From STDP towards Biologically Plausible Deep Learning

Yoshua Bengio[1], **Asja Fischer, Thomas Mesnard, Saizheng Zhang and Yuhai Wu**

Montreal Institute for Learning Algorithms, University of Montreal, Montreal, QC, H3C 3J7
[1]CIFAR Senior Fellow

Abstract

We introduce a predictive objective function for the rate aspect of spike-timing dependent plasticity (STDP), i.e., ignoring the effects of synchrony of spikes but looking at spiking *rate changes*. The proposed weight update is proportional to the presynaptic spiking (or firing) rate times the *temporal change* of the integrated postsynaptic activity. We present an intuitive explanation for the relationship between spike-timing and weight change that arises when the weight change follows this rule. Spike-based simulations agree with the proposed relationship between spike timing and the temporal change of postsynaptic activity and show a strong correlation between the biologically observed STDP behavior and the behavior obtained from simulations where the weight change follows the gradient of the predictive objective function. Finally, we draw links between this objective function, neural computation as inference, score matching, and variational EM.

1. Introduction

Spike-Timing Dependent Plasticity (STDP) is believed to be the main form of synaptic change in neurons (Markram and Sakmann, 1995; Gerstner *et al.*, 1996) and it relates the expected change in synaptic weights to the timing difference between postsynaptic spikes and presynaptic spikes. Although it is the result of experimental observations in biological neurons, its generalization (outside of the experimental setups in which it was measured) and interpretation as part of a learning procedure that could explain learning in deep networks remains a topic where more exploration is needed. This paper aims at contributing to this exploration by proposing a predictive objective function for STDP, or at least of its rate aspect, such that STDP performs stochastic gradient descent (SGD) on that objective.

This proposal follows up on Xie and Seung (2000), who nicely showed how STDP can correspond to a differential Hebbian or anti-Hebbian plasticity, i.e., by dropping terms related to the spikes' specific timing (by considering spikes as random events and averaging out that randomness), the synaptic change is seen to be proportional to the product of presynaptic firing rate and the temporal derivative of the postsynaptic firing rate. We revisit this idea and come up with a very close proposal for a learning rule that corresponds to stochastic gradient descent on a predictive objective. The proposed learning rule captures the effect of changes in firing rates (and the underlying aggregated and integrated activity) but does not attempt to capture phase information in the spike trains, only the aspect due to the average firing rates of neurons. The phase aspect of STDP may also have a machine learning role but is left for future work.

As usual, we assume the existence of a non-linear transformation ρ that is monotonically increasing, from the integrated activity (the expected membrane potential, averaged over the random effects of both pre- and postsynaptic spikes) to the actual firing rate. In deriving a link between STDP and their rate-based learning rule Xie and Seung (2000) started by assuming a particular pattern relating spike timing and weight change, and then showed that the resulting weight change could be approximated by the product of presynaptic firing rate and temporal rate of postsynaptic firing rate. Instead, we go in the other direction, showing that if the weight change is proportional to the product of presynaptic firing rate (or simply the presence of a spike) and the postsynaptic *integrated activity* (not firing rate), then we recover a relationship between spike timing and weight change that has the precise characteristics of the one observed experimentally by biologists. We present both an easy to understand theoretical justification for this result, as well as, simulations that confirm it.

Furthermore, we show that this proposed learning rule corresponds to stochastic gradient descent on a predictive objective function: collectively, *neurons would be trying predict their future state better*. In addition to the value this brings in terms of fitting sequences of observed sensory inputs, we conjecture that this complies with the already well known hypothesis (Hinton and Sejnowski, 1986; Friston and Stephan, 2007; Berkes *et al.*, 2011) that, given a state of sensory information (current and past inputs), neurons are collectively performing *inference*, i.e., moving towards configurations that better "explain" the observed sensory data. We can think of the configuration of internal neurons (hidden units or latent variables) as an "explanation" for the observed sensory data. As we argue, this predictive objective function could help the brain to move its parameters towards making those more likely (and more coherent with the observed sensory input) future states more probable in the future, which makes sense in an EM perspective.

2. Spike-timing dependent plasticity

Spike-timing dependent plasticity (STDP) is a central subject of research in synaptic plasticity but much more research is needed to solidify the links between STDP and a machine learning interpretation of it at the scale of a whole network, i.e., with "hidden layers" which need to receive a useful training signal. See Markram *et al.* (2012) for a recent review of the neuroscience research on STDP.

What has been observed in STDP is that there is a weight change if there is a presynaptic spike in the temporal vicinity of a postsynaptic spike: that change is positive if the postsynaptic spike happens just after the presynaptic spike (and larger if the timing difference is small), negative if it happens just before (and again, larger if the timing difference is small, on the order of a few milliseconds), as illustrated with the biological data shown in Figure 1 (left) from Bi and Poo (2001). The amount of change decays to zero as the temporal difference between the two spikes increases beyond a few tens of milliseconds. We are thus interested in this temporal window around a presynaptic spike during which a postsynaptic neuron spikes, before or after the presynaptic spike, and this induces a change of the weight.

2.1. Rate-based aspect of STDP

Let s_i represent an abstract variable characterizing the integrated activity of neuron i, with $\dot{s}_i$ being its temporal rate of change. To give a precise meaning to s_i, which we are going to consider as the main *state variable* in our theory, we define $\rho(s_i)$ as being the firing rate of neuron i, where ρ is a bounded non-linear activation function that converts the integrated voltage potential into a probability of firing.

A hypothesis inspired by Xie and Seung (2000) and Hinton (2007) is that the STDP weight change can be associated with the temporal rate of change of postsynaptic activity, as a proxy for its association with post minus presynaptic spike times. Both of the above contributions focus on the rate aspect of neural activity, and this is also the case of this paper. However, like Hinton (2007) we prefer to consider the rate of change of the integrated activity s_i rather than that of its corresponding firing rate, $\rho(s_i)$, because, as we will see below, it makes STDP perform stochastic gradient descent on a clear and interpretable objective function.

The proposed equation for the average weight change is thus the following:

$$\Delta W_{i,j} \propto \dot{s}_i \rho(s_j) \ . \tag{1}$$

Since we are talking about the *average change*, we could as well have written it

$$\Delta W_{i,j} \propto \dot{s}_i \xi_j \ , \tag{2}$$

where ξ_j is the binary indicator of a spike from neuron j. This works if we assume that the input spikes are randomly drawn from a Poisson process with a rate proportional to $\rho(s_j)$, i.e., we ignore spike synchrony effects (which we will do in the rest of this paper, and leave for future work to investigate). Note that in our simulations, we approximated the Poisson process by performing a discrete-time simulation with a binomial draw of the binary decision spike vs no spike within the time interval of interest.

2.2. Why it works

Consider a rising postsynaptic activity level, i.e., $\dot{s}_i > 0$, as in Figure 2 (left) and a presynaptic spike occurring somewhere during that rise. We assume a Poisson process for the postsynaptic spike as well, with a rate proportional to $\rho(s_i)$. According to this assumption, postsynaptic spikes are more likely in a fixed time window following the presynaptic spike than in a window of the same length preceding it. Therefore, if only one spike happens over the window, it is more likely to be after the presynaptic spike, yielding a positive spike timing difference, at the same time as a positive weight change. The situation would be symmetrically reversed if the activity level was decreasing, i.e., $\dot{s}_i < 0$ and negative spike times are more likely.

Furthermore, the stronger the slope $\dot{s}_i$, the greater will this effect be, also reducing the spike timing. This assumes that when spikes occur on both sides, the effects on $W_{i,j}$ cancel each other. Of course, the above reasoning runs directly in reverse when the slope of s_i is negative, and one gets negative updates, in average. To validate these hypothesis, we ran the simulations presented in Section 5. They confirm these hypotheses and show that Eq. 1 yields a relationship between spike timing difference and weight change that is consistent with biological observations (Fig. 1).

3. An objective function for STDP

3.1. Proposed STDP criterion

In the following, we move to discrete time notation, with s_t being the whole state vector of all neurons and $s_{t,i}$ the state

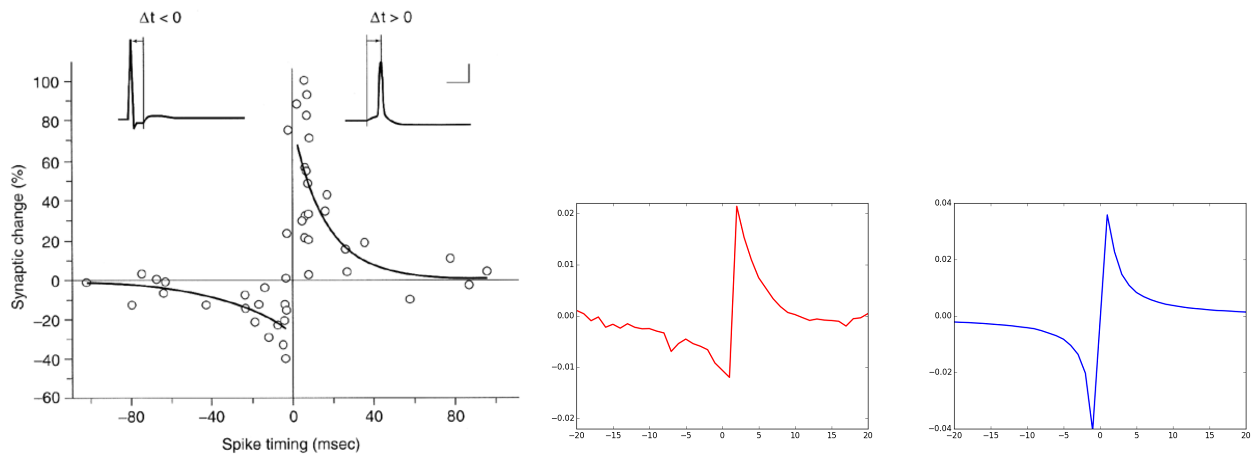

Figure 1. Left: Biological observation of STDP weight change, vertical axis, for different spike timing differences (post minus pre), horizontal axis. From Shepherd (2003), with data from Bi and Poo (2001). Compare with the result of the simulations using the objective function proposed here (middle).
Middle and right:Spike-based simulation shows that when weight updates follow SGD on the proposed predictive objective function, we recover the biologically observed relationship between spike timing difference (horizontal axis, postsynaptic spike time minus presynaptic spike time) and the weight update (vertical axis). Middle: the weight updates are obtained with the proposed update rule (Eq. 1). Right: the weight updates are obtained using the nearest neighbor STDP rule. Compare with the biological finding, left.

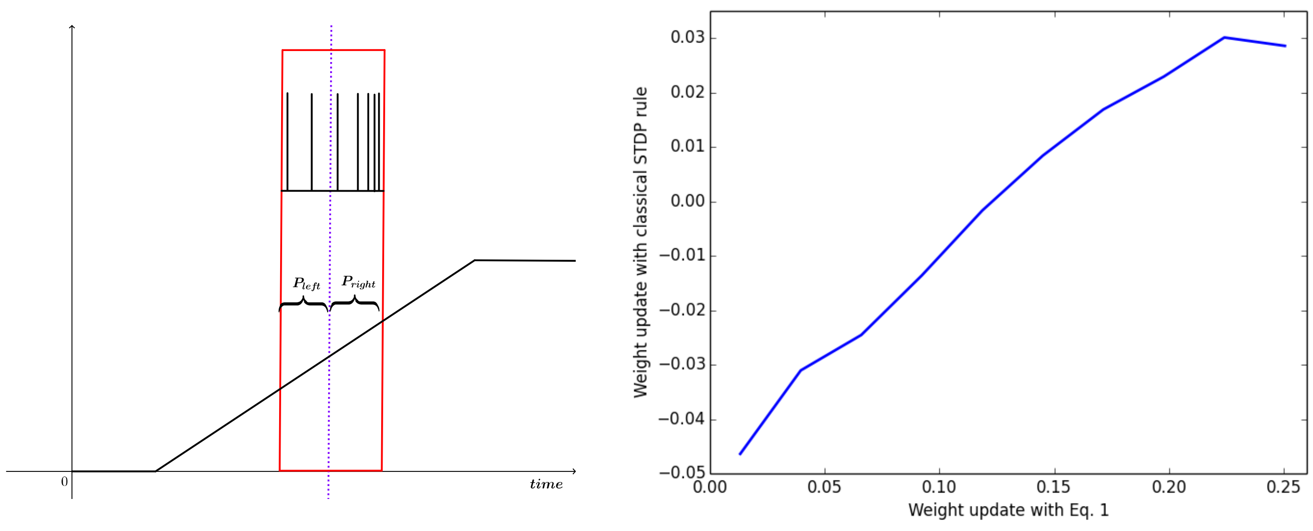

Figure 2. Left: When the postsynaptic rate (y-axis) is rising in time (x-axis), consider a presynaptic spike (middle dotted vertical line) and a window of sensitivity before and after (bold red window). Because the firing probability is greater on the right sub-window, one is more likely to find a spike there than in the left sub-window, and it is more likely to be close to the presynaptic spike if that slope is higher, given that when spikes occur on both sides, no weight update occurs. This induces the appropriate correlation between spike timing and the temporal slope of the postsynaptic activity level, which is confirmed by the simulation results of Section 5 and the results on the right.
Right: The average STDP update according to the STDP nearest neighbor rule (vertical axis) vs the weight update according to the proposed rule (Eq. 1) (horizontal axis). We see that in average over samples (by binning values of the x-axis values), the two rules agree closely, in agreement with the visual inspection of Fig. 1.

of neuron i at time t. If we take for granted that the learning rule in Eq. 1 is consistent with STDP, there remains the question of its machine learning interpretation, and this paper is only a first step in the direction of answering this difficult question.

Towards this goal, we propose the following objective function J_{STDP} per time step t, that makes the above weight update equation perform stochastic gradient descent on J_{STDP}:

$$J_{\text{STDP}} = \frac{1}{2}||f(s_{t-1}, \eta_{t-1}) - s_{t+1}||^2 \tag{3}$$

where $f(s_{t-1}, \eta_{t-1}) = s_t$ computes the next state value and f is the parametrized function that represents the stochastic transformation from the previous neural state s_{t-1} to the next neural state s_t, with injected noise η_{t-1}. That noise captures the effects of synaptic noise and spike quantization (modeled as a Poisson distribution, or equivalently with an independent binomial probability of spiking in each small time interval).

3.2. How it yields the proposed STDP update rule

We now show how the above objective function J_{STDP} can give rise to Eq. 1, **if** the following condition (which we call **condition 1**) is satisfied:

$$\frac{\partial f_i(s_{t-1}, \eta_{t-1})}{\partial W_{i,j}} \propto \rho(s_{t-1,j}) \tag{4}$$

i.e., there is a linear relationship between the updated state $f_i(s_{t-1}, \eta_{t-1})$ and $W_{i,j}$, proportional to the average input signal $\rho(s_{t-1,j})$ from neuron j. Under **condition 1**, we then obtain the weight gradient

$$\begin{aligned}\frac{\partial J_{\text{STDP}}}{\partial W_{i,j}} &= \frac{\partial J_{\text{STDP}}}{\partial s_{t,i}} \frac{\partial f(s_{t-1}, \eta_{t-1})}{\partial W_{i,j}} \\ &\propto \frac{\partial J_{\text{STDP}}}{\partial s_{t,i}} \rho(s_{t-1,j}) \\ &\propto (s_{t,i} - s_{t+1,i})\rho(s_{t-1,j})\end{aligned} \tag{5}$$

which corresponds to Eq. 1, as claimed, when $\Delta W_{i,j} \propto -\frac{\partial J_{\text{STDP}}}{\partial W_{i,j}}$.

Something notable and not apparent in the typical analysis of STDP is that this learning rule *also predicts that the average weight change will be 0 when the postsynaptic firing rate remains constant*, even if it is large. According to this prediction, it is not enough that the presynaptic and postsynaptic neurons fire together, averaged over multiple runs, change only occurs if the postsynaptic firing rate changes.

3.3. Neural computation as leaky integrator

Let us now consider the stochastic update operation f in more detail. As usual in neuron models, we assume that unclamped neurons are performing leaky temporal integration of their past inputs. Let us denote x_t for the clamped part of s_t (i.e., the externally driven input) and h_t for the unclamped part, i.e., $s_t = (x_t, h_t)$. Let $f = (f_x, f_h)$ to denote the parts of f that respectively outputs the predictions on the clamped units and on the unclamped units. The time evolution of the unclamped units is assumed to follow a leaky integration equation, i.e.,

$$h_{t+1} = f_h(s_t, \eta_t) = h_t + \epsilon(R_h(\tilde{s}_t) - s_t) \tag{6}$$

where $\tilde{s}$ represents a noisy corruption of s due to synaptic noise and spiking effects, grossly modeled by additive noise:

$$\tilde{s}_t = s_t + \eta_t \ , \tag{7}$$

and we see that the above equation corresponds to the discretization of a differential equation

$$\tau \dot{h} = R_h(s + \eta) - h$$

which brings h exponentially fast towards the "target" value $R_h(s)$, along with the random walk movements brought by the noise η. In the above equations, $R(s) = (R_x(s), R_h(s))$ represents the network-generated pressure on neurons, i.e., $R_i(s)$ is what the rest of the network asks neuron i to move towards. With this leaky integrator structure, we see that **condition 1** is converted into a similar condition on $R_i(s)$, which we call **condition 2**:

$$\frac{\partial R_i(\tilde{s})}{\partial W_{i,j}} \propto \rho(\tilde{s}_j) \ . \tag{8}$$

In section 4.1 we introduce a definition of R which satisfies the above property and also makes sense from the point of view of a machine learning interpretation of J_{STDP} introduced next, as a form of score matching.

3.4. More on the Continuous Time Perspective

In continuous time, the proposed STDP update rule can be written as in Eq. 1. This can be viewed as the derivative of a continuous-time version of the objective function J_{STDP}, with respect to $W_{i,j}$ with

$$J = \frac{1}{2}||\dot{s}||^2 \tag{9}$$

where s is the continuous-time vector-valued state of the neural network, and $\dot{s}$ is its temporal derivative. According to the neural energy function proposed above, the elements $\dot{s}_i$ of $\dot{s}$ are as follows:

$$\dot{s}_i \propto \rho'(s_i) \sum_j W_{i,j}\rho(s_j) - s_i \ . \tag{10}$$

This yields the weight gradient

$$\frac{\partial J}{\partial W_{i,j}} = 2\dot{s}_i \rho(s_j)\rho'(s_i) \ , \tag{11}$$

which would lead to a weight update proportional to the proposed STDP rule for $\Delta W_{i,j}$, Eq. 1.

4. Predictive interpretation of the STDP criterion

What could be the purpose of minimizing J_{STDP} in Eq. 3 from a machine learning perspective?

If we consider a sequence of inputs, then predicting what will come next corresponds to maximizing the likelihood of the observed sequence, a very natural criterion for fitting a sequence of observations $x_1, \ldots, x_T$, via the decomposition

$$\begin{aligned} P(x_1, \ldots, x_T) &= \prod_t P(x_t | x_{t-1}, x_{t-2}, \ldots, x_1) \\ &= \prod_t P(x_t | s_{t-1}) \end{aligned} \tag{12}$$

when the past sequence $(x_1, \ldots, x_{t-1})$ is summarized by a state s_{t-1}, i.e. we are trying to predict the next observation given the current state.

However, this leaves an important question out of the picture, which has been central in the last decades of research in unsupervised learning algorithms for neural networks: what if x_t is a high-dimensional vector with strong dependencies between its elements, like the pixels in an image or the frequency bins in a spectral representation of speech? In that case, it is important not just to model the dependencies between consecutive "frames" x_t but also to model the dependencies between the elements x_{ti} of each x_t.

We now shift our attention to this static aspect of modeling the interactions between different sensory elements (like pixels) co-occurring in time together in a way that carries important meaning for the learning agent. There have been many machine learning proposals to capture the joint distribution of a set of joint observations (such as the pixels in an image, i.e., the elements of the vector x associated with a particular static input), and it remains a very active field of research where more investigation is needed. Indeed whereas deep supervised learning has been extremely successful in AI applications such as speech recognition, computer vision and natural language processing (see LeCun *et al.* (2015) for a recent review), deep learning of unsupervised models remains challenging but needed to handle the large quantities of unlabeled data the world has to offer.

4.1. Neural computation does inference: going down the energy

We consider the hypothesis that a central interpretation of neural computation (what neurons do, on a short time scale at which weights can be considered fixed) is that it consists in performing *iterative inference*. Iterative inference means that the hidden units h of the network are gradually changed towards configurations that are more probable, with the given sensory input x and according to the current "model of the world" associated with the parameters of the model. In other words, they are approximately moving towards configurations more probable under $P(h|x)$, and eventually sampling from $P(h|x)$. With this in mind, $R(\tilde{s}_t)$ in Eq. 6 represents a guess for a new configuration, with $R(\tilde{s}_t) - s_t$ a noisy direction of movement. A noise-free direction would be $R(s_t) - s_t$, but injecting noise is known to be important in order to find not just a single local mode of $P(h|x)$ but explore the full distribution.

We now draw an interesting link between this observation and recent work on unsupervised learning using denoising auto-encoders and denoising score matching (Vincent, 2011; Alain and Bengio, 2013). If $R(s)$ is the linear combination of input rates $\rho(s)$, the above papers make a link between $R(s) - s$ and the energy of a probabilistic model $P(s) \propto e^{-E(s)}$ with energy function $E(s)$, i.e., they find that

$$R(s) - s \propto \frac{\partial \log P(s)}{\partial s} = -\frac{\partial E(s)}{\partial s}. \tag{13}$$

With this interpretation, the leaky integration neural computation of Eq. 6 seems to follow a Langevin Monte-Carlo Markov chain (Andrieu *et al.*, 2003):

$$\begin{aligned} s_{t+1} &= s_t + \epsilon(R(\tilde{s}_t) - s_t) \\ &= s_t + \epsilon(R(\tilde{s}_t) - \tilde{s}_t + \tilde{s}_t - s_t) \\ &= s_t + \epsilon(-\frac{\partial E(\tilde{s}_t)}{\partial \tilde{s}_t} + \eta_t) \end{aligned} \tag{14}$$

where for the last line we used Eqs. 13, 7, and we see that we are going down the gradient from $\tilde{s}_t$. Hence from the point of view of the noisy states $\tilde{s}$, we see that the update equation corresponds to

$$\begin{aligned} \tilde{s}_{t+1} - \eta_{t+1} &= \tilde{s}_t - \eta_t - \epsilon \frac{\partial E(\tilde{s}_t)}{\partial \tilde{s}_t} + \epsilon \eta_t \\ \tilde{s}_{t+1} &= \tilde{s}_t - \epsilon \frac{\partial E(\tilde{s}_t)}{\partial \tilde{s}_t} + \eta_{t+1} - (1-\epsilon)\eta_t \end{aligned} \tag{15}$$

which we recognize as going down the gradient of the energy with "learning rate" ϵ and adding "noise" $\eta_{t+1} - (1 - \epsilon)\eta_t$.

4.2. A neurally motivated energy function

An important missing ingredient is condition 2 (Eq. 8), which depends on the specific choice of parametrization for the energy function. We would like this condition to be satisfied, while yielding a neural computation that roughly approximates real neural computation, and R corresponding to the derivative of the energy as per Eq. 13. Towards that objective, we propose the following energy function:

$$E(s) = \sum_i \frac{s_i^2}{2} - \sum_{i<j} W_{i,j}\rho(s_i)\rho(s_j) - \sum_i b_i \rho(s_i) \ . \tag{16}$$

It yields the following neural update:

$$\begin{aligned} s_{t+1,i} &= (1-\epsilon)s_{t,i} + \epsilon(\rho'(\tilde{s}_{t,i})(b_i + \sum_j W_{i,j}\rho(\tilde{s}_{t,j})) - \tilde{s}_{t,i}) \\ &= (1-2\epsilon)s_{t,i} + \epsilon(\rho'(\tilde{s}_{t,i})(b_i + \sum_j W_{i,j}\rho(\tilde{s}_{t,j})) - \eta_{t,i}) \end{aligned} \tag{17}$$

where we recognize three terms: (1) the leaky integration going exponentially towards zero, (2) the external input as a weighted sum of the input firing rates, scaled by the non-linearity derivative ρ', and (3) added noise. If we choose ρ to be linear by part, as follows, then ρ' is either 1 (in the active region) or 0 (in the saturation regions):

$$\rho(v) = \begin{cases} 0 & \text{if } v < \beta_1 \\ v - \beta_1 & \text{if } \beta_1 \leq v \leq \beta_2 \\ 1 & \text{if } v > \beta_2, \end{cases} \tag{18}$$

with thresholds β_1 and β_2 such that $\beta_2 - \beta_1 = 1$ and $\beta_1 < 0 < \beta_2$. In the saturation regions the neuron ignores the external input and goes exponentially to 0, thus coming back into the active region. Importantly, we have satisfied condition 2, and thus condition 1.

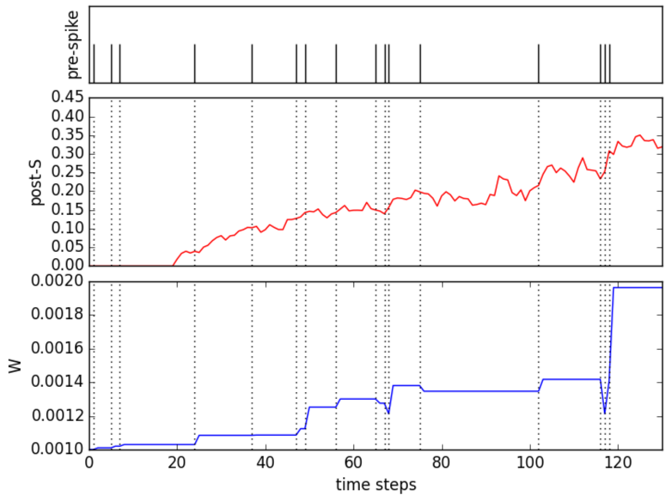

Figure 3. Example of rate and spike sequence generated in the simulations, along with weight changes according to the spike-dependent variant of our update rule, Eq. 2. Top: pre-synaptic spikes ξ_j (which is when a weight change can occur). Middle: integrated post-synaptic activity s_j. Bottom: value of the updated weight $W_{i,j}$.

5. Simulation results

5.1. Method

We simulate random variations in a presynaptic neural firing rate $\rho(s_j)$ as well as random variations in the postsynaptic firing rates $\rho(s_i)$ induced by an externally driven voltage. By exploring many configurations of variations and levels at pre- and postsynaptic sides, we hope to cover the possible natural variations. We generate and record pre- and postsynaptic spikes sampled according to a binomial at each discrete t with probability proportional to $\rho(s_i)$ and $\rho(s_j)$ respectively, and record $\dot{s}_i$ as well, in order to implement either a classical nearest neighbor STDP update rule or Eq. 1, which is SGD in J_{STDP} (Eq. 3). Python scripts for those simulations are available at `http://www.iro.umontreal.ca/~bengioy/src/STDP_simulations`. The nearest neighbor STDP rule is as follows. For every presynaptic spike, we consider a window of 20 time steps before and after. If there is one or more postsynaptic spike in both left and right windows, or no postsynaptic spike at all, the weight is not updated. Otherwise, we measure the time difference between the closest postsynaptic spike (nearest neighbor) and the presynaptic spike. If both spikes coincide, we make no weight change. To compute the appropriate averages, 500 random sequences of rates are generated, each of length 160 time steps, and 1000 randomly sampled spike trains are generated according to these rates.

For measuring the effect of weight changes, we measure the average squared rate of change $E[||\dot{s}_i^2||]$ in two conditions: with weight changes (according to Eq. 1), and without.

5.2. Results

Examples of the spike sequences and underlying pre- and postsynaptic states s_i and s_j are illustrated in Fig. 3. Fig. 1 (middle and right) shows the results of these simulations, comparing the weight change obtained at various spike timing differences for Eq. 1 and for the nearest neighbor STDP rule, both matching well the biological data (Fig. 1, left). Fig. 2 shows that both update rules are strongly correlated, in the sense that for a given amount of weight change induced by one, we observe in average a linearly proportional weight change by the other.

6. Connection to Score Matching and MCMC EM

Let us first decompose our objective function into terms for the clamped neurons x and terms of the unclamped neurons h, i.e., $J_{\text{STDP}} = J_h + J_x$, where

$$\begin{aligned} J_h &= \frac{1}{2}||h_{t+1} - f_h(s_{t-1}, \eta_{t-1})||^2 \\ J_x &= \frac{1}{2}||x - f_x(s_{t-1}, \eta_{t-1})||^2 \end{aligned} \tag{19}$$

6.1. J_x as a score matching criterion

With the interpretation of $R(s)$ as the target towards which neurons want to move in order to reduce the energy of an energy-based model (Eq. 13 and the rest of Section 4.1), we have

$$f(s_{t-1}, \eta_{t-1}) = s_{t-1} - \epsilon \frac{\partial E(s_{t-1})}{\partial s_{t-1}} + \text{noise}. \qquad (20)$$

Ignoring the noise injection, we can rewrite our objective function in terms of the energy gradient:

$$\begin{aligned} J_{\text{STDP}} &= \frac{1}{2} || s_{t+1} - f(s_{t-1}, \eta_{t-1}) ||^2 \\ &\approx \frac{1}{2} || s_{t+1} - s_{t-1} + \epsilon \frac{\partial E(s_{t-1})}{\partial s_{t-1}} ||^2. \end{aligned} \qquad (21)$$

Let us first consider the terms associated with the clamped units, x, for which $x_{t+1} = x_{t-1} = x$, i.e.,

$$J_x \approx \frac{\epsilon^2}{2} || \frac{\partial E(x, h_{t-1})}{\partial x} ||^2. \qquad (22)$$

where the approximation is only due to ignoring the effect of the noise.

One might recognize that this is one term of the standard score matching criterion J_{SM} (see Theorem 1 in (Hyvärinen, 2005))

$$J_{\text{SM}} = \frac{1}{2} || \frac{\partial \text{E}(x)}{\partial x} ||^2 + \sum_i \frac{\partial^2 \text{E}(x)}{\partial x_i^2} \qquad (23)$$

where $\text{E}(x)$ represents an energy function on x, and in our case we consider h given, i.e.,

$$\text{E}(x) = E(x, h).$$

What about the second derivative term? It can be ignored so long as the diagonal second derivatives of the energy function do not depend on the parameters, i.e.,

$$\frac{\partial^3 E(x, h)}{\partial x_i^2 \partial \theta} = 0 \qquad (24)$$

which we call **condition 3**. It turns out that this condition is satisfied by our neural energy function, Eq. 16. Hence we obtain that, given h, *minimizing J_x is equivalent to minimizing the score matching criterion on x.*

6.2. J_h to speed-up inference and as a score matching criterion

What about the h part of our objective function? We find two interesting interpretations for it and we then consider the big picture with inference included as a form of the EM algorithm.

First, consider Eq. 21 in the case of h, i.e.,

$$J_h \approx \frac{1}{2} || h_{t+1} - h_{t-1} + \epsilon \frac{\partial E(x, h_{t-1})}{\partial h_{t-1}} ||^2.$$

Because of Eq. 20 and because

$$h_t = f_h(s_{t-1}, \eta_{t-1})$$

we obtain that

$$\frac{h_t - h_{t-1}}{\epsilon} = - \frac{\partial E(x, h_{t-1})}{\partial h_{t-1}}$$

and

$$\frac{h_{t+1} - h_{t-1}}{\epsilon} = - \frac{\partial E(x, h_t)}{\partial h_t} - \frac{\partial E(x, h_{t-1})}{\partial h_{t-1}},$$

i.e., J_h is trying to make the current score $-\frac{\partial E(x, h_{t-1})}{\partial h_{t-1}}$ match sum of the current score and the next score, i.e., it is trying to change the parameters so that after the next update, we will reach a local minimum of the energy, where $\frac{\partial E(x, h_t)}{\partial h_t} = 0$. Another way to see it is that it is trying to *speed up the Langevin Markov chain by making a single step do the work of two steps.* Both arguments suggest that this criterion is trying to make the Markov chain converge faster to its stationary distribution (where, in average over the random moves, the score is 0).

Second, let us consider what happens when the distribution of h has converged near a fixed point where $h_{t+1} \approx h_{t-1}$, i.e., after inference: at that point the training objective becomes similar to what we have found for x, i.e.,

$$J_h \approx \frac{\epsilon^2}{2} || \frac{\partial E(x, h_{t-1})}{\partial h_{t-1}} ||^2$$

which corresponds to the score matching objective, with the inferred h as the "target", i.e., the region where the model will try to put a minimum of the energy.

6.3. Impatient MCMC-EM Score Matching

Note that the standard score matching algorithm is a consistent estimator for learning an energy-based probabilistic model *without latent variables*. Here we have latent variables, and we have seen that our predictive objective function J_{STDP} approximates a score matching objective on the joint of h and x, with x clamped to the observed values and h the result of samples along the path of the Langevin MCMC (Eq. 15).

This is clearly reminiscent of an MCMC EM procedure, i.e., we use an MCMC to approximately sample $h \sim P(h|x)$ (i.e., the E-step of the EM algorithm, executed by Monte-Carlo) and we use the resulting (x, h) pairs as if they were "fully visible data" for our learning procedure. Besides, instead of using the usual maximum likelihood update for the M-step of EM, we have a score matching update. This therefore constitutes an extension of score matching to the realm of models with latent variable, using an MCMC-EM formulation to deal with the latent variables. We call it *MCMC-EM score matching*.

Finally, there is another difference between the STDP updates and a standard application of the MCMC-EM ideas to score matching. Instead of waiting for MCMC inference to converge to the stationary distribution of the Markov chain, we perform parameter updates *after every step of inference*. We call this *Impatient MCMC-EM score matching*. As the number of inference steps increases and the burn-in period becomes small compared to the length of the sequence, the impatient MCMC-EM score matching update become the same, in average, as those for the regular MCMC-EM score matching. However, there may also be advantage to these impatient updates. As argued in the previous section, it should encourage the parameters to move in a way that would *make the inference converge faster*. This might be particularly important for a biological agent that has finite computational resources and not enough time between input events of interest.

7. Fixed Point Behavior

With the above definition of the energy function and the neural non-linearity, a fixed point of the deterministic version of the iterative inference update gives rise to an equation that looks even more like the traditional neural computation. Indeed, at a fixed point or a local minimum of the energy, we have

$$\frac{\partial E(s)}{\partial s} = 0 \Rightarrow R(s) = s \Rightarrow f(s, 0) = s \qquad (25)$$

which means that

$$s_i = \rho'(s_i)(b_i + \sum_j W_{i,j}\rho(s_j)). \qquad (26)$$

We have two possible values for $\rho'(s_i)$, so let us consider them both. Let us consider the hypothesis where $\rho'(s_i) = 0$, i.e., the unit is positively or negatively saturated. In that case, $R_i(s) = 0$ the neural update becomes

$$s_{t+1,i} = (1 - \epsilon)s_{t,i}$$

which *converges towards* $s_i = 0$. Since we have defined $\beta_1 < 0 < \beta_2$, and since $\rho'(s_i) = 0$ only when s_i is *outside of the interval* (β_1, β_2), we get a contradiction: **when** $\rho'(s_i) = 0$**, the network cannot be at a fixed point**. A global fixed point of the noise-free dynamics thus corresponds to a solution satisfying

$$s = b + W\rho(s)$$

which corresponds to the usual weighted sum of non-linear activations neural computation. In practice the network will probably not reach such a fixed point but *approach a fixed point*. A stable attractor of the deterministic version of the update will exist if and only if the weights are small enough, more precisely if the determinant of the Jacobian $R'(s)$ is less than 1.

8. Link to Back-propagation and Target-propagation

Consider what happens when a network such as described above sits near equilibrium, i.e., near a fixed point as per Eq. 25. At that point, as per that equation, the average gradient of the energy is 0 and weight updates are also 0 in average.

Now suppose that the external input into some "visible" unit x changed, being influenced by external sensors to move towards some target value X. Note that because we assume that neurons are temporal integrators, it is not the state of x that is being clamped, but instead its external input X that is being clamped. When that happens, the actual state of x will *gradually change towards the externally provided target* X, and we will denote $\Delta x = \epsilon(X - x)$ that initial change of x from time 0 to time 1 (following the neural update equation 6, but with $R_x(s)$ replaced by X). That would push the global state s away from the equilibrium where it was sitting, and into a region where $\frac{\partial E(s)}{\partial s}$ is non-zero. Initially, the only part of the objective function which would be non-zero would be due to the prediction error at x, i.e., corresponding to the mismatch between the prediction $R_x(s)$ and the clamped state x,

$$J = ||R_x(s) - (x + \Delta x)||^2 = ||\Delta x||^2 = \epsilon^2 ||X - x||^2 \quad (27)$$

because at equilibrium $R_x(s) = x$. Now, how would the rest of the network react to this external perturbation? Each hidden neuron would approximately move in the direction of the gradient of J, but only those (call them h_1) that are directly connected to x would initially feel the pressure to minimize J. That perturbation (in the form of a volley of additional spikes, for real neurons) would then travel to the next circle of neurons, those directly connected to h_1 but not to x, etc.

Let us look at this in more detail. Consider a typical multi-layer architecture with connections between x and h_1, between h_1 and h_2, etc. The change Δx would propagate to h_1 via the neural update, which, when we ignore the effect of the injected noise, would yield a change in h_1

$$\Delta h_1 = \epsilon(R_{h_1}((x + \Delta x, h)) - h_1).$$

With Δx small (arising out of our assumption that the visible units only gradually move towards their target), we can approximate the above by taking the Taylor expansion of R around x,

$$R_{h_1}((x + \Delta x, h)) = h_1 + \frac{\partial R_{h_1}((x, h))}{\partial x}\Delta x + o(\Delta x)$$

exploiting $R_{h_1}((x, h)) = h_1$ at the fixed point, yielding

$$\Delta h_1 = \epsilon \frac{\partial R_{h_1}(s)}{\partial x}\Delta x + o(\Delta x).$$

Note that

$$\Delta x = 2\epsilon^2 \frac{\partial J}{\partial x}$$

where J (Eq. 27) is the error we are getting immediately after x changed to $x + \Delta x$, hence we have that

$$\Delta h_1 = 2\epsilon^3 \frac{\partial R_{h_1}(s)}{\partial x} \frac{\partial J}{\partial x} + o(\Delta x). \qquad (28)$$

In order to obtain backprop, what we would like to get, though is not $\frac{\partial R_{h_1}(s)}{\partial x}\Delta x$ but $\frac{\partial R_x(s)}{\partial h_1}\Delta x$ since that would correspond to an application of the chain rule and we would have $\Delta h_1 \propto \frac{\partial J}{\partial h_1}$. The good news is that this equality is true because R is a first derivative of a function related to the energy function:

$$R(s) = s - \frac{\partial E(s)}{\partial s} = \frac{\partial L(s)}{\partial s}$$

where

$$L(s) = \frac{1}{2}||s||^2 - E(s)$$

and $R_x(s) = \frac{\partial L(s)}{\partial x}$, $R_{h_1}(s) = \frac{\partial L(s)}{\partial h_1}$, so that $\frac{\partial R_{h_1}(s)}{\partial x}$ and $\frac{\partial R_x(s)}{\partial h_1}$ are cross-derivatives of L. As we know that cross derivatives are symmetric,

$$\frac{\partial R_{h_1}(s)}{\partial x} = \frac{\partial^2 L}{\partial x \partial h_1} = \left(\frac{\partial^2 L}{\partial h_1 \partial x}\right)^T = \frac{\partial R_x(s)}{\partial h_1}^T. \qquad (29)$$

Now note that since at the fixed point $x = R_x(s)$

$$\frac{\partial R_x(s)}{\partial h_1} = \frac{\partial x}{\partial h_1}$$

and we are ready to exploit that to rewrite Δh_1 (Eq. 28) in a form that equates it with a backpropagated gradient:

$$\begin{aligned}
\Delta h_1 &= \epsilon \frac{\partial R_x(s)}{\partial h_1}^T \Delta x + o(\Delta x) \\
&= 2\epsilon^3 \frac{\partial R_x(s)}{\partial h_1}^T \frac{\partial J}{\partial x} + o(\Delta x) \\
&= 2\epsilon^3 \frac{\partial x}{\partial h_1}^T \frac{\partial J}{\partial x} + o(\Delta x) \\
&= 2\epsilon^3 \frac{\partial J}{\partial h_1} + o(\Delta x) \\
&= 2\epsilon^3 \frac{\partial ||X - x||^2}{\partial h_1} + o(\Delta x) \qquad (30)
\end{aligned}$$

Similarly, the perturbation Δh_1 will be transmitted at the next time step to the units h_2 that are directly connected to h_1 (but not to x), and yield

$$\begin{aligned}
\Delta h_2 &= \epsilon \frac{\partial h_1}{\partial h_2}^T \Delta h_1 + o(\Delta x) \\
\Delta h_2 &= 2\epsilon^4 \frac{\partial h_1}{\partial h_2}^T \frac{\partial ||X - x||^2}{\partial h_1} + o(\Delta x) \\
\Delta h_2 &= 2\epsilon^4 \frac{\partial ||X - x||^2}{\partial h_2} + o(\Delta x). \qquad (31)
\end{aligned}$$

Furthermore, the initial change in weights will follow the back-propagation update, since it corresponds to $\Delta h \frac{\partial h}{\partial W}$, except for the factors ϵ^{k+2} for units h_k at layer k, that make the initial changes much slower for the more remote layers. This is because the leaky integration neurons have not had time to integrate the information yet, so practically it will take on the order of the time constant times k for the change in h_k to become significant, unless we adjust the per-layer learning rates accordingly.

What about the case of inputs x and outputs y? A situation similar to the above would take place if we first clamp x and let the network settle to an equilibrium, and then clamp the external input into y to some target value Y, producing a Δy that propagated into the hidden layers and induces changes towards reducing $||\Delta y||^2 = \epsilon^2 ||Y - y||^2$.

Although we see that the proposed neural dynamics and weight updates will behave approximately like back-propagation, there are differences, especially when we consider what happens after more time steps. But maybe the most important take-home message from this link with back-propagation is the following. We know that back-propagation works extremely well to train both supervised and unsupervised networks. We see here that back-propagation essentially corresponds to a variational update when the inference is infinitesimal, i.e., we only allow a single step of inference corresponding to small moves in the direction of reducing the energy function. This is very encouraging, because it suggests that the proposed machine learning framework can work well even when the approximate inference is very inexpensive, not requiring a large number of inference iterations to achieve useful weight updates.

9. Related work, contributions and future work

The idea of neural computation corresponding to a form of stochastic relaxation towards lower energy configurations is of course very old, e.g., with the Boltzmann machine (Hinton and Sejnowski, 1986) and its Gibbs sampling procedure. More recently, see also (Berkes *et al.*, 2011). What differs here from the Boltzmann machine is that we consider the state space to be continuous (associated with the expected voltage potential, integrating out the random effects due to spikes), rather than discrete, and that we consider very small steps (going down the gradient of the energy), which is more like a Langevin MCMC, rather than allowing each neuron to jump to its optimal state (plus noise), given its neighbors configuration, which is what Gibbs sampling does.

The closest work is probably that of Xie and Seung (2000), which we have already discussed. Notable additions to this work include demonstrating that the spike-timing to weight change relationship is a *consequence* of Eq. 1, rather than the other way around (and a small difference in the use of ds/dt vs $d\rho(s)/dt$). But most importantly, the proposed update rule is SGD on a predictive objective function, J_{STDP}.

There are of course many other papers on theoretical interpretations of STDP, and the reader can find many references in Markram *et al.* (2012), but more work is needed to explore the connection of STDP to an unsupervised learning objective that could be used to train not just a single layer network (like PCA and traditional Hebbian updates) but also a deep unsupervised model. Many approaches (Fiete and Seung, 2006; Rezende and Gerstner, 2014) rely on variants of the REINFORCE algorithm (Williams, 1992) to estimate the gradient of a global objective function (basically by correlating stochastic variations at each neuron with the changes in the global objective). Although this principle is simple, it is not clear that it will scale to very large networks due to the linear growth of the variance of the estimator with the number of neurons. It is therefore tempting to explore other avenues, and we hope that the building blocks introduced here and the links made with energy-based approaches with variational inference for unsupervised learning can form useful material for a more efficient unsupervised learning principle for deep networks that is also consistent with STDP.

A practical outcome of this work is a prediction that synaptic weight changes should vanish when the postsynaptic firing rate remains constant, as seen by inspection of Eq. 1. It would clearly be interesting to test this prediction in actual biological experiments.

Much remains to be done to obtain a complete probabilistic theory of unsupervised learning that is consistent with STDP, but we believe that we have put interesting ingredients in place. One aspect that requires a lot more development is how the proposed STDP objective function helps to fit the sensory observations x. If, as hypothesized above, neural computation is approximately doing inference (e.g. Langevin MCMC), then each step of inference, in average, brings us towards an equally likely or even more likely configuration of h, given x, according to the model. Hence each step is approximately pointing down the energy of $P(h|x)$. Now, in an EM or variational EM context such as discussed in Neal and Hinton (1999); Kingma and Welling (2014); Bengio *et al.* (2015), with x fixed, the distribution we want to model and consider as a target towards which parameters should be updated is precisely the joint of $h \sim P(h|x)$ and $x \sim$ the training data, which we now call $Q(h, x)$ (the inference distribution), following the above papers. By minimizing a predictive criterion such as J_{STDP}, we conjecture that the model parameters move in the direction that makes the model more consistent with $Q(h, x)$, which is what is required to maximize the variational EM bound on the data likelihood $P(x)$. The idea is that we change the inference process so that it would reach its final state faster, which corresponds to a configuration of h that fits well the observed x.

References

Alain, G. and Bengio, Y. (2013). What regularized auto-encoders learn from the data generating distribution. In *ICLR'2013*. also arXiv report 1211.4246.

Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. (2003). An introduction to MCMC for machine learning. *Machine Learning*, **50**, 5–43.

Bengio, Y., Lee, D.-H., Bornschein, J., and Lin, Z. (2015). Towards biologically plausible deep learning. arXiv:1502.04156.

Berkes, P., Orban, G., Lengyel, M., and Fiser, J. (2011). Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science*, **331**, 83—87.

Bi, G. and Poo, M. (2001). Synaptic modification by correlated activity: Hebb's postulate revisited. *Annu. Rev. Neurosci.*, **24**, 139—166.

Fiete, I. R. and Seung, H. S. (2006). Gradient learning in spiking neural networks by dynamic perturbations of conductances. *Physical Review Letters*, **97**(4).

Friston, K. J. and Stephan, K. E. (2007). Free-energy and the brain. *Synthese*, **159**, 417—458.

Gerstner, W., Kempter, R., van Hemmen, J., and Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding. *Nature*, **386**, 76–78.

Hinton, G. E. (2007). How to do backpropagation in a brain. Invited talk at the NIPS'2007 Deep Learning Workshop.

Hinton, G. E. and Sejnowski, T. J. (1986). Learning and relearning in Boltzmann machines. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*, pages 282–317. MIT Press, Cambridge, MA.

Hyvärinen, A. (2005). Estimation of non-normalized statistical models using score matching. *J. Machine Learning Res.*, **6**.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, **521**, 436–444.

Markram, H. and Sakmann, B. (1995). Action potentials propagating back into dendrites triggers changes in efficacy. *Soc. Neurosci. Abs*, **21**.

Markram, H., Gerstner, W., and Sjöström, P. (2012). Spike-timing-dependent plasticity: A comprehensive overview. *Frontiers in synaptic plasticity*, **4**(2).

Neal, R. and Hinton, G. (1999). A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*. MIT Press, Cambridge, MA.

Rezende, D. J. and Gerstner, W. (2014). Stochastic variational learning in recurrent spiking networks. *Frontiers in Computational Neuroscience*, **8**(38).

Shepherd, G. M. (2003). *The synaptic organization of the brain*. Oxford University Press.

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, **23**(7).

Williams, R. J. (1992). Simple statistical gradient-following algorithms connectionist reinforcement learning. *Machine Learning*, **8**, 229–256.

Xie, X. and Seung, H. S. (2000). Spike-based learning rules and stabilization of persistent neural activity. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 199–208. MIT Press.