Supervised Learning in a Recurrent Network of Rate-Model Neurons Exhibiting Frequency Adaptation

Pierre A. Fortier
pfortier@uottawa.ca
Department of Cellular and Molecular Medicine, Univ of Ottawa, Canada, K1H 8M5

Emmanuel Guigon
guigon@ccr.jussieu.fr
INSERM U483, Université P. et M. Curie, 75005 Paris, France

Yves Burnod
burnod@isc.cnrs.fr
UMR 5015 CNRS, Université Claude Bernard, 69675 Lyon, France

For gradient descent learning to yield connectivity consistent with real biological networks, the simulated neurons would have to include more realistic intrinsic properties such as frequency adaptation. However, gradient descent learning cannot be used straightforwardly with adapting rate-model neurons because the derivative of the activation function depends on the activation history. The objectives of this study were to (1) develop a simple computational approach to reproduce mathematical gradient descent and (2) use this computational approach to provide supervised learning in a network formed of rate-model neurons that exhibit frequency adaptation.

The results of mathematical gradient descent were used as a reference in evaluating the performance of the computational approach. For this comparison, standard (nonadapting) rate-model neurons were used for both approaches. The only difference was the gradient calculation: the mathematical approach used the derivative at a point in weight space, while the computational approach used the slope for a step change in weight space. Theoretically, the results of the computational approach should match those of the mathematical approach, as the step size is reduced but floating-point accuracy formed a lower limit to usable step sizes. A systematic search for an optimal step size yielded a computational approach that faithfully reproduced the results of mathematical gradient descent.

The computational approach was then used for supervised learning of both connection weights and intrinsic properties of rate-model neurons to convert a tonic input into a phasic-tonic output pattern. Learning

produced biologically realistic connectivity that essentially used a monosynaptic connection from the tonic input neuron to an output neuron with strong frequency adaptation as compared to a complex network when using nonadapting neurons. Thus, more biologically realistic connectivity was achieved by implementing rate-model neurons with more realistic intrinsic properties. Our computational approach could be applied to learning of other neuron properties.

1 Introduction

The building blocks of nervous systems evolved very early. Most of the small neurotransmitters as well as peptides and their associated G-protein coupled receptor systems are present in protozoa (Harris-Warrick, 2000; Ranganathan, 1994). Furthermore, the major families of ion channels probably evolved from prokaryote precursors, and most of the major classes of ion channels were present about a billion years ago by the time the first nervous systems began to evolve (Harris-Warrick, 2000). Modern families of animals share a similar set of ion channel genes, yet there has been considerable evolution in channels. Changes have been in domains that are not responsible for channel formation but for voltage, kinetic, or other properties of channels. Single neurons and even neuronal circuits can show dramatic alterations in activity with only minute changes in channel properties (Katz & Harris-Warrick, 1999). Thus, evolutionary change has packaged enormous information processing power by utilizing not only connectivity but also intrinsic neuronal properties in the formation of nervous systems (Barish, 1988; Finlay & Darlington, 1995).

Neurobiologists are revealing more and more of the connectivity and intrinsic properties of neurons in an attempt to reveal the representation and processing of information by nervous systems and also to explain the mechanisms underlying behavior. Among all the nervous systems that have been studied, the cerebral cortex still continues to be one of the most intensively studied nervous tissues because of its unique ability in humans to implement adaptive measures and realize creative genius. Animal research has provided considerable detail on the synaptic inputs to the cerebral cortex, the intrinsic cortical circuitry and neuronal firing properties, and the output projections to extracortical regions (Jones, 2000; deCharms & Zador, 2000). This detail is nonetheless insufficient to fully explain the transformation of information and its ultimate contribution to behavior. There is a need for simultaneous information on the connectivity and activity of neurons contributing to cortical network behavior. The most direct approach would be to obtain this information in live preparations of cortical tissue, but this involves technically challenging experiments. A less direct but more feasible approach would be to get a preview of this information from neural network simulations of cortical tissue (Arbib & Erdi, 2000; Koch

& Segev, 1989). The word *preview* is used to acknowledge the fact that insight gained from simulations must be confirmed in neurobiological experiments. However, less technically challenging experiments may suffice for confirmation.

The likelihood that the connectivity achieved after learning in a neural network simulation will be recognizable within the true cortical architecture giving rise to the biologically observed behavior is dependent on the accuracy to which simulated neurons reproduce their biological counterparts. For example, a single monosynaptic connection to a regular-spiking pyramidal cell is all that is required to convert a presynaptic tonic discharge into a postsynaptic phasic-tonic discharge (Schwindt, Spain, & Crill, 1992); however, a complicated network architecture is required by a simulated neural network using rate-model neurons that do not exhibit the frequency adaptation of regular-spiking pyramidal cells. The obvious solution is to implement frequency adaptation in rate-model neurons, but the consequence is that supervised learning using backpropagation or other mathematical variations of gradient descent cannot be used straightfowardly because the standard form of these learning algorithms does not take into account the activation history of a neuron that gives rise to the frequency adaptation. The problem is that the derivative of the activation function depends on the activation history of a neuron with frequency adaptation.

A straightforward solution to this problem was sought using a computational approach that does not require defining the derivative of the activation function in order to reproduce the mathematical implementation of gradient descent. The distinction between the mathematical and computational approaches is that the mathematical approach uses calculus to derive the gradient, while the computational approach evaluates the change in neuronal activity following a change in connection weight. Theoretically, the derivative of the activation function is best approximated as the tested change in connection weight approaches zero (i.e., $\lim \Delta w_{ij} \rightarrow 0$ where Δw_{ij} is the change in connection weight between neurons i and j that is used to measure an effect on neuronal activity). However, our initial attempts using minute changes in connection weight achieved minimal learning. It became obvious that more work was required before we could use a computational approach to reproduce gradient descent learning. Therefore, there were two objectives in this study. The first was to develop a simple generalized computational gradient descent approach that reproduces the classical mathematical approach described by Williams and Zipser (1989). Note that the purpose was to reproduce and not to validate the gradient descent approach since this approach has been extensively studied over the years (Hertz, Krogh, & Palmer, 1991). The second objective was to demonstrate the ability of this computational approach to achieve supervised learning in a network composed of rate-model neurons with frequency adaptation.

2 Methods

The rate-model neuron with frequency adaptation will be described before the computational approach of gradient descent learning so that we can highlight the features of the rate-model neuron that prohibit the use of mathematical gradient descent. Among all the possible ways to model frequency adaptation, the rate-model neuron of Cartling (1995, 1996) was chosen because it is directly based on the biology of neurons, which use calcium-sensitive potassium (K_{Ca}) channels to provide frequency adaptation (Schwindt et al., 1992, 1988). This frequency adaptation is due to calcium entry during membrane depolarization and a progressive increase in intracellular calcium during repetitive firing, which leads to calcium binding to K_{Ca} channels that open to hyperpolarize the membrane and gradually reduce the firing rate. Cartling (1995) derived a rate-model neuron that reproduced the frequency adaptation of a Hodgkin-Huxley formalism incorporating K_{Ca} channels. The level of frequency adaptation in this model can be varied over a wide range, from zero to maximum frequency adaptation, such that the same current step applied to a neuron could evoke discharges ranging from tonic to phasic-tonic profiles. This reflects the wide range of discharge profiles observed during single unit recordings in behaving animals (Fortier, Smith, & Kalaska, 1993; Fetz, Cheney, Mewes, & Palmer, 1989).

These adapting rate-model neurons were incorporated into a fully recurrent neural network. The influence of calcium on neuron firing was implemented in the otherwise standard activation function of neurons (Williams & Zipser, 1989). Neuronal activity was calculated by taking the sum of all neuronal inputs (see equation 2.1) and then applying a squashing function to limit values between -1 and $+1$ (see equation 2.2). In the present case, these values represented an input current of -1 to $+1\eta$A to the postsynaptic neuron. Such a range of input currents produced, according to the equations of Cartling (1996), neuronal activities ranging from $0 - 224$ Hz (see equation 2.3). These activities were then divided by the maximum firing rate of 224 Hz (see equation 2.4) in order to linearly rescale the activities between $0 - 1$. Thus, the inputs to a neuron represented an input current that produced firing rates scaled between 0 and 1 according to the following equations:

$$s_i(t) = \sum_j w_{ij} y_j(t) \tag{2.1}$$

$$i_i(t) = \frac{e^{s_i(t)} - e^{-s_i(t)}}{e^{s_i(t)} + e^{-s_i(t)}} \tag{2.2}$$

$$g_i(t) = \begin{cases} \phi(i_i(t) - v_i c_i(t) - \epsilon)^{\rho} & \text{if } i_i(t) - v_i c_i(t) - \epsilon \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

$$y_i(t+1) = g_i(t)/\omega, \tag{2.4}$$

where $s_i(t)$ is the net input; $y_i(t)$ is the neuron activity scaled between 0 and 1; $i_i(t)$ is the input current; $g_i(t)$ is the neuron activity in Hz; $c_i(t)$ is the intracellular calcium concentration; w_{ij} is the synaptic weight ranging between positive and negative values for excitation and inhibition; v_i is the sensitivity of firing to calcium, which ranges between 0 and 1; and the others are constants with values $\omega = 224$ Hz, $\phi = 254.7$, $\epsilon = 0.12$, and $\rho = 1$. The indexes are defined as $j \in$ presynaptic neurons (includes input, hidden, and output neurons), $i \in$ postsynaptic neurons (includes hidden and output neurons), and t is the time steps in the activity of a neuron. This implementation does not include a time constant of 1 ms (Cartling, 1996) for the change in neuron discharge frequency because each time step was 10 ms.

The calcium concentration is calculated according to the differential equation derived by Cartling (1996): $\mathrm{d}c_i(t)/\mathrm{d}t = q(c_i(t))g_i(t)/1000 - c_i(t)/\tau_c$ where τ_c is the calcium time constant equal to 111 ms and $q(c_i(t))$ is the increase in intracellular calcium during neuron activity calculated as $q(c_i(t)) = 0.11/(0.9 + c_i(t))$.

Supervised learning in a fully recurrent network of rate-model neurons can be achieved using the gradient descent approach of Williams and Zipser (1989). This gradient descent procedure requires changing the synaptic weights along the negative of the gradient of the network error function,

$$\Delta w_{ij}(t) = -\alpha \frac{\partial J(t)}{\partial w_{ij}}, \tag{2.5}$$

where w is a connection weight; $j \in$ presynaptic neurons, which includes input, hidden, and output neurons; $i \in$ postsynaptic neurons, which includes hidden and output neurons; t is a time step in the activity of a neuron; α is the learning rate; and J is the network error function, which is defined as the sum of differences-squared between the target and actual activities. For a neuron whose net input is $s_i(t) = \sum_j w_{ij} y_j(t)$ and whose activation is $y_i(t+1) = \frac{1}{1+e^{-s_i(t)}}$, Williams and Zipser (1989) derived the gradient of the error function as

$$-\frac{\partial J(t)}{\partial w_{ij}} = \sum_k e_k(t) \frac{\partial y_k(t)}{\partial w_{ij}}, \tag{2.6}$$

where $k \in$ postsynaptic neurons (hidden and output units) and e is the difference between the target ($d_k(t)$) and actual ($y_k(t)$) activity (hidden units have no target activity so their error would be 0). This equation essentially determines how the activity of a target neuron is affected by a change in weight anywhere in the network (i.e., it may be a weight to the target neuron or to another neuron that could ultimately have a polysynaptic influence on the target neuron activity). Mathematical calculation of the gradient requires knowledge of the derivative of the activation function. For rate-model neurons with frequency adaptation, the derivative of the activation function ($\partial y_k(t)/\partial w_{ij}$) constantly changes with the calcium concentration. Moreover, the calcium concentration depends on the neuronal activation history since it changes slowly with a time constant of 111 ms. Of all the possible methods to manage the complexity of finding the derivative of the activation function for adapting rate-model neurons, we chose to compute the slope of the activation function and use this value as an estimate of the derivative. For this simple computational approach, a connection weight (w_{ij}) was increased by a fixed step size (Δw_{ij}), and the change in activities ($\Delta y_k(t)$) was calculated. This was repeated for all connection weights. The weights remained fixed throughout the trajectory and then were updated in parallel from the following estimate of the negative gradient of the error function:

$$\sum_k e_k(t)\frac{\partial y_k(t)}{\partial w_{ij}} = \sum_k e_k(t)\frac{\Delta y_k(t)}{\Delta w_{ij}}. \tag{2.7}$$

This computational gradient descent is identical to the mathematical approach of Williams and Zipser (1989) except that the derivative ($\partial y_k(t)/\partial w_{ij}$) is estimated by computing the slope ($\Delta y_k(t)/\Delta w_{ij}$) for a given change in connection weight (Δw_{ij}).

Two techniques were used to optimize mathematical gradient descent (Hertz et al., 1991) and significantly improve learning. The first technique was weight initialization (Nguyen & Widrow, 1990) where the weights of synaptic inputs to a neuron are set randomly and then each weight is divided by the norm of these random inputs, and then all the connection weights are adjusted linearly so that the neurons will not saturate during the initial propagation of activities. Learning time with the mathematical and computational approaches was typically reduced by a factor of 3.0 to 3.5 using this technique. The second optimization technique used was to add a momentum term to equation 2.5,

$$\Delta w_{ij}(t) = -\alpha\frac{\partial J(t)}{\partial w_{ij}} + \beta\ \Delta w_{ij}(t-1), \tag{2.8}$$

where β is $0 - 1$. Several values of momentum were tested, but the optimal value was consistently about 0.5 for all simulations in this study.

3 Results

This section begins by showing how the computational approach of gradient descent can be used to reproduce the results of the mathematical approach defined by Williams and Zipser (1989) when both approaches use standard (nonadapting) rate-model neurons. This is followed by showing how our computational approach can be used for supervised learning in a network with rate-model neurons that exhibit frequency adaptation (Cartling, 1996).

3.1 Reproducing Mathematical Gradient Descent. The task that the network was trained on involved mapping from a tonic input discharge pattern on a single input unit to a phasic-tonic output pattern on a single output unit. The mathematical approach of Williams and Zipser (1989) uses rate-model neurons, without frequency adaptation, whose activation is set by the logistic squashing function $(1/(1+e^{-s_k(t)}))$ since its derivative is known. These rate-model neurons were used to create a network formed of one bias, one input, three hidden, and one output neurons. Both the mathematical and computational approaches of gradient descent were required to transform a tonic input discharge pattern into a phasic-tonic output pattern. Ten samples of training data were used. Thus, the conditions for both mathematical and computational gradient descent were identical.

Our computational gradient descent approach was designed to reproduce the mathematical approach of Williams and Zipser (1989) except that the derivative of the activation function $(\partial y_k(t)/\partial w_{ij})$ was estimated by using a finite difference approximation to its slope in the interval $[w_{ij}, w_{ij} + \Delta w_{ij}]$, resulting in $\Delta y_k(t)/\Delta w_{ij}$. The optimal step size (Δw_{ij}) had to be determined empirically. The results describe the selection of an optimal step size that was defined as that which yielded results reproducing mathematical gradient descent (Williams and Zipser, 1989). Figure 1 shows learning using mathematical calculation of the gradient with $\alpha = 0.04$ and $\beta = 0.5$. Learning was stopped after 1298 cycles when the error (measured as the sum of the differences-squared between the target and actual activities) dropped from an initial value of 3.14 to a value below 0.03. This served as the point of reference for comparison with our computational gradient descent approach.

We first sought an optimal step size yielding the least absolute difference between the mathematical and computational gradient trajectories. Step sizes within $\Delta w_{ij} = 10^{-15} - 10$ were tested at different levels of network error obtained through successive learning cycles. There was a sigmoidal relationship such that the optimal step size decreased together with network error. We also examined gradient ratios calculated by dividing the amplitude of the mathematical gradient trajectory by the corresponding amplitude of a computational gradient trajectory obtained using the optimal step size. These gradient ratios also formed a sigmoidal relationship, which decreased along with network error. At the initial error of the network (3.14), the optimal step size was 1.5. Figure 2 shows an example of computational gradient descent using this step size. The learning rate

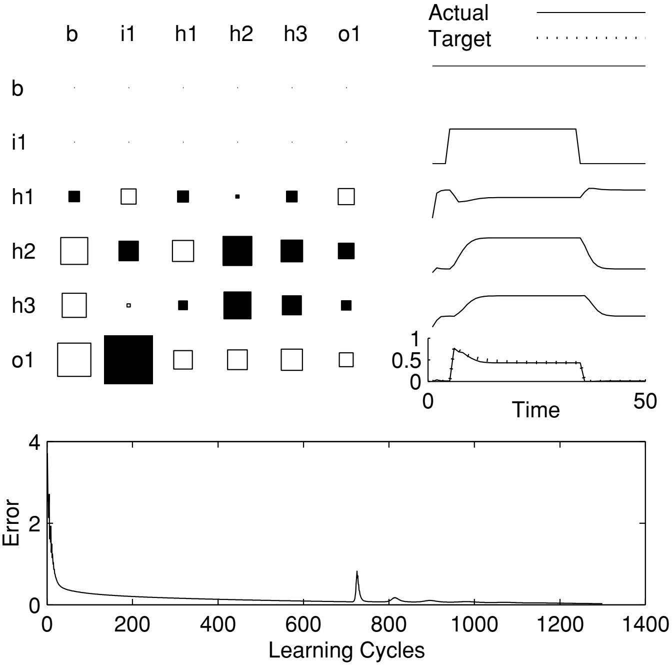

Figure 1: Learning of the transformation of a tonic input into a phasic-tonic output pattern using mathematical gradient descent (Williams & Zipser, 1989). Learning was stopped after 1298 learning cycles ($\alpha = 0.04$ and $\beta = 0.5$) when error fell below 0.03 and the actual output pattern (o1 solid line) closely matched the target pattern (o1 dotted line). The area of the blocks reflects strength of connection (black is excitatory and white is inhibitory) between the presynaptic neuron (column) and the postsynaptic neuron (row). b = bias, i1 = input unit, h1-3 = hidden units, o1 = output unit.

($\alpha = 0.16$) was selected as four times that used in the mathematical gradient descent ($\alpha = 0.04$ in Figure 1) because the amplitude of the gradient trajectory obtained using a step size of 1.5 was one-fourth that obtained using mathematical gradient descent.

Although the network error was similar in Figures 1 and 2, the connection weights were slightly different and, consequently, so were the neuron activities.

We sought to improve learning by using a sigmoidal step size and learning rate; however, this did not improve on learning with a fixed step size of 1.5. The sigmoidal drop in both optimal step size and gradient ratios suggested that a single optimal step size could be defined better based on the least variance of gradient ratios throughout the trajectory. Recalculating

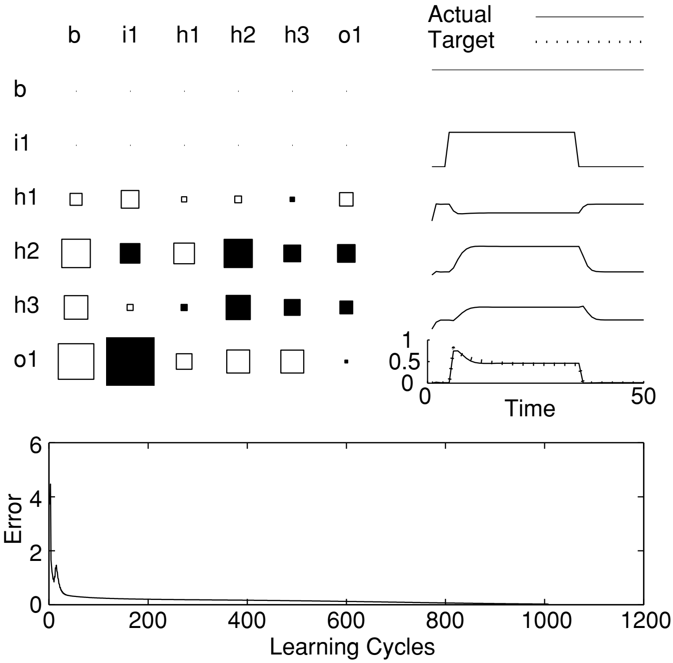

Figure 2: Same as Figure 1 except that error was reduced below 0.03 after 1007 learning cycles of computational gradient descent with $\alpha = 0.16$, $\beta = 0.5$, and step size $= 1.5$.

the optimal step size on this basis yielded consistently small optimal step sizes (1.26E-3 ± 1.62E-3) at all levels of network error. These were associated with gradient ratios of 3.59 ± 1.36. Thus, a single small step size, based on least variance of gradient ratios, could yield gradients that were consistently proportional to those obtained mathematically at any level of error. Multiplying these computationally derived gradients by a constant could then provide gradients that reproduced the mathematically derived ones. A step size of 1.26E-3 with a learning rate of 0.16 yielded the same network as that using mathematical gradient descent (visual inspection of the network could not reveal any difference from the reference case; see Figure 1). After 45,000 learning cycles, both mathematical gradient descent ($\alpha = 0.04$ and $\beta = 0.5$) and computational gradient descent ($\alpha = 0.16$, $\beta = 0.5$, and step size = 1.26E-3) reduced the error below 1.5E-4.

In theory, the slope of the activation function between w_{ij} and $w_{ij} + \Delta w_{ij}$ should approach the tangent of the activation function at w_{ij} as $\lim_{\Delta w_{ij} \to 0}$. In practice, however, this was not the case. This was likely related to two factors specific to our computational approach. First, the resolution of 32-bit floating-point processors limits the smallest step size to 1E-15 in comparison to the infinitesimal virtual step of the mathematical approach. Second, the effects of a finite step (more than 1E-15) may be so small that it falls below the floating-point resolution and consequently fails to propagate through the network. The smallest perceptible step capable of propagating network activity was found to have a sigmoidal relationship with network error. The smallest step was 2.2E-7 ± 4.2E-7 when error was below an apparent transition of about 0.5 and it was 2.3E-03 ± 3.3E-03 when above this transition. Smaller step sizes were incapable of faithfully propagating activity through the network and consequently incapable of providing an accurate estimate of the gradient.

3.2 Computational Gradient Descent with Adapting Rate-Model Neurons. The previous results were from networks with standard rate-model neurons (without frequency adaptation) in order to reproduce mathematical gradient descent using our computational approach. We now show the behavior of rate-model neurons with frequency adaptation and then how a network of such adapting neurons can undergo supervised learning with our computational approach to gradient descent.

As explained in section 2, frequency adaptation of rate-model neurons was implemented according to Cartling (1996). Neuronal discharge causes calcium entry leading to stimulation of K_{Ca} channels and the expression of frequency adaptation. The adapting rate-model neuron equation (see equation 2.3) allows setting a calcium sensitivity that reproduces the effects of K_{Ca} channel density on frequency adaptation. The discharge properties of a neuron with different levels of calcium sensitivity and current inputs are shown in Figure 3. In Figure 3A, the neuron received a fixed step input of 1.0 ηA, but its calcium sensitivity was varied linearly between 0 and 1.0. At a calcium sensitivity of 0, the discharge rate was maximal (224 Hz) and followed the step profile of the input current. This yielded the highest level of intracellular calcium because it is directly related to neuronal firing rate. As the calcium sensitivity of the neuron was increased by steps of 0.2, there was a gradual drop of the initial firing rate and the peak calcium concentration. This yielded a phasic-tonic discharge profile that was most pronounced for the neuron with maximal sensitivity to calcium. In Figure 3B, the neuron had a fixed maximal calcium sensitivity of 1.0, but its input consisted of a first step that varied between 0.2 and 1.0 ηA and a second step always to 0.8 ηA. As the first step was increased from 0.2 to 1.0 ηA, there was an increase in both the initial firing rate and intracellular calcium concentration, which subsequently caused more pronounced attenuation of the firing rates and clearer phasic-tonic discharges. Although the second step was always to the same amplitude, the firing rates were inversely related to the prior activities. These results indicate that the neuron activity depends on the prior activation history and yields a range of firing profiles, from purely

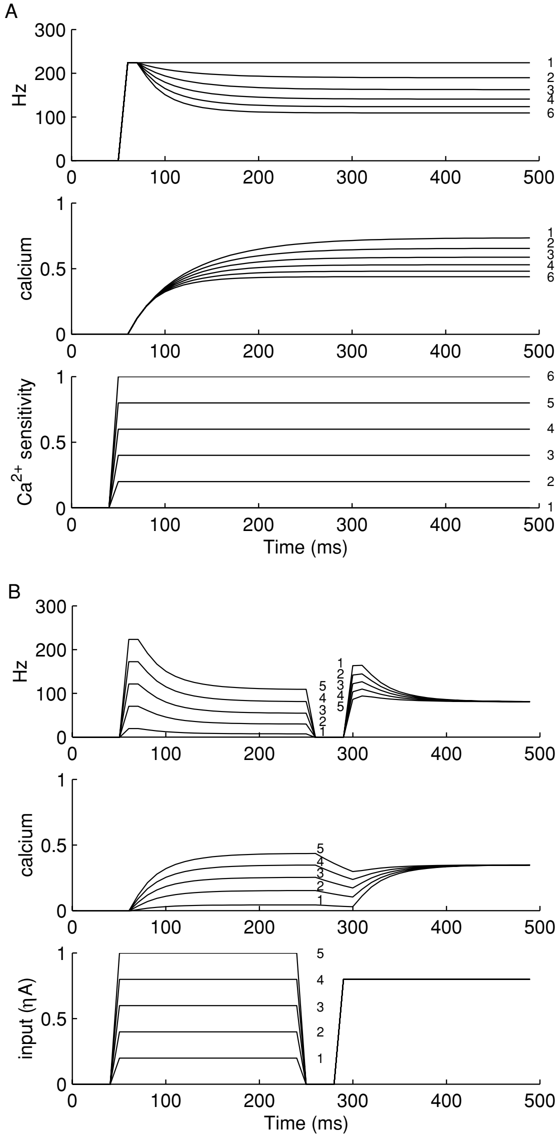

Figure 3: Responses of rate-model neurons (Cartling, 1996) with intrinsic properties producing frequency adaptation. (A) Frequency response and intracellular calcium levels in response to a fixed 1 ηA input current step to a neuron at different levels of calcium sensitivity. (B) Frequency response and intracellular calcium levels of a neuron with fixed calcium sensitivity of 1.0 in response to a first step to different step sizes of input current and a second step always to 0.8 ηA. Waveforms resulting from the same conditions are labeled at the right with the same number.

tonic to strongly phasic-tonic, depending on the size of the input current and the sensitivity to calcium.

Our computational gradient descent approach was used to produce the same transformation as in Figure 2 but with adapting rate-model neurons whose calcium sensitivity was held at zero for comparison. It remained to be determined whether the optimal weight change parameters of $\alpha = 0.16$, $\beta = 0.5$, and step size of 0.001 used with nonadapting neurons would apply to adapting neurons. Optimal weight learning with adapting neurons required lower α values and a narrower range of step sizes, but it included the step size of 0.001 used earlier for nonadapting neurons. The network obtained by using $\alpha = 0.001$, $\beta = 0.5$, and step size of 0.001 for weight changes, while calcium sensitivity was held at zero, is shown in Figure 4. As described in section 2, the activities displayed between 0 and 1 represent a linear rescaling of firing rates between 0 and the maximal firing rate of 224 Hz. Figure 4 shows that the network quickly converged (error less than 1.26E-3) using computational gradient descent. It was not surprising to see that the resulting weight matrix was very different from that in Figure 2 since neurons with different properties were used for the networks in these two figures.

The next step was to use the computational gradient descent approach to modify not only connection weights but also the calcium sensitivity of neurons. The learning procedure was identical: weights ($w_{ij}(t)$) were changed in proportion to $\sum_k e_k(t)\frac{\Delta y_k(t)}{\Delta w_{ij}}$, while calcium sensitivities ($v_i(t)$) were changed in proportion to $\sum_k e_k(t)\frac{\Delta y_k(t)}{\Delta v_i}$. Since the calcium sensitivities are limited to values within 0 to 1, it was expected that smaller step sizes would be optimal. This was observed, but there were minimal benefits from using smaller step sizes. The values $\alpha = 0.001$, $\beta = 0.5$, and step size of 0.001 were selected for changing both the connection weights and calcium sensitivities in order to produce the network results shown in Figure 5. Fig. 5A shows that the network converged (error less than 2.79E-4) to a solution that largely involved a single excitatory connection from the input to the output neuron with a calcium sensitivity (v_i) of 0.64. Neuron h1 was inactive (its connection to o1 had no impact), neuron h2 had negligible activity, and neuron h3 made a small excitatory connection to the output neuron. Figure 5B shows that eliminating the connections from h2 and h3 to the output neuron did not ruin the match between actual and target output activities (error less than 2.76E-2). This indicates that the transformation of the tonic input into a phasic-tonic output was largely achieved by the frequency adaptation of the output neuron. These results show that our computational gradient descent approach can correctly modify both weights and calcium sensitivity in adapting rate-model neurons in such a way as to achieve supervised learning and produce connectivity consistent with real biological networks where a tonic input is converted into a phasic-tonic ouput by the intrinsic properties of a single cell (Schwindt et al., 1992).

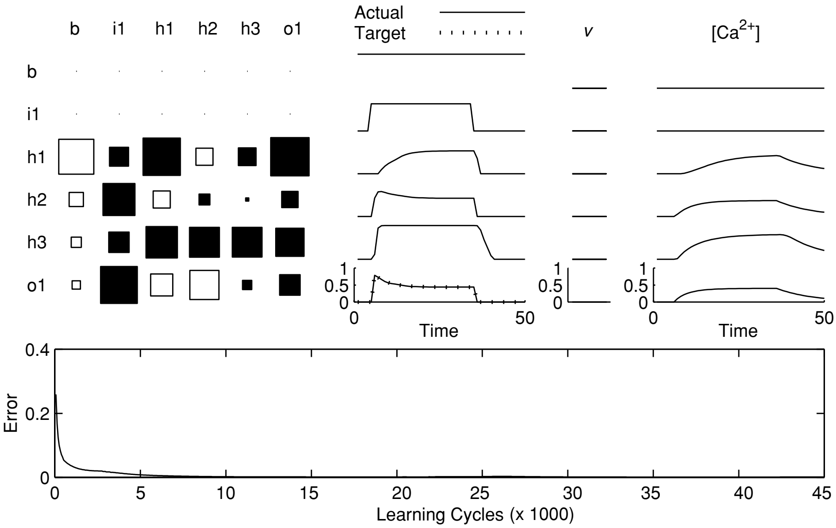

Figure 4: Learning of the transformation of a tonic input into a phasic-tonic output pattern using computational gradient descent ($\alpha = 0.001$, $\beta = 0.5$, and step size of 0.001) in a network formed of adapting rate-model neurons with intrinsic properties that are capable of producing frequency adaptation. The bottom panel describes the exponential decline in error to less than 1.26E-3 with successive learning cycles. The area of the blocks reflects strength of connection (black is excitatory and white is inhibitory) between the presynaptic neuron (column) and the postsynaptic neuron (row) after learning: b = bias, i1 = input unit, h1–3 = hidden units, o1 = output unit. The actual firing pattern of each neuron after learning is displayed. The dotted line for the target output activity is completely overlapped by the actual activity of the neuron. The right-most column describes the calcium concentration of the neurons. The column labeled v contains histogram bars of size 0 for the calcium sensitivity. The calcium sensitivity was set to zero for comparison with the network in Figure 2 containing rate-model neurons without frequency adaptation.

4 Discussion

The results showed that mathematical gradient descent could be reproduced using a simple computational approach that empirically determines the change in network error for a given step change in connection weight. It was shown that selection of an appropriate step size was key to reproduction of mathematical gradient descent. Theoretically, the results of mathematical gradient descent should be approached by computational gradient descent as smaller weight steps are used. However, our results showed that numerical resolution formed a lower limit to usable step sizes such that smaller

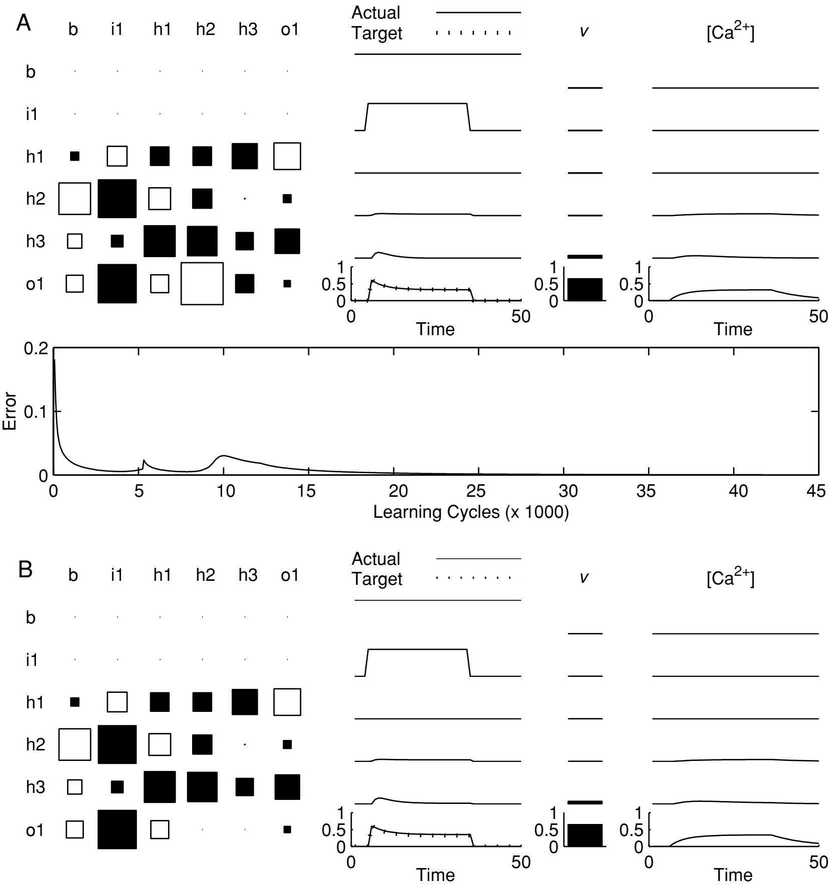

Figure 5: Learning of connection weights and calcium sensitivities in a network with rate-model neurons exhibiting frequency adaptation. (A) The layout is the same as in Figure 4 and the neural network is the same except that learning of calcium sensitivity was enabled. The values $\alpha = 0.001$, $\beta = 0.5$, and step size of 0.001 were used for changing both the connection weights and calcium sensitivities (error less than 2.79E-4). (B) Copy of the network in A except for the removal of the connections from h2 and h3 to o1 in order to show that the transformation of input activity was largely due to its connection with the output neuron exhibiting frequency adaptation (error $<$ 2.76E-2).

steps were ineffective in changing network activity and, consequently, ineffective in providing information about the gradient. This is unlike the mathematical approach, which always provides information about the gradient.

Large step sizes always propagate some activity through the network, but the local details of the error function can be detected only by small step sizes. The smallest step size that could be used to reveal the gradient was directly related to network error such that individual neurons became more sensitive to weight changes as learning occurred. However, the smallest usable step size was not always optimal. It appears that by not being able to take an infinitesimal step size, the smallest optimal step size (defined as that which reproduced the results of the mathematical approach) was then arbitrarily determined by the step on the error function that produced a gradient that most closely reproduced the mathematically derived gradient.

The amplitude of the gradient trajectory was smaller when estimated by the computational approach. This reflects underestimation of a tangent by measurement of the slope on an exponentially increasing or decreasing function. This underestimation of the gradient was offset by using higher learning rates (α). These higher learning rates were no longer appropriate when we switched from nonadapting to adapting rate-model neurons. It was not because the actual gradient was better estimated but rather because the adapting neurons were more sensitive to step changes in weights and consequently yielded larger gradients. Smaller learning rates had to be used; otherwise, learning was very erratic. On the other hand, the optimal step size and momentum identified for nonadapting neurons were appropriate for the adapting neurons.

The study showed that our computational gradient descent approach could be used not only to change connection weights but also the sensitivity of frequency adaptation to calcium. The parameters used for weight changes were also applied to calcium sensitivity changes. Network learning converged onto the appropriate weight connections and neuron intrinsic properties that could transform a tonic input pattern into a phasic-tonic output pattern. This transformation was largely achieved by a monosynaptic connection from the tonic input neuron to the output neuron that exhibited strong frequency adaptation, as is the case for real biological neurons. For example, a single monosynaptic connection to a regular-spiking pyramidal cell is all that is required to convert a tonic input current into a phasic-tonic discharge (Schwindt et al., 1992). This is a simple yet fundamental transformation achieved by intrinsic neuron properties rather than connectivity of the network. Thus, it becomes easier to recognize known biological circuitry in neural network simulations when the model neurons express more features of real biological neurons. Consequently, it is more likely that network properties suggested from simulations could be confirmed from neuron properties and connectivity observed in real biological networks.

Our computational approach to supervised learning is both simple to implement and generalizable to any conceivable model neuron with modifiable intrinsic properties. The key learning parameters are α and step size, which must be determined empirically for each problem in order to achieve optimal performance. Moreover, future simulations could not only adjust connection weights and modifiable intrinsic properties but also examine the effects of using different learning rates for the changes in connection weights and the changes in modifiable intrinsic properties. We showed the architecture of the network when learning of weight and calcium sensitivity was identical, but other results could be obtained when learning rates differ. Biological neurons certainly undergo different rates of changes in synaptic potentiation (Dittman, Kreitzer, & Regeh, 2000; Salin, Malenka, & Nicoll, 1996).

References

Arbib, M. A., & Erdi, P. (2000). Precis of neural organization: Structure, function, and dynamics. *Behav. Brain Sci.*, *23*, 513–571.

Barish, M. E. (1988). Ion channels as a source of behavioral diversity: Doing more with less in simpler organisms. *Trends. Neurosci.*, *11*, 558–561.

Cartling, B. (1995). A generalized neuronal activation function derived from ion-channel characteristics. *Network*, *6*, 389–401.

Cartling, B. (1996). Response characteristics of a low-dimensional model neuron. *Neural Comput.*, *8*, 1643–1652.

deCharms, R. C., & Zador, A. (2000). Neural representation and the cortical code. *Annu. Rev. Neurosci.*, *23*, 613–647.

Dittman, J. S., Kreitzer, A. C., & Regehr, W. G. (2000). Interplay between facilitation, depression, and residual calcium at three presynaptic terminals. *J. Neurosci.*, *20*, 1374–1385.

Fetz, E. E., Cheney, P. D., Mewes, K., & Palmer, S. (1989). Control of forelimb muscle activity 21 by populations of corticomotoneuronal and rubromotoneuronal cells. *Prog. Brain Res.*, *80*, 437–449.

Finlay, B. L., & Darlington, R. B. (1995). Linked regularities in the development and evolution of mammalian brains. *Science*, *268*, 1578–1584.

Fortier, P. A., Smith, A. M., & Kalaska, J. F. (1993). Comparison of cerebellar and motor cortex activity during reaching: Directional tuning and response variability. *J. Neurophysiol.*, *69*, 1136–1149.

Harris-Warrick, R. M. (2000). Ion channels and receptors: Molecular targets for behavioral evolution. *J. Comp. Physiol.*, *186*, 605–616.

Hertz, J., Krogh, A., & Palmer, R. G. (1991). *Introduction to the theory of neural computation*. Reading, MA: Addison-Wesley.

Jones, E. G. (2000). Microcolumns in the cerebral cortex. *Proc. Natl. Acad. Sci.*, *97*, 5019–5021.

Katz, P. S., & Harris-Warrick, R. M. (1999). The evolution of neuronal circuits underlying species-specific behavior. *Curr. Opin. Neurobiol.*, *9*, 628–633.

Koch, C., & Segev, I. (1989). Methods in neuronal modeling: From synapses to networks. Cambridge, MA: MIT Press.

Nguyen, D., & Widrow, B. (1990). Improving the learning speed of 2-layer neural networks 22 by choosing initial values of the adaptive weights. *International Joint Conference of Neural Networks*, *3*, 21–26.

Ranganathan, R. (1994). Evolutionary origins of ion channels. *Proc. Natl. Acad. Sci.*, *91*, 3484–3486.

Salin, P. A., Malenka, R. C., & Nicoll, R. A. (1996). Cyclic AMP mediates a presynaptic form of LTP at cerebellar parallel fiber synapses. *Neuron*, *16*, 797–803.

Schwindt, P. C., Spain, W. J., & Crill, W. E. (1992). Calcium-dependent potassium currents in neurons from cat sensorimotor cortex. *J. Neurophysiol.*, *67*, 216–226.

Schwindt, P. C., Spain, W. J., Foehring, R. C., Stafstrom, C. E., Chubb, M. C., & Crill, W. E. (1988). Multiple potassium conductances and their functions in neurons from cat sensorimotor cortex in vitro. *J. Neurophysiol.*, *59*, 424–449.

Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural. Comput.*, *1*, 270–280.

Received July 19, 2004; accepted February 24, 2005.