LETTER Communicated by Christian Liebold

Adaptive Synchronization of Activities in a Recurrent Network

Thomas Voegtlin
voegtlin@loria.fr
INRIA—Campus Scientifique, F-54506 Vandoeuvre-les-Nancy Cedex, France

Predictive learning rules, where synaptic changes are driven by the difference between a random input and its reconstruction derived from internal variables, have proven to be very stable and efficient. However, it is not clear how such learning rules could take place in biological synapses. Here we propose an implementation that exploits the synchronization of neural activities within a recurrent network. In this framework, the asymmetric shape of spike-timing-dependent plasticity (STDP) can be interpreted as a self-stabilizing mechanism. Our results suggest a novel hypothesis concerning the computational role of neural synchrony and oscillations.

1 Introduction

Synaptic plasticity that depends on the time difference between pre- and postsynaptic action potentials has been observed in a variety of neuronal types (Markram, Lübke, Frotscher, & Sakmann, 1997; Bi & Poo, 1998). In general, a causal firing order (the presynaptic neuron fires before the postsynaptic neuron) induces long-term potentiation (LTP) of the synapse, while an anticausal order induces long-term depression (LTD). However, anticausal STDP has been observed in some inhibitory interneurons (Tzounopoulos, Kim, Oertel, & Trussell, 2004). Although the mechanism of spike-timing-dependent plasticity (STDP) is not fully understood, it is believed that action potentials backpropagated from the soma to the dendrite play a major role in propagating information about spiking to the synaptic site (Stuart & Sakmann, 2002; Shouval, Bear, & Cooper, 2002).

Following these observations, STDP has been proposed as a possible mechanism for the development of sensory and motor representations (Kempter, Gerstner, & van Hemmen, 1999; Song, Miller, & Abbott, 2000). In general, these models learn by correlation, following Hebbian organization principles. However, the stability of the learning process has remained a problem (Abbott & Nelson, 2000; Gütig, Aharonov, Rotter, & Sompolinsky, 2003). Indeed, it remains to be shown how STDP could yield a rich repertoire of receptive fields that can be combined in a distributed representation. The emergence of a rich repertoire of synaptic weight patterns requires both competition between the synapses and stability

of the learning process. However, competition tends to make learning unstable and results in synaptic weights that have a bimodal distribution, with peaks located on the extreme values of the allowed interval (Kempter et al., 1999; Cateau & Fukai, 2003; Legenstein, Naeger, & Maass, 2005).

A very efficient method to obtain stable learning algorithms is to build negative feedback into the learning rule. In this case, the activity produced by the network is compared to a desired pattern of activity, and the sign of synaptic modifications depends on the difference, or error, between the expected pattern and the actual pattern of activity. This type of feedback is present in well-known supervised learning rules (Widrow & Hoff, 1960), where a desired output is provided to the network. Negative feedback is also used in a broad class of unsupervised learning rules that we will call *predictive* learning rules (Oja, 1989; Olshausen & Field, 1996; Rao & Ballard, 1997; Rao, 1999). Predictive learning can be expressed in the probabilistic framework of generative models. In this framework, a random input vector plays the role of the desired pattern, and it is compared to a reconstruction, or prediction, that is generated by the network from an internal representation. In contrast to standard Hebbian rules, predictive rules are self-stabilizing, and they naturally induce a competition between synapses. Stability is ensured because synaptic modifications depend on a prediction error, which can be positive or negative. Competition is ensured because the error minimization implicitly renormalizes the vector of synaptic weights afferent to a neuron.

This raises the question of the biological relevance of predictive rules. If we assume that information in natural neurons is encoded in the timing of spikes, then it is natural to interpret STDP as a comparison between spike times. In this context, Rao and Sejnowski (2001) formulated STDP as a form of temporal difference learning; they demonstrated that it is possible to train a neuron to predictively cancel its inputs, using feedback and inhibition. Kitano and Fukai (2004) have shown that a model trained with STDP can account for predictive synchronous firing of premotor and motor neurons. However, these proposals are limited because they do not allow combining the activities of several neurons in a distributed representation. We have demonstrated a network of spiking neurons trained with a predictive learning rule that was able to learn and combine complex receptive fields in a distributed representation (Voegtlin, 2007). However, the learning rule used in that proposal is not plausible because it involves the spiking times of three neurons.

Indeed, a fundamental problem that arises with the biological implementation of predictive learning rules is that they need to combine three sources of information: a sensory input, an internal representation of that input, and a top-down expectation derived from the internal representation and compared to the input. In contrast, biological synapses are binary sites, where plasticity is believed to depend on the difference between pre- and postsynaptic spiking times. Thus, in order to be compatible with

experimental observations, a synaptic modification rule may depend on the algebraic difference between spike times, but it may not use these spiking times directly.

In this letter, we propose a solution to this problem. We show that it is possible to implement predictive learning using time coding and STDP. Our solution involves a feedback loop between excitatory and inhibitory cells, and it takes advantage of the synchronization of neural activities within a recurrent network. The proposed network learns in a way that is qualitatively similar to the Oja network (Oja, 1989); its neurons learn receptive fields that are orthogonal and span the subspace of the highest variance of a time-coded input vector.

In order to achieve this, we make the following hypotheses about neural coding and information processing:

- We use an implicit temporal code, defined by the dynamic response properties of neurons. In this code, the value encoded by a spike depends on the position of the postsynaptic neuron on its firing cycle when it receives the spike (Voegtlin, 2007).
- We interpret the synchronization of neural activities within a recurrent network as a predictive mechanism. In this interpretation, the firing phases of the neurons encode an arbitrary input. Synchronization occurs because the network actively cancels its phases by subtracting a prediction of the input through inhibitory synapses.
- We posit the existence of a credit assignment term in synaptic plasticity that gates STDP. This term depends on the potential of the postsynaptic neuron when a current passes through the synapse. Therefore, it uses information local to the synapse.
- We interpret the shape of asymmetric STDP as a self-stabilizing mechanism that works in combination with feedback. Within a feedback loop, asymmetric STDP creates a balance between LTP and LTD, which results in precise spike timing.

These hypotheses are explained in sections 2 to 7. The learning rules that we use are detailed in section 6, and our results are presented in section 8.

2 Time Coding Based on Neural Responses

In order to design rate-coded neural networks, one is relatively free to encode stimuli using arbitrary values. However, for spiking neurons, the choice of a time code is much less neutral because it can alter processing capabilities significantly; for example, scale-invariant recognition of stimuli can be obtained by logarithmically encoding the intensity values of a stimulus in time delays (Hopfield, 1995).

Another important aspect is the distributed nature of neural representation. In order to achieve interesting generalization capabilities, it is desirable to use distributed representations, where the activities of several

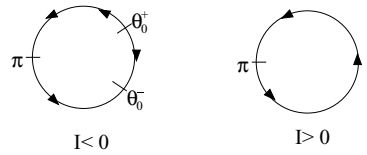

Figure 1: Phase circle of the theta model. The neuron fires every time θ crosses π. For $I > 0$, the neuron fires regularly. For $I < 0$, there are two fixed points: an unstable point $\theta_0^+ = arccos\frac{1+\tau I}{1-\tau I}$, and an attractor $\theta_0^- = -\theta_0^+$.

neurons can be combined in a meaningful way. Classically, in order to combine synaptic currents in a meaningful way, computational models of spiking neural networks have taken advantage of the shape of postsynaptic potentials (PSPs) (Maass, 1996; Bohte, Kok, & La Poutré, 2002; Gütig & Sompolinsky, 2006). In these models, PSPs are linearly combined in order to determine a neuron's firing time. In theory, universal function approximation is possible within this framework. However, a practical limitation of PSP-based models is that the available coding interval is limited by the length of the rising segment of the PSPs (Maass, 1996). This is because the shape of the PSPs, which is imposed by the synapses, is directly relevant for computations.

In this letter, we use a different type of time coding; it is based not on PSPs but on the dynamic response properties of nonlinear integrate-and-fire neurons. In these models, the effect of a synaptic current depends not only on the synaptic weights, but also on the time when this current is received. For neurons spiking regularly, this dependency is commonly described by the phase response curve (PRC) of the neuron (Gutkin, Ermentrout, & Reyes, 2005).

We use theta neurons, which are mathematically equivalent to quadratic integrate-and-fire neurons (Ermentrout, 1996; Gerstner & Kistler, 2002). The dynamics of the theta model are given by a first-order ordinary differential equation,

$$\frac{d\theta}{dt} = \frac{1-\cos\theta}{\tau} + (1+\cos\theta)I(t), \tag{2.1}$$

where θ is called the potential of the neuron and $I(t)$ is a variable input current, measured in radians per unit of time. For convenience, units of time are called "milliseconds." The neuron is said to spike every time θ crosses π. The dynamics of the theta neuron can be represented on a phase circle (see Figure 1).

In the theta model, the effect of an input current is not identical across the phase circle; currents that occur near the spike (for θ close to π) have little effect on θ, while currents that arrive when θ is close to zero have a

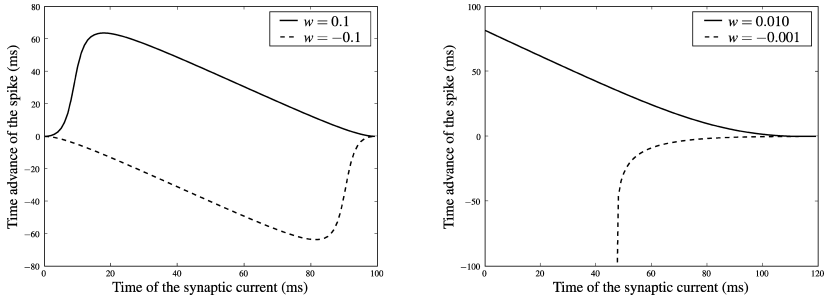

Figure 2: Response curves of the theta model for finite-size currents. The neuron receives a baseline current I_0 plus a Dirac impulsion of weight w at time t_D: $I(t) = I_0 + w\delta(t - t_D)$. Curves show how the firing time t_f of the neuron changes with t_D. (Left) Responses for $I_0 > 0$ ($I_0 = 0.001$, $\theta(0) = -\pi$). With $I_0 > 0$, the neuron fires regularly, and the positive curve is called the phase response curve (PRC). If w is infinitesimal, the curves corresponding to $w > 0$ and $w < 0$ are symmetric (not shown here). Here w is finite, and the curves are not symmetric; the ascending and descending phases have different slopes. (Right) Response for $I_0 < 0$. The initial condition is slightly above the unstable equilibrium point ($I_0 = -0.001$, $\theta(0) = \theta_0^+ + 0.0001$) so that the neuron fires if it is not perturbed. For $w > 0$, the response curve is approximately linear until it reaches zero. For $w < 0$, the current might cancel the spike if it occurs early.

maximal effect. Figure 2 shows the response curves of the theta model, for finite-size currents.

The PRC describes how a neuron transforms input spike times into output spike times. In the framework of time coding, the PRC can be seen as a transfer function. This allows us to define an implicit time code, where the analog value encoded by a spike time depends on the internal state of the neuron that receives it.

To explain this, let us consider the effect of a spike transmitted through a synapse. If the presynaptic neuron sends a spike when the postsynaptic neuron is in such a state that its PRC takes a high value, then that presynaptic spike will have a great effect on the postsynaptic spike time. Thus, this point in time should code for a high value. If the spike arrives when the PRC of the postsynaptic neuron is close to zero, then its arrival time should code for a small value. Hence, the value encoded by a spike time should not be chosen arbitrarily; it should depend on the internal state of the postsynaptic neuron. This code is implicit, in the sense that the values assigned to spike times are not imposed externally but result from internal neural dynamics.

PRC-based time coding was first introduced by Lengyel, Kwag, Paulsen, and Dayan (2006), and it has been further developed in Voegtlin (2007) and McKennoch, Voegtlin, and Bushnell (2009). An important advantage is that in these models, computations are not limited by the shape of PSPs;

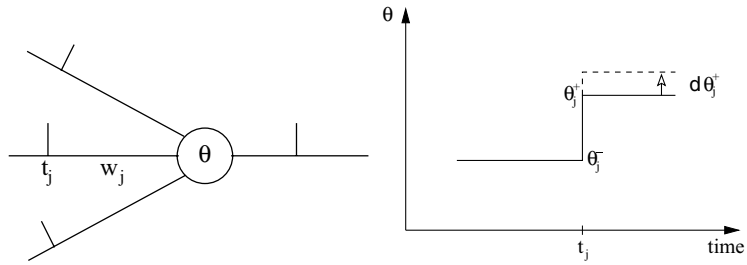

Figure 3: Notations used in the text. θ_j^- (resp. θ_j^+) denotes the postsynaptic potential before (resp. after) the presynaptic spike.

postsynaptic integration is performed over a timescale imposed by the internal dynamics of the neuron and goes well beyond the timescale of synaptic currents (McKennoch et al., 2009).

Our notation is as follows. In order to model synaptic interactions, the input current $I(t)$ is written as the sum of a constant baseline I_0 and transient synaptic currents $I_j(t)$:

$$I(t) = I_0 + \sum_{j=1}^{N} I_j(t). \tag{2.2}$$

In this notation we assume that the neuron receives one single spike on each synapse, and t_j denote the time of arrival of an action potential on synapse j (see Figure 3).

The hypothesis behind PRC-based models is that the response properties of neurons are more relevant to neural computation than the exact shape of synaptic currents. This comes from the observation that neural response properties are rich enough to sustain computations, with no need to involve the exact shape of synaptic currents (Voegtlin, 2007; McKennoch et al., 2009). Thus, we do not model synaptic dynamics. Instead, we use highly simplified synapses, where synaptic currents are modeled as Dirac functions:

$$I_j(t) = w_j \delta(t - t_j), \tag{2.3}$$

where w_j denotes the weight of the synapse j. A spike arriving on a synapse triggers an instantaneous change of the potential θ_j of the postsynaptic neuron. Let θ_j^- (resp. θ_j^+) respectively denote the potential before (resp. after) t_j. Integrating the Dirac current in equation 2.1 yields

$$\tan \frac{\theta_j^+}{2} = \tan \frac{\theta_j^-}{2} + w_j. \tag{2.4}$$

Note that $V = \tan \frac{\theta}{2}$ is the membrane potential used in the equation of the quadratic integrate-and-fire neuron (Ermentrout, 1996).

3 Adaptive Phase Cancellation

Börgers and Kopell (2003) have studied gamma oscillations in a recurrent network of excitatory and inhibitory theta neurons. In this network, the E-cells drive and synchronize the I-cells, while the I-cells in turn gate and synchronize the E-cells. This interaction between excitatory neurons and inhibitory interneurons has been proposed as the source of beta and gamma oscillations in a wide variety of structures, such as the insect antennal lobe (Martinez, 2005), cat visual cortex, rat enthorinal cortex (Cunningham, Davies, Buhl, Kopell, & Whittington, 2003), and hippocampus (Whittington, Traub, Kopell, Ermentrout, & Buhl, 2000).

Here we use the same architecture, with a few differences. The major novelty is that we use plastic synapses. In addition, we are less restrictive concerning the sign of synaptic weights. We consider two populations of neurons, X and Y, fully connected to each other by synapses that can be excitatory or inhibitory, depending on the sign of the synaptic weight. We do this for pure convenience, since our learning algorithm does not always result in synaptic weights that all have the same sign. However, modifications of our algorithm that use only E-cells and I-cells are easy to derive from this work.

Despite this difference, our model is faithful to the original network by Börgers and Koppell (2003), in the sense that the X cells drive the Y cells, and the Y cells in turn gate the X cells. The X cells have a positive baseline current, and they fire periodically even in the absence of feedback. The Y cells have a negative baseline current; they have no baseline activity, and they fire only when excited by the X cells.

Another difference lies in the synchronization mechanism. In Börgers and Kopell (2003), synchronization occurs because the E-cells receive inhibition at the beginning of their firing cycle, where inhibition has a synchronizing effect. In our model, the feedback sent by the Y cells is a mixture of excitation and inhibition, because synaptic weights can be of both signs. In addition, the X cells receive this feedback at the end of their firing cycle, where excitation tends to synchronize them and inhibition tends to desynchronize them. We initialize the synaptic weights with a balanced mixture of positive and negative values, so that the overall effect of feedback on synchrony is neutral. Thus, if the network is not trained, the Y cells do not synchronize the X cells.

Synchronization does occur, however, because our network learns. Our learning algorithm is designed so that the Y cells tend to synchronize the X cells. Once the network is trained, every X cell receives a different pattern of inhibition that depends on the current set of phases but also on the

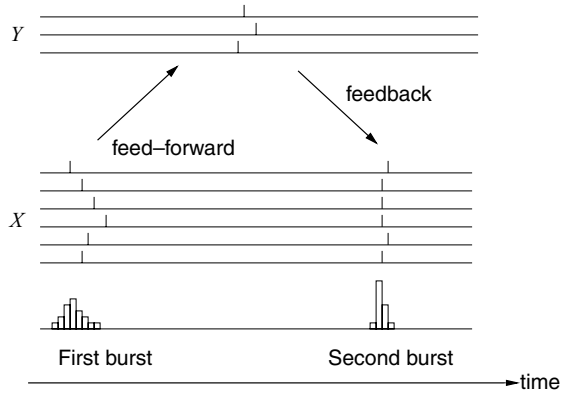

Figure 4: Predictive cancellation of spiking phases. An input vector is encoded in the firing phases of the X cells (first burst). The Y cells are activated by X cells through feedforward connections. A prediction of the initial phases is sent through feedback connections. The phases during the second burst are the difference between initial and predicted phases.

cell itself. Thus, synchronization results not from ensemble properties at the population level but from adaption of the inhibition pattern to each particular neuron.

In order to train the network, we consider trials of finite duration. A trial includes two bursts of the X cells and one burst of the Y cells. On each trial, an input vector is drawn from a statistical distribution and encoded in the relative spike times (phases) of the X cells. This constitutes the initial burst of X. Next, the X cells excite the Y cells, and the Y cells in turn gate the X cells. Thus, the second burst of X is modulated by the Y cells (see Figure 4).

On each trial, synaptic weights are slightly modified according to a spike-timing-dependent learning rule. The network is trained over many trials. Modifications of the weights are made so that the Y cells tend to synchronize the X cells. Synchronization is measured by the variance of the second burst of the X cells.

In the absence of feedback, the second burst of X would be identical to the first burst. Thus, in order to describe the feedback sent by the Y cells, it is convenient to consider the vector of phase modifications of the X cells, because it belongs in the same space as the time-coded input. This vector can be seen as a reconstruction of the input. The actual set of phases of the X cells during the second burst is the difference (error) between this reconstruction and the initial set of phases. Hence, this phase cancellation implements a comparison between an input and its reconstruction; a good reconstruction leads to synchronous activities during the second burst.

It should be noted that in general, the Y cells will not perfectly synchronize the X cells. The input is a statistical ensemble, and only the predictable

part of the phases can be cancelled. Thus, the vector of phases of X during the second burst represents the part of the input that is not predictable.

A priori, the comparison between a stimulus and its reconstruction could be performed more simply, and it is not clear why one would want to use oscillations for that (e.g., one could use direct inhibition of neural activities, as in Rao & Sejnowski, 2001). Here we show that the cancellation of phases during an oscillation provides a biologically plausible scenario for learning, because it can be achieved by a learning rule that depends on the time difference between pre- and postsynaptic spikes.

Let x_i denote the firing time of neuron i during the second burst of X (which encodes the reconstruction error), and let y_j denote the firing time of neuron j in Y (which encodes the internal representation). Consider the following error,

$$E = \sum_{i=1}^{N} \sum_{j=1}^{M} (x_i - y_j - D)^2, \tag{3.1}$$

where $D > 0$ is a fixed delay that controls the average interval between the X and Y bursts.

Let us assume that the feedforward weights are fixed and that we modify the feedback weights from Y to X. Using stochastic gradient descent, it is possible to minimize E as follows:

$$\Delta w_{ij} = -\eta (x_i - y_j - D) \frac{\partial x_i}{\partial w_{ij}} \tag{3.2}$$

with $\eta > 0$.

The left term of this learning rule moves the weight w_{ij} in a direction that depends on only the time difference between the pre- and postsynaptic spikes, x_i and y_j, and not directly on these quantities. Hence, it is compatible with the known properties of STDP. The right term (credit assignment) will be derived in section 5.

Here we show that minimizing E leads to phase cancellation in the sense defined above. For this, let $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ and $\bar{y} = \frac{1}{M} \sum_{j=1}^{M} y_j$ denote the average firing times in X (second burst) and in Y. E can be decomposed using the Huygens theorem on inertia:

$$E = MN \left(\mathrm{var}(x) + \mathrm{var}(y) + (\bar{x} - \bar{y} - D)^2 \right). \tag{3.3}$$

In this expression, $\mathrm{var}(x)$ and $\mathrm{var}(y)$ denote the variance of the second burst in X, and of the burst in Y, respectively. The third term is the squared difference between the centers of mass of these bursts, minus the delay D. Given that the learning rule is adapting the weights from Y to X (for

the moment, we consider that the weights from X to Y are constant), it will center the X burst D milliseconds after the Y burst, and it will also minimize $\mathrm{var}(x)$, which means that it will lead to phase cancellation. Hence, minimization of a reconstruction error can be achieved by applying an STDP-type learning rule to the feedback weights.

4 The Oja Network

The network we have described can be considered as an autoencoder for time-coded vectors. It transforms an input vector (the first burst in X) into an internal representation (the burst in Y); the second burst in X encodes the difference between the input and its reconstruction. The reconstructed vector is not directly observable, but it can be deduced by subtraction. If the parameters of our network are appropriately chosen, then we may assume that neurons receive bursts of spikes during an interval where their PRC is approximately linear (see Figure 2). Thus, we may consider our network as a linear autoencoder (in section 8.2, we examine how close we are from that ideal case).

A canonical example of a linear auto-encoder is Oja's principal subspace network (Oja, 1989). The learning rule of Oja's network is derived by minimizing the mean-squared error between a random input $\mathbf{x} \in \mathbb{R}^n$ and its reconstruction $\mathbf{x}' = \mathbf{W}\mathbf{y}$. The vector $\mathbf{y} \in \mathbb{R}^m$ is an internal representation of $\mathbf{x}$, and $\mathbf{W}$ is an $m \times n$ matrix of synaptic weights. The internal representation is a linear transform of the input: $\mathbf{y} = \mathbf{A}\mathbf{x}$, with $\mathbf{A} \in \mathbb{R}^{n \times m}$. Given that there are too many degrees of freedom in this problem, Oja (1989) proposed to use feedforward weights that are equal to the transpose of the feedback weights: $\mathbf{A} = \mathbf{W}^T$. This choice leads to a very simple, predictive Hebbian learning rule:

$$\Delta \mathbf{w}_{ij} \propto \mathbf{y}_j(\mathbf{x}_i - \mathbf{x}'_i).$$

This learning rule minimizes the mean-squared error, and the error surface has no local minima (Baldi & Hornik, 1988). During learning, activities in $\mathbf{y}$ become uncorrelated, and the weights vectors become orthonormal ($\mathbf{W}^T\mathbf{W} = \mathbf{I}$). Learning stops when the weights vectors span the m-space of maximal variance of the input vector. However, convergence of the synaptic weights is not observed; the weights vectors may drift within the subspace of solutions. During learning, one observes a rapid phase of weights orthonormalization, followed by a slower search for the principal subspace.

It is important to note that this learning rule is local; the information needed to update each synapse depends on only the pre- and postsynaptic neurons. Indeed, principal components analysis can also be derived by maximizing the variance of the output neurons, under the constraint of normalized weights. However, such a normalization constraint is nonlocal,

and it is difficult to see how it could take place in a biological system. In contrast, the Oja (1989) rule does not involve an explicit renormalization. It is derived by minimizing the reconstruction error, under the constraint of symmetric weights. The choice that is important is the choice of symmetric weights. Self-normalization of the weights, and principal subspace extraction, are consequences of this choice.

In what follows, we show that this choice of symmetric weights for feedforward and feedback synapses can be generalized here and that this leads to a spiking neural network that extracts the principal subspace of its time-coded input. As in the Oja case, the network is trained not by maximizing the variance of its output but by minimizing the mean-squared reconstruction error.

5 Credit Assignment and STDP

Here we derive the credit assignment term used in learning rule (3.2). For simplicity, we consider a single neuron, so we do not need to write the index i of the postsynaptic neuron:

$$\Delta w_j = -\eta(x - y_j - D)\frac{\partial x}{\partial w_j}. \tag{5.1}$$

The partial derivative $\frac{\partial x}{\partial w_j}$ expresses the credit assignment problem for synapses. Let F denote the remaining time function of the neuron, that is, the time that remains before the next spike: $F(t) = x - t$. We have

$$F(t) = \int_{\theta(t)}^{\pi} \frac{d\theta}{(1-\cos\theta)/\tau + I(1+\cos\theta)}. \tag{5.2}$$

We shall rewrite equation 5.2 on the intervals where the integrand is continuous. In order to keep notation simple, we assume that incoming action potentials are ordered: $t_k \le t_{k+1}$ for all k in $\{1 \dots M\}$. For consistency, we use the notation $\theta^-_{M+1} = \pi$. We may write

$$F(t_j) = \sum_{k \ge j} \int_{\theta_k^+}^{\theta^-_{k+1}} \frac{d\theta}{(1-\cos\theta)/\tau + I_0(1+\cos\theta)}. \tag{5.3}$$

The partial derivative of x_i can be expressed as

$$\frac{\partial x}{\partial w_j} = \frac{\partial F}{\partial \theta_j^+}\frac{\partial \theta_j^+}{\partial w_j} + \sum_{k>j}\left(\frac{\partial F}{\partial \theta_k^+}\frac{\partial \theta_k^+}{\partial w_j} + \frac{\partial F}{\partial \theta_k^-}\frac{\partial \theta_k^-}{\partial w_j}\right). \tag{5.4}$$

In this expression, the first term describes how a modification of weight w_j directly changes the effect of a spike arriving on synapse j. The sum that follows this term describes how modifying w_j indirectly changes the effect of other spikes, for $k > j$. Each term of this sum depends on the time t_k of the kth spike.

However, at the level of a given synapse j, it is not biologically realistic to use information about the timing of spikes that arrive on other synapses k because this information is not local. In addition, we have no a priori information on the distribution of these spike times. Thus, we shall consider that these indirect terms are not correlated with $\frac{\partial E}{\partial w_j}$. For that reason, we will neglect these terms in the gradient:

$$\frac{\partial x}{\partial w_j} \approx \frac{\partial F}{\partial \theta_j^+} \frac{\partial \theta_j^+}{\partial w_j}, \tag{5.5}$$

which yields the following credit assignment term for synapse j:

$$c_j = -\frac{\partial x}{\partial w_j} \approx \frac{1}{\frac{1}{\tau} \tan^2 \frac{\theta_j^+}{2} + I_0}. \tag{5.6}$$

Note that by using $V = \tan \frac{\theta}{2}$, this term can be identified as $\frac{dt}{dV}|_{t_j^+}$. This term gates plasticity, and it depends on only the potential of the postsynaptic neuron. Therefore, this credit assignment term uses information that is local to the synapse. Its biological plausibility will be discussed in section 9.

6 Learning Rules

In this section, we detail the learning rules used to train our network. Feedback synapses are trained using the credit assignment term computed above. Feedforward synapses are trained using a symmetric learning rule.

The spike times during the first burst of X are not taken into account by our learning rules. Indeed, parameters are chosen so that the burst of Y occurs shortly before the second burst of X; the time interval between the first burst of X and the burst of Y is long and falls well outside the time window of STDP. We do this because the first burst encodes the input, while we are interested in the reconstruction error, which is encoded in the second burst. Thus, our learning rules consider only the time interval between the burst in Y and the second burst in X. This holds for both feedback and feedforward synapses.

6.1 Feedback Synapses. The feedback synapses are modified so that the Y cells tend to cancel the spiking phases of the X cells. In order to achieve this, we have seen that it is possible to use an STDP-type learning rule:

$$\Delta w_{ij} = \eta c_{ij}(x_i - y_j - D), \tag{6.1}$$

where $c_{ij} > 0$ is the credit assignment term of weights w_{ij}, computed in equation 5.6, and η is a fixed learning rate.

A potential problem is that this learning rule was derived from a quadratic cost function, which is not bounded. The learning term might take values that are too high if the interval between pre- and postsynaptic spikes is too long. This might create instabilities in the learning process, especially in the initial phase when the average interval between the Y and X bursts is not yet equal to D. In order to make the learning rule robust to this, we add a window function, f:

$$\Delta w_{ij} = \eta c_{ij} f(x_i - y_j - D). \tag{6.2}$$

In order to fit with biological observations, we used an exponential function, with decay $\tau_f = 20\,\text{ms}$:

$$\begin{cases} f(\Delta t) = -e^{\Delta t/\tau_f} & \text{if } \Delta t < 0 \\ f(\Delta t) = e^{-\Delta t/\tau_f} & \text{if } \Delta t > 0. \\ f(0) = 0 \end{cases} \tag{6.3}$$

Using this exponential function implies that the learning rule performs a gradient descent of the following error,

$$E = \sum_{i=1}^{N} \sum_{j=1}^{M} F(x_i - y_j - D), \tag{6.4}$$

where F is the integral of f.

Note that the Huygens decomposition, equation 3.3, does not hold with this error because of the nonlinearity. Thus, with this window function, it is not possible to demonstrate that minimizing E is equivalent to synchronizing the X cells. However, our experiments suggest that learning is relatively robust with respect to the choice of f. Synchronization of the X cells was obtained with various window functions f that were causal and antisymmetric (i.e., with $f(x) > 0$ for $x \geq 0$, and $f(-x) = -f(x)$). In order to demonstrate that our model can be trained using a biologically inspired window function, equation 6.3 was used in the experiment in section 8.1. In the experiment in section 8.3, $f(x) = x$ was used because faster learning is achieved in that case.

6.2 Feedforward Synapses. In order to train the feedforward synapses, we generalize the Oja choice of using symmetric weights. However, it is not possible to directly use symmetric values because there is a difference between the X and Y cells. The Y cells gate the activity of the X cells, while the X cells have to drive the Y cells. This means that the feedforward weights need to be strong enough to excite the Y cells, while this constraint is not present for the feedback weights. In order to take this into account, we added a positive constant $K > 0$ to the feedforward weights; this constant ensures that the Y cells receive enough excitation from the X cells.

Let w'_{ij} denote the weight of the feedforward synapse from neuron x_j to neuron y_i. The feedforward weights will be set as follows:

$$w'_{ji} = K - \frac{M}{N} w_{ij}. \tag{6.5}$$

In this equation, the factor $\frac{M}{N}$ balances the effect of the different sizes of populations X and Y.

In practice, K is chosen empirically, so that the average value of the learned feedback weights is close to zero. Hence, synapses from X to Y will be mostly excitatory, and synapses from Y to X will be both excitatory and inhibitory. The reason for this choice is that we want to avoid a possible synchronization of the X neurons that would not result from a prediction. In the descending phase of the PRC, inhibition tends to desynchronize the X neurons, while excitation tends to synchronize them. Thus, if the average values of the feedback weights is zero, we expect their overall effect to be neutral. In practice, the effect of K on the weights can be checked very early during learning; it is not necessary to wait until the principal subspace has been extracted.

6.3 Joint STDP Rules. It is well known that the symmetric weights matrices used in the Oja network can be arrived at through learning by using symmetric learning rules for the feedforward and the feedback weights. For this, it is necessary to add a small weight decay term, which erases the effect of the initial conditions. In fact, doing so actually increases the stability of the learning process because it introduces redundancy in the weights.

Similarly, we can replace equations 6.2 and 6.5 with two coupled STDP learning rules,

$$\begin{cases} \Delta w_{ij} = +\eta c_{ij} f(x_i - y_j - D) - \epsilon w_{ij} \\ \Delta w'_{ji} = -\eta c_{ij} \dfrac{M}{N} f(x_i - y_j - D) - \epsilon (w'_{ji} - K) \end{cases}, \tag{6.6}$$

where ϵ is a small decay term.

If the weight w_{ij} is positive, these learning rules are causal, in the sense that pre- before post- induces LTP (in the second learning rule, the roles of

x and y are reversed, and the sign is changed too). However, if the synapse is inhibitory ($w_{ji} < 0$), then $\Delta w_{ij} > 0$ means that the absolute value of the weight will become smaller (i.e., the synapse will be depressed). Therefore, our learning rule is causal for excitatory weights and anticausal for inhibitory weights. Note that this is compatible with recent experimental observations that inhibitory interneurons recurrently connected to excitatory neurons may have an anticausal STDP (Tzounopoulos et al., 2004).

The STDP window functions are not centered in zero but at an offset D. This offset is positive for feedback weights and negative for feedforward weights. The biological plausibility of this offset is discussed in section 9.

7 Stability

In order to motivate our choices, we have used an analogy with the rate-coded Oja network. However, in order to understand the dynamics of our learning rules, it is interesting to examine our model at the level of spikes and synapses without resorting to the Oja analogy. Here we show that our learning rules are stable, because feedback has a stabilizing effect on STDP.

Note that the learning rule used to train feedback weights (from Y to X) is intrinsically stable (if parameters are reasonably chosen), because it was derived from from an error minimization. Thus, the question is whether the feedforward weights will also be stable.

In order to see this, we will consider a simple feedback loop between an excitatory neuron A and an inhibitory neuron B (see Figure 5). In this simple example, there are no other neurons, so the spike time of B depends on only the spike time of A in a deterministic manner.

Neuron A excites B through a plastic synapse S, and B in turn inhibits A. We assume that A receives a positive baseline current, which causes it to spike. In order to describe the behavior of these neurons, we refer to the shape of their response curves. The response curve of A has a shape similar to that shown in Figure 2, left (dashed line, because it receive inhibition from B). The response curve of B has a shape shown in Figure 2, right (continuous line).

The first spike sent by A triggers a spike of B. Later A will spike again, even though it is inhibited by B, because it is driven by its positive baseline current. However, the inhibition sent by B will delay the second spike of A by a certain amount. This amount depends on the timing of the inhibitory spike in a way that is described by the PRC of A. Here we assume that A is close to the end of its firing cycle when it receives inhibition, so that the effect of inhibition monotonically decreases with the time of arrival of this inhibition (see Figure 2).

The action potential sent by B is backpropagated along the dendrite to the postsynaptic site of S. The weight modification of the synapse will depend on the time interval Δt between the arrival times of the AP from A and of the backpropagated AP from B.

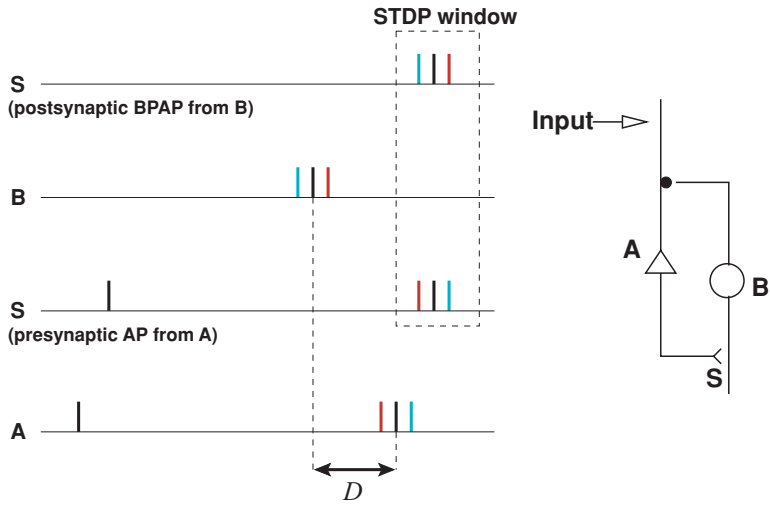

Figure 5: STDP in a feedback loop. A excites B through synapse S. B inhibits A. Black: The weight of the synapse is at the equilibrium value. Red: The weight is smaller than the equilibrium value. Blue: The weight is higher than the equilibrium value. See the text for explanations.

For simplicity, we assume that the sign of the STDP window function, f, changes for $\Delta t = 0$, and that the backpropagated AP from B takes more time to reach the synaptic site than the AP from A. Let this time difference be equal to D. Mathematically, it is equivalent to assume that the window function has an offset D, as in the previous section; however, this paragraph is easier to understand if we assume that D is caused by a difference of propagation delays. The equilibrium value of the synapse will be reached if the backpropagated AP from B and the second spike sent by A arrive simultaneously at the synaptic site S. Thus, at the equilibrium, B should fire D milliseconds before A (in black on Figure 5).

Let us now consider what happens if the weight of synapse S is below this equilibrium value (in red on Figure 5). In that case, B will fire later, because it will receive less excitation. Hence, A will be inhibited later in its firing cycle. This will cause A to fire earlier, as can be seen from its PRC. Thus, at the synaptic site, the second AP sent by A will arrive before the backpropagated AP from B. This pre- before post- order will induce LTP. Conversely, if S is too strong, the spike of B will occur earlier (blue in Figure 5). A will receive inhibition earlier in its firing cycle, which will further delay the second spike of A. The AP in S will be delayed by the same amount, and the BPAP from B will arrive first at the synaptic site. This post- before pre- order will induce LTD.

This shows that plasticity is self-stabilizing in this loop; a synapse that is too weak is potentiated, and a synapse that is too strong is depressed. The stable equilibrium is reached when the presynaptic spike and the backpropagated AP arrive at the same time.

In our network, the neurons in X and Y play the role of A and B, respectively. However, things are a bit more complicated, because feedback synapses are not necessarily inhibitory and because there are many more neurons.

Let us examine what happens if the synapse from $B \in Y$ to $A \in X$ is excitatory. To keep things simple, we consider, as previously, that only the feedforward synapse is modified and that there are only two neurons. Let us assume that the synaptic weight is above its equilibrium value. B will receive more excitation; therefore, it will fire earlier than its equilibrium firing time. As a consequence, A will be excited earlier, so its firing time will be advanced too. However, the time advance of A will always be smaller than the time advance of B, because the slope of the relevant portion of the PRC is, in absolute value, always smaller than 1 (see Figure 2, left, continuous line). This is because excitation cannot delay the next spike in the theta model. Therefore, at the synaptic site, the AP sent by A will arrive after the BPAP sent by B. This *post- before pre-* order implies that the synapse will be depressed. Thus, the learning rule is stable for excitatory synapses too.

In a full network, neurons receive several spikes, and thus spiking times cannot be predicted deterministically. Thus, spike times should be considered as a random variable. The equilibrium value of a synapse will be reached when it receives, on average, the same amount of LTP and LTD. To compute this, it is necessary to know the distribution of Δt and integrate its product with the window function.

8 Results

Now we present results from computer simulations. In each experiment, an input vector was encoded with spike times. It was necessary to make sure that the values taken by the input are within the STDP window, but also that they are in the coding interval of the neurons, that is, the range of values where the PRC is not zero. Spikes that arrive too late in the firing cycle are not taken into account by the learning rule because the credit assignment term becomes small.

Each presentation of a learning example was made during a trial of fixed duration. At the beginning of a trial, the Y neurons were initialized to their stable fixed point. The X neurons were excited by spike injection in order to encode the input. For each synapse, the credit assignment term c_{ij} was updated every time a current was transmitted.

Neurons were simulated using Euler integration of equation 2.1 with a time step of 0.2 ms. We did not model propagation delays between neurons.

Following Börgers and Koppell (2003), we used $\tau = 1$ ms and small values for the currents. The baseline currents were $I_0 = 0.001$ for the X cells and $I_0 = -0.0001$ for the Y cells (no baseline activity). Under these conditions, the intrinsic firing period of the X cells is 100 ms. In order to include two bursts of X, each trial lasted for 170 ms of simulated time.

8.1 Principal Component Analysis of a 2D Gaussian Distribution. Input patterns were constructed from pairs of spike times that were drawn from a two-dimensional gaussian distribution. Since the network does not have an absolute time reference, the input vector was encoded using relative spike times. Three input neurons (two degrees of freedom) were used to encode the 2D input. The output layer had two neurons (one degree of freedom). The network had to find a 1D representation of a 2D variable that minimizes the mean-squared reconstruction error.

The ellipsoid had a long axis of standard deviation 10 ms and a short axis of deviation 5 ms, and it was rotated by $\pi/3$. The input was encoded as follows:

$$\begin{cases} t_0 = 30 \\ t_1 = 30 + 2\nu_1 \cos(\pi/3) + \nu_2 \sin(\pi/3) \\ t_2 = 30 + \nu_2 \cos(\pi/3) + 2\nu_1 \sin(\pi/3) \end{cases}, \tag{8.1}$$

where ν_1 and ν_2 are two independent random variables drawn from a gaussian distribution of variance 1. Input spikes times were centered around $t = 30$ ms, where $t = 0$ denotes the beginning of a trial. Network parameters were $D = 35$ ms and $K = 0.0095$. The learning rate was $\eta = 10^{-8}$, and the weights were initialized with random values drawn from a uniform distribution between -0.001 and 0.001.

The network was trained for 8×10^5 iterations. As mentioned earlier, K was chosen so that the average value of the synaptic weights was zero; thus, during the first iterations of learning, the variance of the second burst of X was almost the same (15.54 ms^2) as the variance of the first burst (15.76 ms^2).

Figure 6 shows the evolution of the weights, the error E and the variance of the bursts in X and Y. The weights started from small initial values and settled to stable final values. In addition, var(x) decreased while var(y) increased, which indicates that the Y cells learned to cancel the variance of the X cells. Eventually the average variance of the second burst of X was 4.61 ms^2.

Note that E is not minimized during learning, even though the learning rule was derived by minimizing it. This is because we considered only the feedback weights in this error minimization, assuming that the feedforward weights were constant. However, in our network, we also modify the feedforward weights in a way that increases the variance in Y. As a result, E does not necessarily decrease during learning.

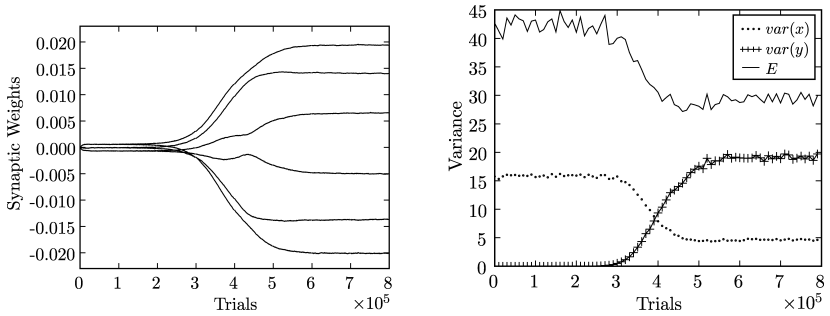

Figure 6: Evolution of the network during 8×10^5 learning iterations. (Left) Synaptic weights. (Right) Variance of the bursts in X and Y, and the error E.

Figure 7 shows the distributions of the input, prediction, and error after the network has been trained. Relative spike times of the X cells are plotted. In the first figure (top left), the input (first burst of X) is plotted with white dots, and the error (second burst of X) is plotted with dark dots. The second figure (top right) shows the prediction made by the Y cells (using dark dots). The prediction was obtained by subtracting the error from the input. Prediction dots are distributed along the principal axis of the input, which demonstrates that the principal component of the distribution has been extracted.

In this experiment, there is no drift of the synaptic weights once the network is trained, and the final values of the weights do not depend on the initial conditions, up to a permutation of the Y neurons. This is because the network extracts only the first component of the input, which is unambiguous. This, however, does not hold for larger networks.

8.2 Linear "Phase" Model. In the choice of our learning rules, we tried to replicate the properties of the Oja network. However, the Oja model is not completely isomorphic to our network, for the following reasons.

First, theta neurons have a positive PRC (type 1 neurons). This means that spike times can encode only positive values, even though synaptic weights can be of both signs. In order to encode variables that can take both signs, one would need a PRC that crosses zero, so that the effect of a current is reversed when its arrival time crosses the zero of the PRC. Hence, we may view the code used by our network as a positive code: early spikes code for high values and late spikes code for values close to zero.

Second, the direction of the weight update in our learning rule is given by a difference between presynaptic times x_i and postsynaptic times y_j, which means that, unlike the Oja rule, it mixes components of the output and components of the reconstruction error.

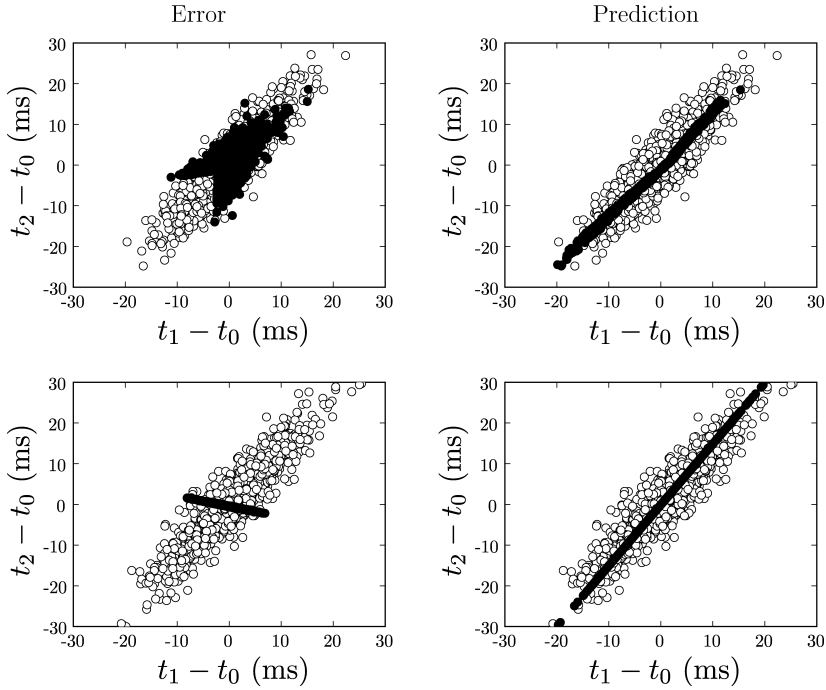

Figure 7: Relative spiking times of the X cells during the first and second burst, after the network has been trained. (Left) The first burst (white dots), which encodes the input, and the second burst (dark dots), which encodes the prediction error. (Right) The same input distribution (white) and its reconstruction (dark dots), that is, the difference between input and error shown on the left side. (Bottom row) Results obtained with the linear phase model.

Finally, we assumed that the PRC of our neurons is linear in the interval where the bursts of spikes are spread. Since this is only approximately true (see Figure 2), we should expect nonlinearities to have an impact on performance; spikes that arrive in nonlinear regions of the PRC are likely to fail at canceling the input phase, which should increase the reconstruction error.

Thus, in order to describe our network in a more accurate way, we defined a linear "phase" model. This model is supposed to replicate the behavior of our network, assuming that the theta neurons respond linearly. The phase model was defined as follows:

- The input vector input $\mathbf{x}$ is an n-vector with positive components, not centered around zero.

- The internal representation $\mathbf{y}$ is obtained linearly by multiplying the input with an $n \times m$ weight matrix: $\mathbf{y} = \mathbf{W}\mathbf{x}$.
- The reconstruction of the input is obtained with the transpose of $\mathbf{W}$: $\mathbf{x}' = \mathbf{W}^T\mathbf{y}$. The error is $\mathbf{x} - \mathbf{x}'$.
- The learning rule attempts to replicate our STDP learning rule. It uses the difference between components of the output and of the error:

$$\Delta \mathbf{w}_{ij} = -\eta \mathbf{y}_j(\mathbf{y}_j - (\mathbf{x}_i - \mathbf{x}'_i)).$$

We compared our spiking network to this phase model. The phase model was trained on the same gaussian distribution as above but centered around $t = 15$ ms. The resulting prediction and error distributions are plotted, respectively, on Figure 7 (bottom). These distributions are qualitatively similar to what is obtained with the spiking network, which suggests that the phase model describes well the behavior of our spiking network.

The direction of projection is the principal axis of the input distribution (see Figure 7, bottom right). However, the error is not distributed orthogonally to the input distribution (see Figure 7, bottom left). This nonorthogonality is due to the noncentering of the input. The variance of the error is smaller than for the spiking network, equal to 0.05 times the initial variance. This suggests that nonlinearities of the PRC decrease the performance of our spiking network.

8.3 Encoding Natural Images. In order to demonstrate that the learning algorithm scales to large networks, we trained a network of size 256×64 with a set of whitened natural images, used in Olshausen and Field (1996). (Images were retrieved online from http://redwood.berkeley.edu/bruno/sparsenet/.) For this experiment we used the learning rules 6.2 and 6.5 and $f(x) = x$, because it learns faster. In addition, a propagation delay of 40 ms from X to Y was introduced. Network parameters were $\eta = 10^{-9}$, $D = 35$ ms, and $K = 0.00023$. Synaptic weights were initialized with random values drawn from a uniform distribution between -0.00001 and 0.00001.

On each trial, a random patch of size 16×16 was extracted from a random image of the data set. Gray pixel values from the data set were linearly mapped to time values, with a scaling factor equal to 20. As a result, the variance of the input (first burst in X) was 30 ms^2.

The network was trained for 10^6 iterations. During this interval, the variance of the Y burst increased from zero to 12 ms^2, and the variance of the second burst in X decreased from 29.6 ms^2 to 12.5 ms^2. This means that the average error per pixel decreased from 5.44 ms to 3.61 ms. For comparison, the mean reconstruction error performed by Oja's principal subspace network (Oja, 1989) trained on the same image patches is 2.57 ms/pixel. An image from the data set and its reconstruction are shown in Figure 8. As in the previous experiment, the reconstruction was obtained indirectly, by subtracting the spike times of the first and second bursts in X.

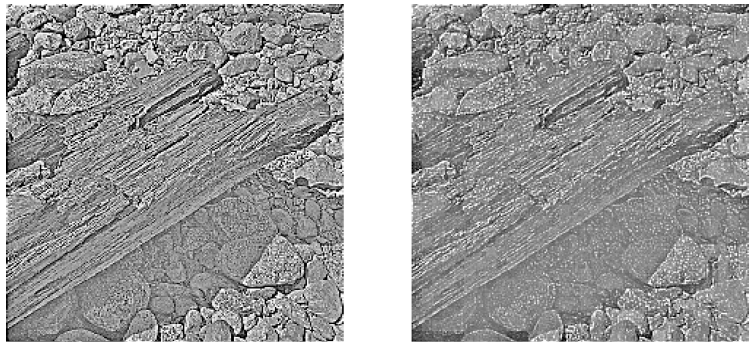

Figure 8: An image from the data set (left), and its reconstruction from spike times (right). The 512×512 image was divided into 16×16 patches and encoded using 64 neurons. The reconstruction is the difference between firing times of the first burst and the second burst of X. The variance of the time-coded images was 30 ms^2 per pixel. After learning, the mean squared error was 12.5 ms^2 per pixel.

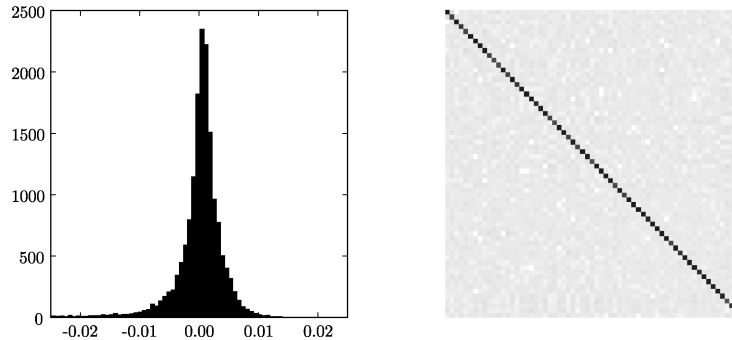

Figure 9: (Left) Distribution of the synaptic weights learned by the network. There are $256 \times 64 = 16{,}384$ weights. The distribution is unimodal and centered around zero. (Right) Inner product of the weights matrix and its transpose. The gray levels represent values from -0.0005 (white) to $+0.006$ (black). The result is close to the identity matrix, which indicates that learned weights vectors are orthogonal.

Figure 9 shows the distribution of the learned synaptic weights and their scalar products. The weights have a unimodal distribution, and weight vectors are almost orthogonal. Figure 10 shows the weight vectors, displayed as a set of receptive fields. Some neurons have nonlocal receptive fields that are similar to the receptive fields learned by Oja's network trained on the same data set. Some other neurons have receptive fields that are

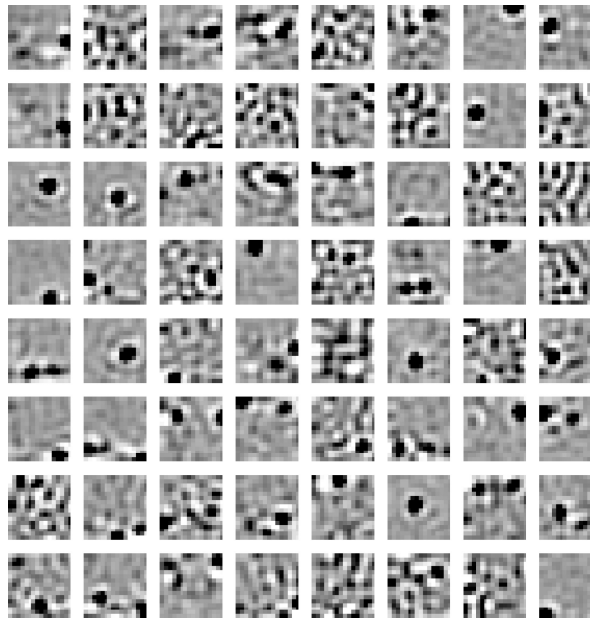

Figure 10: Synaptic weights learned by the network, represented as receptive fields. Sixty-four neurons were trained to represent natural images patches of size 16×16. The gray levels represent values between -0.007 (black) and $+0.007$ (white).

more localized; this could be caused by nonlinearities in the network, as suggested by the previous experiment.

9 Biological Plausibility

The learning rules used in our model are not standard. First, we have assumed the existence of a credit assignment term, c_{ij}, that modulates STDP. Second, the STDP window function is centered not in zero but at an offset D.

The credit assignment term c_{ij} involves only local information. Indeed, this term depends on the potential of the postsynaptic neuron at the time when the spike is transmitted. Experimentally, it has been observed that it is possible to modulate synaptic plasticity by varying the membrane potential of a postsynaptic neuron during presynaptic stimulation. More precisely, the number of postsynaptic N-methyl D-aspartate (NMDA) receptors that are activated during stimulation gates synaptic plasticity (Shouval et al., 2002), and the contribution of NMDA receptors varies with the level of postsynaptic depolarization during stimulation. Thus, the voltage-dependent activation of NMDA receptors is compatible with our credit assignment hypothesis.

In general, the STDP functions observed biologically have a sign reversal in zero. In contrast, our STDP uses an offset D. In section 7 we have interpreted this offset as the result of different propagation times for the AP and BPAP. This, however, does not seem biologically plausible, because propagation times are in general small compared to integration times; in contrast, in our experiments, D is equal to about one-third of the oscillation period. Therefore, we need to examine whether this parameter is plausible and whether it is really needed in our model.

STDP windows with three regions (LTD, LTP, and LTD) and two sign reversals have been described (Abbott & Nelson, 2000). In this case, sign reversals may occur before or after zero. According to Shouval et al. (2002), the second sign reversal of STDP may occur late after the arrival of the dendritic AP due to the long tail of the backpropagated dendritic AP. These sign reversals are located at offsets that are compatible with our proposal.

In addition, it might be possible to do away with that offset in our model. We need the Y cells to gate the X cells. For this, the Y cells should fire before the second burst of X. By adding an offset D to the learning rule, we observe that the Y burst takes place, on average, D milliseconds before the second burst of X, which satisfies our requirements. This is why we introduced D.

However, this gating could in principle be achieved differently. In our simulations, we observe some overlap between the Y burst and the second X burst. Even though most of the Y cells fire early enough to gate the X cells, some Y cells that do not respond well to the current input fire after the X cells. In principle, we could imagine increasing this overlap until both bursts are centered at the same point in time (using $D = 0$). This would mean that on average, only half of the Y cells would be used to gate the X cells on each iteration. In practice, however, we could not train our network with $D = 0$. This is because of the shape of the PRC of theta neurons. If the time interval between spikes in Y and X is too small, learning becomes very slow because the credit assignment term decreases as the PRC of the X cells slowly approaches zero. In order to learn efficiently, the Y cells must spike in a time window where the PRC of the X cells is not constant. In our experiments, the best results were obtained when D was about one-third of the oscillation period. However, this would certainly be different with a more realistic PRC.

Finally, another way to avoid using an offset might be found by using more realistic STDP window functions. To see this, we need to consider the statistical nature of the learning process. In section 7, we considered a simplified example with only two neurons. In this case, all the timings depend on the initial spike in a deterministic manner. However, in our network, neurons receive inputs from many synapses, and the time interval Δt between the AP and BPAP on a synapse is a random variable. Thus, the equilibrium value of a synapse needs to be understood in a stochastic sense. Given a distribution of input vectors, a synapse will reach its equilibrium value when it receives, on average, the same amount of LTD and LTP. The

expected drift of a synaptic weight can be computed by integrating the product of the distribution of Δt with the window function of STDP. If we assume that the STDP window is exactly antisymmetric and centered in zero (no offset), then the equilibrium is reached if the distribution of Δt is centered around zero. However, this does not hold if the window function is not antisymmetric (i.e., the positive area of the curve is not equal to the negative area, as is often the case in biological synapses). In that case, the equilibrium value of the weight will be reached for a distribution of Δt that is not centered around zero. This means that it might be possible to remove the delay D from our equations by using a skewed STDP window function. This analysis, however, falls beyond the scope of this letter.

10 Conclusion

Using spiking neurons and STDP, we have shown how to implement a predictive learning rule that is similar to Oja's learning rule. For this, we proposed a new interpretation of the shape of STDP; asymmetric STDP is seen here as a self-stabilizing mechanism that takes place within a feedback loop between excitatory and inhibitory neurons.

Our results suggest a possible computational role for neural synchrony and oscillations. In our model, synchronization of the X cells is useful because it allows the Y cells to learn an optimized representation of the stimulus. However, the resulting representation is available in the Y cells only, not in the cells that are being synchronized. Thus, the second burst of X encodes a residual error. Hence, the benefits of synchronization are indirect; synchronization is used not to represent the input but to learn a higher-level representation.

This hypothesis is supported by some recent experimental observations about oscillations and the familiarity of stimuli. In the olfactory system of insects and mammals, it has been observed that the amplitude of odor-induced oscillations increases when a presented stimulus becomes familiar (Stopfer & Laurent, 1999; Martin, Gervais, Hugues, Messaoudi, & Ravel, 2004). If one assumes that the amplitude of oscillations is a correlate of neural synchrony, then our model might provide an explanation for this experimental observation. Indeed, when a stimulus becomes familiar, its internal representation becomes more accurate, and its reconstruction error becomes smaller, which in our model will result in increased synchrony.

However, our current results are not sufficient to account for sustained oscillations. The main problem is that our model considers only two bursts of the X cells: a first burst that encodes the input and a second burst that encodes the reconstruction error. This means that the code is not consistent across bursts, and it remains to be shown how this could be generalized to an oscillation lasting for several cycles. One possibility could be to assume that the input vector is being received by the X cells on each cycle of the oscillation (it could be transmitted as a continuous current, or it could be

received on each cycle). In this case, the phases of the X cells would encode the residual error, and the phases of the Y cells would cancel out the input on each cycle, performing a continuous reestimation of the input.

References

Abbott, L. F., & Nelson, S. B. (2000). Synaptic plasticity: Taming the beast. *Nature Neuroscience, 3*, 1178–1183.

Baldi, P., & Hornik, K. (1988). Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks, 2*(1), 53–58.

Bi, G. Q., & Poo, M. M. (1998). Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *J. Neurosci, 18*(24), 10464–10472.

Bohte, S., Kok, J., & La Poutré, H. (2002). Spike-prop: Error-backpropagation in multi-layer networks of spiking neurons. *Neurocomputing, 48*, 17–37.

Börgers, C., & Kopell, N. (2003). Synchronization in networks of excitatory and inhibitory neurons with sparse, random connectivity. *Neural Computation, 15*, 509–538.

Cateau, H., & Fukai, T. (2003). A stochastic method to predict the consequence of arbitrary forms of spike-timing-dependent plasticity. *Neural Computation, 15*, 597–620.

Cunningham, M. O., Davies, C. H., Buhl, E. H., Kopell, N., & Whittington, M. A. (2003). Gamma oscillations induced by kainate receptor activation in the entorhinal cortex in vitro. *Journal of Neuroscience, 23*(30), 9761–9769.

Ermentrout, G. B. (1996). Type I membranes, phase resetting curves, and synchrony. *Neural Computation, 8*, 979–1001.

Gerstner, W., & Kistler, W. M. (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge: Cambridge University Press.

Gütig, R., Aharonov, R., Rotter, S., & Sompolinsky, H. (2003). Learning input correlations through non-linear temporally asymmetric Hebbian plasticity. *Journal of Neuroscience, 23*, 3697–3714.

Gütig, R., & Sompolinsky, H. (2006). The tempotron: A neuron that learns spike timing-based decisions. *Nature Neuroscience, 9*, 420–428.

Gutkin, B. S., Ermentrout, G. B., & Reyes, A. D. (2005). Phase-response curves give the responses of neurons to transient inputs. *J. Neurophysiol, 94*, 1623–1635.

Hopfield, J. J. (1995). Pattern recognition computation using action potential timing for stimulus representation. *Nature, 376* (6535), 33–36.

Kempter, R., Gerstner, W., & van Hemmen, J. (1999). Hebbian learning and spiking neurons. *Physical Review E, 59*, 4498–4514.

Kitano, K., & Fukai, T. (2004). Temporal characteristics of the predictive synchronous firing modeled by spike-timing-dependent plasticity. *Learning and Memory, 11*, 267–276.

Legenstein, R., Naeger, C., & Maass, W. (2005). What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation, 17*, 2337–2382.

Lengyel, M., Kwag, J., Paulsen, O., & Dayan, P. (2006). Matching storage and recall: Hippocampal spike timing-dependent plasticity and phase response curves. *Nature Neuroscience, 8*, 1677–1683.

Maass, W. (1996). Lower bounds for the computational power of networks of spiking neurons. *Neural Computation, 8*(1), 1–40.

Markram, H., Lübke, J., Frotscher, M., & Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science, 275*, 213–215.

Martin, C., Gervais, R., Hugues, E., Messaoudi, B., & Ravel, N. (2004). Learning modulation of odor-induced oscillatory responses in the rat olfactory bulb: A correlate of odor recognition? *Journal of Neuroscience, 24* (2), 389–397.

Martinez, D. (2005). Oscillatory synchronization requires precise and balanced feedback inhibition in a model of the insect antennal lobe. *Neural Computation, 17*, 2548–2570.

McKennoch, S., Voegtlin, T., & Bushnell, L. (2009). Spike-timing error backpropagation in theta neuron networks. *Neural Computation, 21*, 9–45 .

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

Olshausen, B., & Field, D. (1996). Sparse coding of natural images produces localized, oriented, bandpass receptive fields. *Nature, 381*, 607–609.

Rao, R. P. N. (1999). An optimal estimation approach to visual perception and learning. *Vision Research, 39*(11), 1963–1989.

Rao, R. P. N., & Ballard, D. H. (1997). Dynamic model of visual recognition predicts neural response properties in the visual cortex. *Neural Computation, 9*(4), 721–763.

Rao, R. P. N., & Sejnowski, T. J. (2001). Spike-timing dependent plasticity as temporal difference learning. *Neural Computation, 13*, 2221–2237.

Shouval, H., Bear, M., & Cooper, L. (2002). A unified model of NMDA receptor-dependent bidirectional synaptic plasticity. *Proc. Natl. Acad. Sci. USA, 99*(16), 10831–10836.

Song, S., Miller, K. D., & Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience, 3*, 919–926.

Stopfer, M., & Laurent, G. (1999). Short-term memory in olfactory network dynamics. *Nature, 402*, 664–668.

Stuart, G. J., & Sakmann, B. (2002). Active propagation of somatic action potentials into neocortical pyramidal cell dendrites. *Nature, 367*, 69–72.

Tzounopoulos, T., Kim, Y., Oertel, D., & Trussell, L. O. (2004). Cell-specific, spike timing-dependent plasticities in the dorsal cochlear nucleus. *Nature Neuroscience, 7*(7), 719–725.

Voegtlin, T. (2007). Temporal coding using the response properties of spiking neurons. In B. Schölkopf, J. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems, 19* (pp. 1457–1464). Cambridge, MA: MIT Press.

Whittington, M. A., Traub, R. D., Kopell, N., Ermentrout, B., & Buhl, E. H. (2000). Inhibition-based rhythms: Experimental and mathematical observations on network dynamics. *Int. J. Psychophysiol., 38*(3), 315–336.

Widrow, B., & Hoff, M. E. (1960). Adaptive switching circuits. In *IRE WESCON Convention Record* (pp. 96–104).

Received February 7, 2008; accepted November 6, 2008.