor is capable of scaling to much larger population sizes than the direct supervisor, and it does so with no noticable performance degredation during the learning phase. Efficient scaling with respect to system size has also been achieved in standard reinforcement learning procedures (Arleo and Gerstner, 2000; Foster et al., 2000; Doya, 2000, Stroesslin et al., 2005).

Number of components and diminishing returns

In the uniform case, supervision through PCA-based dimensional reduction is equivalent to approximating a waveform by scaling its Fourier modes. Drawing on more Fourier modes by increasing the number of supervisor units improves the potential accuracy of the final function approximation. This effect can be seen in the results shown in figure 7, which indicates the final error in approximating the same function as in figure 3 using different numbers of supervisor units.

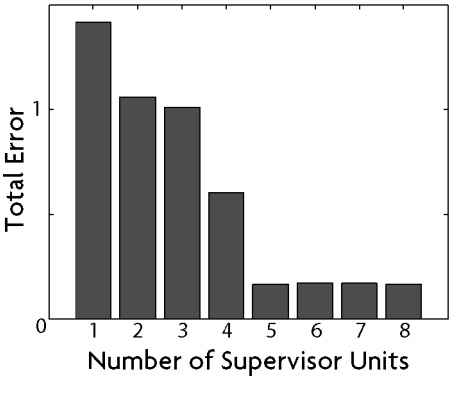

Figure 7. Function approximation errors for different numbers of supervisor units. The bars indicate the final error reached for various numbers of supervisor units (values of n) when the simulation is allowed to run until improvement stops. Adding additional principal components is an effective means for improving accuracy, but the improvement saturates rather quickly. In this case it appears that using any more than five components is unnecessary, and using more than this slows learning.

Two noteworthy effects are revealed by this study. The first is that using a smaller number of components leads to a faster convergence toward the optimal solution. For example, in the single-component case, the supervisor finds its optimal (though still quite poor) approximation in about 1000 trials, while the eight-component supervisor requires nearly 20 times as many trials to reach its more globally optimal solution. Second, increasing the number of components does in fact reduce the error in the ultimate approximation reached through learning as can be seen by the steady decreases in the higher unit-count cases. However, this effect tends to saturate. Clearly adding more supervisory units improves performance up to a point, but in this case showing it is not necessary to use more than five components.

Of course, the number of components used depends on the target function being approximated. Were it less complicated and more easily approximated by a pair of low-frequency sinusoids, improvement would stop after the third component. If the target function varied quickly with respect to the stimulus value, more than five components would be needed to capture the higher-frequency structures.

Including Hebbian modification of network synapses

The most obvious benefit of reducing the dimensionality of the space that the supervisor must search is that it allows a simple search algorithm to work. We now explore an additional benefit, that it allows a reinforcement-based random walk strategy to bring the network to a successful state sufficiently quickly to guide Hebbian synaptic plasticity within the network itself. When Hebbian plasticity is applied to the synapses from the input units to the output unit, the slower speed with which the direct supervisor guides learning does not just delay success, it prevents the network from performing the task.

Up to now, the synaptic weights between the input units and the output unit were held fixed throughout learning. We now allow them to be modified by an Oja plasticity rule during the learning process,

$$w_i \rightarrow w_i + \eta'' R (r_i - R w_i) \ , \tag{3.12}$$

with η'', which was set to 0.0003, determining the learning rate. As we have demonstrated previously (Swinehart and Abbott, 2005), applying Hebbian-type plasticity to these synapses causes the learning originally applied by the supervisor through bias currents to transfer to the synaptic weights. Ultimately, this process allows the network to perform the function approximation task even when the supervisory bias currents are removed. Provided that the supervisory biases are established quickly enough relative to the time scale of Hebbian plasticity, these two processes can take place simultaneously.

An example of this process is seen in figure 8a. In both columns of figure 8, the output response of the network in the presence of supervisory bias is plotted with filled dots, and open dots show this response when the supervisory input is removed. By the middle panel of figure 8a, Hebbian effects have begun to change the network, as evidenced by the inflections in the curve of output firing rate versus stimulus angle in the absence of bias currents. Ultimately, this leads to almost equal network performance with and without the supervisory bias currents (bottom panel, figure 8a).

The situation is quite different for the direct supervisor (figure 8b). Because the direct supervisor has difficulty with an input population of this size ($N = 4000$), Hebbian plasticity locks in correlations that reflect incorrect network configurations and the network fails to develop or attain the correct biases and synaptic weights to perform the task either with or without supervisory input (middle and bottom panels of figure 8b).

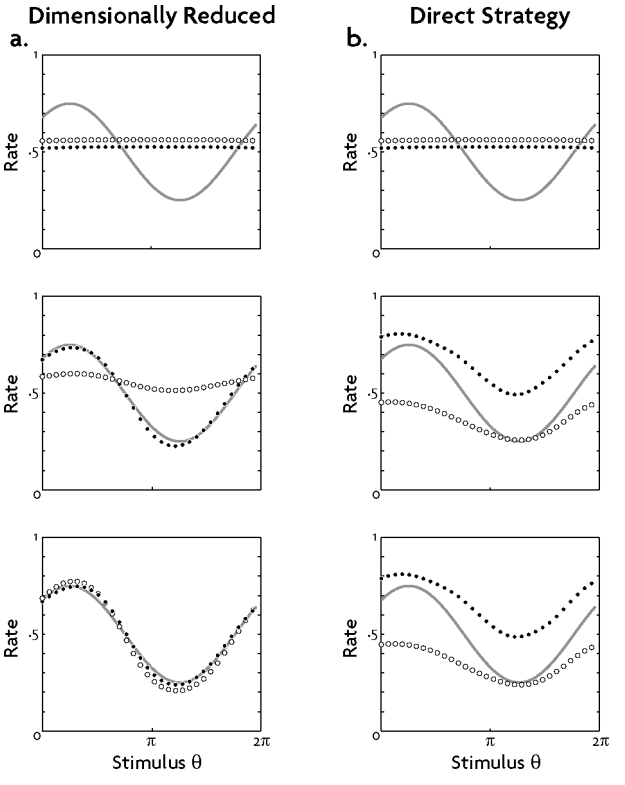

Figure 8. Learning with Hebbian plasticity. When the synapses between the input units and output unit are modulated by an Oja rule, the correlations imposed by the supervisor through the bias currents transfer to the network weights. In all panels, the solid line denotes the target function, the solid dots show the responses of the output unit in the presence of supervisory bias currents, and the open circles show the output unit responses when supervisory activity is turned off and bias currents are zero. The top panels show the initial state of the network, and the middle and lower panels show intermediate and convergent results, respectively. a) With dimensional reduction, the reward-based random walk results in learning that is transferred to the synaptic weights by Hebbian plasticity. Ultimately (bottom panel) the supervisory bias currents can be removed and the network still performs appropriately. b) In the direct approach with no dimensional reduction, the supervisory biases establish too slowly, and Hebbian plasticity locks in inaccuracies. The task cannot be performed either with or without supervisory bias currents.

Discussion

Neural circuits often represent stimulus variables through the activities of large populations of neurons with overlapping tuning curves, providing an over-complete basis for sufficiently smooth functions of those variables (Salinas and Abbott, 2000). We have exploited this over-completeness to streamline reinforcement learning by performing a PCA dimensional reduction of the space of parameters to be searched. By taking advantage of reciprocal connections between the supervisory circuit and the network it is supervising, we showed that this can be done by standard synaptic plasticity rules. The result is efficient reinforcement-based learning that scales well with network size. Unsupervised Hebbian plasticity applied simultaneously to network synapses ultimately allows the network to perform the task autonomously, so the supervisor and reward system are only needed during training.

In our scheme, the supervisor introduces variability into the output of the network as it explores the range of bias currents in an attempt to obtain a reward. A pair of recent papers

provide evidence for this type of supervisor circuit in the birdsong system (Kao et al., 2005; Ölveczky et al., 2005). Both studies examine the role of a basal ganglion-like nucleus called LMAN that innervates the vocal motor pathway. When LMAN is silenced, the degree of variability in the song produced by a bird is greatly reduced (Kao et al., 2005; Ölveczky et al., 2005). In addition there is some evidence that the LMAN signal guides song production in a specific way, because eliciting a particular pattern of LMAN activity is correlated with a specific variation in the song (Kao et al., 2005). These results suggest that LMAN in birds, and perhaps the basal ganglia in mammals, may be the site of the biological correlate of the supervisor circuit we have been discussing.

Introducing variability is one essential feature of the supervisor we have been studying because it corresponds to exploration of the space of network parameters. Another is modulation of that variability by reward. It will be interesting to see if LMAN is modulated by reward signals, such as dopamine (Schultz et al., 1997), in a manner consistent with the proposed scheme.

Acknowledgments

Research supported by the National Science Foundation (IBN-9817194) and by an NIH Director's Pioneer Award, part of the NIH Roadmap for Medical Research, through grant number 5-DP1-OD114-02.

References

Arleo A, Gerstner W. 2000. Spatial cognition and neuro-mimetic navigation: a model of hippocampal place cell activity. Biol Cybern 83:287–299.

Barto AB, Sutton RS, Anderson CW. 1983. Neuron-like adaptive elements that can solve difficult learning control problems. IEEE Trans on Systems, Man and Cybernetics 13:834–846.

Cauwnberghs G. 1993. A fast stochastic error-descent algorithm for supervised learning and optimization. Adv Neural Info Proc Sys 5:244–251.

Compte A, Brunel N, Goldman-Rakic PS, Wang XJ. 2000. Synaptic mechanisms and network dynamics underlying spatial working memory in a cortical network model. Cereb Cortex 10:910–923.

Doya K. 2000. Reinforcement learning in continuous time and space. Neural Comput 12:219-245.

Doya K, Sejnowski TJ. 1995. A novel reinforcement model of birdsong vocalization learning. Adv Neural Info Proc Sys 7:101–108.

Foster DJ, Morris RG, Dayan P. 2000. A model of hippocampally dependent navigation, using the temporal difference learning rule. Hippocampus 10:1–16.

Jabri M, Flower B. 1992. Weight perturbation—an optimal architecture and learning technique for analog vlsi feedforward and recurrent multilayer networks. IEEE Trans on Neural Networks 3:154–157.

Kao MH, Doupe AJ, Brainard MS. 2005. Contributions of an avian basal ganglion-forebrain circuit to real-time modulation of song. Nature 433:638-643.

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

Oja E. 1982. A simplified neuron model as a principal component analyzer. *J Math Biol* 16:267–273.

Oja E. 1989. Neural networks, principal components, and subspaces, International Journal of Neural Systems 1, 61–68.

Olveczky BP, Andalman AS, Fee MS. 2005. Vocal experimentation in the juvenile songbird requires a basal ganglia circuit. PLoS Biology 3:1:8.

O'Reilly RC. 1996. Biologically plausible error-driven learning using local activation differences: The generalised recirculation algorithm. Neural Computation 8:895–938.

Poggio T. 1990. A theory of how the brain might work. Cold Spring Harbor Symposium on Quantitative Biology 55:899–910.

Salinas E, Abbott LF. 2000. Do simple units in primary visual cortex form a tight frame? Neural Comp 12:313–336.

Sanger, TD. 1989. Optimal unsupervised learning in a single-layer linear feedforward neural network. Neural Networks 2, 459–473.

Schultz W, Dayan P, Montague PR. 1997. A neural substrate of prediction and reward. Science 275:1593–1599.
Seung HS. 2003. Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. Neuron 40:1063-1073.
Seung HS, Lee DD, Reis BY, Tank DW. 2000. Stability of the memory of eye position in a recurrent network of conductance-based model neurons. *Neuron* 26:259–271.
Strösslin T, Sheynikhovich D, Chavarriaga R, Gerstner W. 2005. Robust self-localisation and navigation based on hippocampal place cells. Neural Network 18:1125-1140.
Swinehart C, Abbott LF. 2005. Supervised Learning Through Neuronal Response Modulation. Neural Computation 17:609-631.
Widrow B, Hoff ME. 1960. Adaptive switching circuits. WESCON Convention Report 4:96–104.
Widrow B, Stearns SD. 1985. Adaptive Signal Processing. Englewood Cliffs, nj: Prentice-Hall.
Williams RJ. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning 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.