Neural Computation
21, in press 2009

On the maximization of information flow between spiking neurons

Lucas C. Parra
Biomedical Engineering Department, City College of New York, New York, NY 10033
parra@ccny.cuny.edu

Jeffrey M. Beck
Brain and Cognitive Sciences, University of Rochester, Rochester, NY 14627, jbeck@bcs.rochester.edu

Anthony J. Bell
Helen Wills Neuroscience Institute, University of California, Berkeley, CA 94720, tbell@berkeley.edu

A feed-forward spiking network represents a non-linear transformation that maps a set of input spikes to a set of output spikes. This mapping transforms the joint probability distribution of incoming spikes into a joint distribution of output spikes. We present an algorithm for synaptic adaptation that aims to maximize the entropy of this output distribution thereby creating a model for the joint distribution of the incoming point processes. The learning rule that is derived depends on the precise pre- and post-synaptic spike timings. When trained on correlated spike trains the network learns to extract independent spike trains thereby uncovering the underlying statistical structure and creating a more efficient representation of the incoming spike trains.

Introduction

Imagine that the goal of a biological neural network is to maximize the information it communicates about its inputs. If the transformation is deterministic then maximizing the information transmission is equivalent to maximizing the entropy of the outputs. This problem has already been solved for abstract rate models where information is encoded in the firing rate of a neuron for feedforward analog networks and leads to standard independent component analysis (Anthony & Terrence, 1995). Here we develop this idea for more realistic networks of spiking neurons which can in principle encode information either in the precise times of spikes or via a population code utilizing firing rates. That is, for a set of input spike patterns we address the question of how to manipulate connection strengths between deterministic spiking neurons so as to generate output spike trains with the maximum entropy. In particular, detailed learning rules are derived for the general Spike Response Model (SRM) (Gerstner & Kistler, 2002), a generalization of the Integrate-and-Fire neuron. The analysis is restricted to a single layer feedforward network and relies on the assumption that neurons fire at a rate which is relatively low compared to the time constant of the refractory periods of the neurons. It differs from previous information theoretic models in that it considers a network of deterministic neurons rather than a single isolated stochastic neuron (Bohte & Mozer, 2005; Toyoizumi, Pfister, Aihara, & Gerstner, 2005). A brief account of our model appeared in conference form focusing on the sensitivity of spike timing (Bell & Parra, 2005). Here we establish the connection between increasing sensitivity and maximizing information flow, and demonstrate how this criterion can be used to extract common spike pattern from a mixture, as well as re-code a correlated population code into a more efficient representation.

Rather than maximizing output entropy directly we propose to maximize the likelihood of the input spike-times under the assumption that the output follows the maximum entropy distribution for a given rate, i.e. the output is a Poisson process of fixed rate. This Maximum Likelihood (ML) approach maximizes the information transfer in the network, and has the additional advantage of controlling the rate of firing, and thus energy consumption, of the output neurons. The resulting algorithm functions by maximizing the sensitivity of the output spike train to variations in the input spike train subject to limitations on channel capacity.

To demonstrate the generality and utility of this algorithm we then show that it is capable of both *demultiplexing* a spike timing code as well as recoding a correlated spatio-temporal population code so as to make it more efficient. In the first case, each input synapse carries a superposition of several independent point processes (sources), and, by analogy to blind source separation, the network learns to extract these sources and send each to a different output neuron. In the second test case, input spikes are generated by a set of inhomogeneous Poisson processes which respond to the position of a particle in motion. This leads to input spike trains which are highly correlated and thus redundant. In this case learning decorrelates these inputs and creates a sparse representation of position.

In both cases, we note that the objective function for learning seeks to maximize the sensitivity of neurons to the spike-timings of their inputs. This is consistent with increasing evidence that learning in real neurons is highly sensitive to the relative timings of input and output spikes (Dan & Poo, 2004) via Spike Timing Dependent Plasticity (STDP) learning rules. Unfortunately, our result does not replicate all of the empirical data, but seems only to capture the causal halve of the STDP learning curve. This issue is addressed in the Discussion section.

This work gives a rigorous derivation for how to maximize spike-timing entropy in a deterministic spiking network, and shows in particular how to maximize the sensitivity of a point process obtained from a threshold crossing mechanism. However, to facilitate the reading of the main text a large portion of this material has been relegated to the Appendix.

Likelihood of a Spike Train

A deterministic network of spiking neurons such as a single-layer feedforward network of Integrate-and-Fire neurons can be thought of as a transformation that maps a set of input spike trains into a new set of output spike trains. The spike trains of all the input neurons can be represented by the firing times and the identities of the neurons firing. We will denote these variables as $\mathbf{t}$ and $\mathbf{i}$ – a pair of vectors each with one element per spike, so that the time of the kth spike is t_k occuring in neuron i_k. Thus, the timing vector is composed of real numbers, and the neural identities vector is composed of integers which represent neuron indices. For a deterministic mapping of spike trains, $f : \mathbf{t},\mathbf{i} \rightarrow \mathbf{t}',\mathbf{i}'$, the likelihood of the output spikes $p(\mathbf{t}',\mathbf{i}')$ is given in terms of the likelihood of the input spikes $p(\mathbf{t},\mathbf{i})$ as (see Appendix A):

$$p(\mathbf{t}',\mathbf{i}') \propto \sum_{(\mathbf{t},\mathbf{i})\in S(\mathbf{t}',\mathbf{i}')} |\mathbf{T}^T\mathbf{T}|^{-1/2} p(\mathbf{t},\mathbf{i})\,. \tag{1}$$

The sum in $(\mathbf{t},\ \mathbf{i})$ extends over all the possible inputs that lead to a specific output $(\mathbf{t}',\mathbf{i}')$, i.e. the set of solutions $S(\mathbf{t}',\mathbf{i}') = \{\mathbf{t},\mathbf{i} | (\mathbf{t}',\mathbf{i}') = f(\mathbf{t},\mathbf{i})\}$. Matrix $\mathbf{T}$ captures the sensitivity of output spike-times vs. input spike-times for a given set of input and output neurons $\mathbf{i},\mathbf{i}'$, and is thus a Jacobian matrix given by

$$\mathbf{T} = \frac{\partial \mathbf{t}'}{\partial \mathbf{t}^T}\,. \tag{2}$$

For a network of noiseless Spike Response Model (SRM) neurons (Gerstner & Kistler, 2002) this sensitivity matrix is derived in Appendix C. In the case of an over-complete transformation, where the number of input spikes generates a larger number of output spikes, this map is likely to be invertible such that the observed output could only have been generated by one specific input. In this case the sum contains a single term giving an expression equivalent to what was derived in (Shriki, Sompolinsky, & Lee, 2000):

$$p(\mathbf{t}',\mathbf{i}') \propto |\mathbf{T}^T\mathbf{T}|^{-1/2} p(\mathbf{t},\mathbf{i}). \tag{3}$$

Since the elements of the matrix $\mathbf{T}^T\mathbf{T}$ grow with the number of spikes in the output layer, maximizing entropy directly using Equation (3) can result in a network which fires at very high rates. This is neither biologically plausible nor compatible with the low firing rate assumption which will be used to simplify the learning rule. Instead, we choose to minimize the Kullback-Leiber divergence between the distribution of the spikes observed in the output layer and maximum entropy distribution for a set of point process with a given rate, i.e. independent homogeneous Poisson processes. This is equivalent to maximizing the likelihood of the observed pattern of input spikes under the assumption that these spikes were generated by applying the inverse mapping $f^{-1} : (\mathbf{t}',\mathbf{i}') \rightarrow (\mathbf{t},\mathbf{i})$ to output spikes generated from an independent Poisson process with a given rate.

Specifically, if we define $q(\mathbf{t}',\mathbf{i}')$ as the probability density function of a selection of independent Poisson processes, then

$$q(\mathbf{t}',\mathbf{i}') = \prod_i q(n_i')\,, \tag{4}$$

where

$$q(n) = \frac{\lambda^n}{n!} e^{-\lambda}\,, \tag{5}$$

and n_i' is the spike count of the ith output neuron and λ is a free parameter which gives the desired mean firing rate of the neurons in the output layer (see Appendix B).

The objective function for learning is then given by

$$\begin{aligned} & -D_{\mathrm{KL}}\left(p(\mathbf{t}',\mathbf{i}')\,||\,q(\mathbf{t}',\mathbf{i}')\right) \\ = \ & \left\langle -\log p(\mathbf{t}',\mathbf{i}') + \log q(\mathbf{t}',\mathbf{i}')\right\rangle & (6) \\ = \ & \left\langle -\log p(\mathbf{t},\mathbf{i}) + \frac{1}{2}\log\left|\mathbf{T}^T\mathbf{T}\right| + \log q(\mathbf{t}',\mathbf{i}')\right\rangle\,. & (7) \end{aligned}$$

Now, the first term in this sum is simply the entropy of the input spike patterns and is independent of the parameters which map input to output spikes. Thus minimizing divergence requires only a consideration of the second and third terms of the equation directly above. Rearranging some terms, we obtain the log-likelihood of the input spike times under the maximum entropy model

$$\begin{aligned} L \equiv \langle \log p(\mathbf{t},\mathbf{i})\rangle \ &= \ D_{\mathrm{KL}}\left(p(\mathbf{t}',\mathbf{i}')\,||\,q(\mathbf{t}',\mathbf{i}')\right) \\ & \quad + \left\langle \frac{1}{2}\log\left|\mathbf{T}^T\mathbf{T}\right| + \log q(\mathbf{t}',\mathbf{i}')\right\rangle \\ & \geq \ \left\langle \frac{1}{2}\log\left|\mathbf{T}^T\mathbf{T}\right| + \log q(\mathbf{t}',\mathbf{i}')\right\rangle\,, \end{aligned} \tag{8}$$

where we have used the fact that the KL divergence is a positive definite quantity. This restates the well-known fact that minimizing KL divergence is equivalent to maximizing a lower bound on the log likelihood, L. In this case we are dealing with the likelihood of the observed input spike-times resulting (via the inverse map f^{-1}) from a set of independent Poisson processes of given rate.

The second term of the resulting objective function in Equation (7) captures the sensitivity of the likelihood on spike timing. Maximizing this term will make the output spike-times maximally sensitive to the timing of the input, and thus, insure that the temporal information contained in the input spike-times is transmitted optimally. The second term encourages Poisson distributed spikes in the output neurons and – as we will see in Section – leads to an Hebbian term that controls the overall spike rate.

To maximize this lower bound we derive now the gradient of these two terms with respect to the weights of the spiking network.

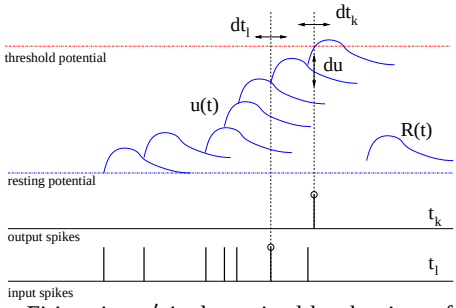

Figure 1. Firing time t'_k is determined by the time of threshold crossing. A change of an input spike-time dt_l affects, via a change of the membrane potential du the time of the output spike by dt'_k.

Spike Time Sensitivity

For two spikes.

We will now discuss how the timing of a postsynaptic (output) spike is affected by the timing of presynaptic (input) spike. Spike l, occurring in presynaptic neuron j_l, may have an effect on spike k, occurring in neuron i_k, if they are connected by a synapse with weights w_{ij}. Since we are primarily concerned with the dependency between different spikes we will adopt a notation that omits explicit neuron index: We use W_{kl}, the weight affecting output spike k due to input spike l, to be the corresponding $w_{ij} = w_{i_k j_l}$.

In the simplest version of the Spike Response Model (Gerstner & Kistler, 2002), spike l has an effect on spike k that depends on the time-course of the evoked EPSP or IPSP, which we write as $R_{kl}(t'_k - t_l)$. In general, this R_{kl} models both synaptic and dendritic linear responses to an input spike, and thus models synapse type and location. Note that, technically, R_{kl} includes refractory effects and is thus also a function of the time of the last spike of neuron k. However, since we ignore this effect for the purposes of learning, this dependence is not included explicitly here. See Appendix C for details. For learning, we need only consider the value of this function when an output spike, k, occurs.

In this model, depicted in Figure 1, a neuron adds up the post-synaptic potentials evoked by all input spikes until its membrane potential, $u_i(t)$, reaches threshold at time t'_k. This potential we will, again for convenience, write as $u_k \equiv u_i(t'_k, \{t_l\})$, and it is given by a sum over spikes l:

$$u_k = \sum_l W_{kl} R_{kl}(t'_k - t_l)\,. \tag{9}$$

This notation may seem unfamiliar, but it has the advantage that the sum over different input neurons and input spikes is combined into a single sum over spikes l, while avoiding cumbersome double indices. The notation exploits the definition, $W_{kl} = w_{i_k j_l}$, and is discussed in more detail in Section .

To compute the entries of the matrix $\mathbf{T}$ and then maximize timing sensitivity, we need to determine the effect of a small change in the input firing time t_l on the output firing time t'_k. (A related problem is discussed in (Banerjee, 2001).) When t_l is changed by a small amount dt_l the membrane potential will change as a result. This change in the membrane potential leads to a change in the time of threshold crossing dt'_k. The contribution to the membrane potential, du, due to dt_l is $(\partial u_k/\partial t_l)dt_l$, and the change in du corresponding to a change dt'_k is $(\partial u_k/\partial t'_k)dt'_k$. We can relate these two effects by noting that the total change of the membrane potential du has to vanish because u_k is defined as the potential at threshold. ie:

$$du = \frac{\partial u_k}{\partial t'_k} dt'_k + \frac{\partial u_k}{\partial t_l} dt_l = 0\,. \tag{10}$$

This is the *total differential* of the function $u_k = u(t'_k, \{t_l\})$, and is a special case of the implicit function theorem. Rearranging this:

$$\frac{dt'_k}{dt_l} = -\frac{\partial u_k}{\partial t_l} \Big/ \frac{\partial u_k}{\partial t'_k} = -W_{kl}\dot{R}_{kl}/\dot{u}_k\,. \tag{11}$$

For over-complete $N \to M$ spike mappings

To maximize the information transfer in a possibly over-complete mapping ($M \geq N$) for a given set of spikes, we must, according to (8) maximize the log-determinant of a Jacobian matrix, $\mathbf{T}^T\mathbf{T}$, where the entries of $\mathbf{T}$ are the timing dependencies $\mathbf{T}_{kl} \equiv \partial t'_k/\partial t_l$. The calculation of this gradient for the full Spike Response Model under the assumption of a low firing rate is in Appendix C. It yields a learning rule in which every interaction between a presynaptic spike at t_l and a postsynaptic spike at t'_k causes a weight change:

$$\Delta_1 W_{kl} \propto \frac{\partial \frac{1}{2}\log|\mathbf{T}^T\mathbf{T}|}{\partial W_{kl}} = \frac{\mathbf{T}_{kl}}{W_{kl}}\left([\mathbf{T}^{T\#}]_{kl} - [\mathbf{T}\mathbf{T}^{T\#}]_{kk}\right)\,, \tag{12}$$

where $\mathbf{T}^{T\#}$ represents the pseudo-inverse of $\mathbf{T}^T$. This is a non-local update involving a matrix inverse at each step. In the *infomax* case, such an inverse was removed by the Natural Gradient transform (see (Anthony & Terrence, 1995)), but in the spike timing case, this has turned out not to be possible, because of the complexity introduced into the $\mathbf{T}$ matrix by the $\dot{R}_{kl}$ term in (11).

Neuron index vs spike index

We have denoted the synaptic weights for each neuron pair as lower case w_{ij}, and the synaptic weights for each spike pair as upper case W_{kl}. These two matrices can be related by

$$\mathbf{W} = \mathbf{I}\mathbf{w}\mathbf{J}\,. \tag{13}$$

Matrix $\mathbf{I}$ has one column for each neuron i and one row for each output spike k with $I_{ki} = 1$ if spike k belongs to neuron i and 0 otherwise. Matrix $\mathbf{J}$ has one column for each input spike l and one row for each input neuron j with $J_{jl} = 1$ if spike l belongs to neuron j. With this notation the corresponding weight update for neuron pairs $\Delta_1\mathbf{w}$ can be computed from the gradient update for spike pairs $\Delta_1\mathbf{W}$ given in (12) with

$$\Delta_1\mathbf{w} = \mathbf{I}^T\Delta_1\mathbf{W}\mathbf{J}^T\,. \tag{14}$$

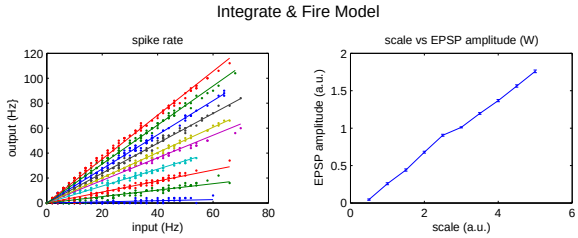

Figure 2. Output firing rate as function of input firing rate (left) and strength of EPSP or synaptic weight (right) for an Integrate-and-Fire neuron. Rates are determined by the number of spikes within a 500 ms observation window.

Spike Rate Sensitivity

When computing the gradients (12) the number of spikes is fixed. Updating synaptic weights will affect the timing of output spikes, but possibly also the number of output spikes. In this section we deal with this dependency explicitly, and derive an approximate expression for the gradient of the second term in Equation (8). Unfortunately, unlike for spike times there is no simple parameterized mapping from input spikes to output spike counts. However, Figure 2 demonstrates that a simple empirical relationship exists between the number of output spikes n within a time window as a function of the number of input spikes m and synaptic weights w for Spike Responce neurons of this type. Specifically, the spike rate n is well approximated by a linear relation in both m and w. Since the Integrate-and-Fire model combines the input from multiple neurons j additively it is fair to write

$$n_i \propto \sum_j w_{ij} m_j \tag{15}$$

so that the contribution to the gradient weight update due to the spike rate can be computed as

$$\Delta_2 w_{ij} \propto \frac{\partial \sum_i \log q(n_i)}{\partial w_{ij}} = -g(n_i) m_j \,, \tag{16}$$

where $g(n) = -\partial \log q(n)/\partial n$. Applying Stirling's approximation[1] of the factorial, $\log n! \approx n \log n - n$, this becomes $g(n_i) \approx \log(n_i/\lambda)$ and we obtain a simple local Hebbian learning rule reminiscent of the traditional *infomax* algorithm(Anthony & Terrence, 1995). Note that when $n_i > \lambda$ this gradient term is negative and n_i will be reduced. In the opposite case the term is positive and n_i will be increased. Therefore this term acts to maintain the spike rate at a chosen value λ.

Results

Demultiplexing spike trains

An interesting possibility in the brain is that specific 'patterns' are embedded in spatially distributed spike timings that are input to neurons. Several patterns could be embedded in single input trains. This is called *multiplexing*. To extract and propagate these patterns, the neurons must *demultiplex* these inputs using its threshold non-linearity. Demultiplexing is the 'point process' analog of the unmixing of independent inputs in Independent Component Analysis. We have been able to robustly achieve demultiplexing, as we now report.

We simulated a feed-forward network with 10 Integrate-and-Fire neurons and inputs from 10 presynaptic neurons. Time was discretized in $1ms$ bins (the SRM does not suffer from numerical problems associated with temporal resolution); firing threshold was set constant to: $\vartheta = 1$; post-synaptic potential was set to: $R(\tau,t) = R(t) = (1-\tau_s/\tau_m)^{-1}(\exp(-t/\tau_m) - \exp(-t/\tau_s))$, with $\tau_m = 20ms$, $\tau_s = 5ms$; and hyperpolarizing after potential to: $\eta(t) = -\exp(-t/\tau_r)$, with $\tau_r = 30ms$ (Gerstner & Kistler, 2002). Learning combines the gradients (12) and (16), $\Delta w_{ij} = \Delta_1 w_{ij} + \Delta_2 w_{ij}$, computed on the spikes generated during many sample intervals each of 500 ms in length. All time derivatives were computed numerically by differentiating neighboring time points. The parameter λ which controls the output spike rate was set to the average spike rate of the input. The network learns to demultiplex mixed spike trains which were generated randomly, as shown in Figure 3. For a small number of neurons (up to 5) this demultiplexing is a robust property of learning using the gradients (12) and (16).

However the final result of training is sensitive to the initialization of the weight matrix. As the number of neurons increases we find only partial demultiplexing. However, we obtain reliable results if the weight matrix is initialized, as in this example, with the transpose of the mixing matrix. This indicates that the training may be impeded by the existence of suboptimal local minima of the cost function. We have chosen this specific example as the correct result for more complex multiplexing is not easily visualized.

Population Recoding

In a second experiment, input spikes were generated from a population of independent Poisson neurons with Gaussian tuning curves which respond to the position of a particle, $s \in [0,10)$, moving on a periodic domain (i.e. a particle moving beyond $s = 10$ reappears at $s = 0$). The particle follows a Brownian motion with positive drift at a fixed velocity. The tuning curve of each input neuron has a preferred position which corresponds to the index of that neuron. Tuning curves are chosen to be sufficiently wide so that nearly simultaneous spikes in adjacent neurons are common. Since the position of the particle is correlated in time and space, strong positive correlations exist between the induced spike trains and many spikes seem to be redundant. See Figure 4. Application of the proposed learning rule with a target firing rate which is half the input rate yields a population code with sharper, more localized tuning curves. This is achieved by learning an asymmetric connectivity which strongly inhibits the firing of neurons with a preferred particle position through which the

[1] This implies that in our simulations weight updates according to (16) are only executed after a sufficiently large number of output spikes has accumulated.

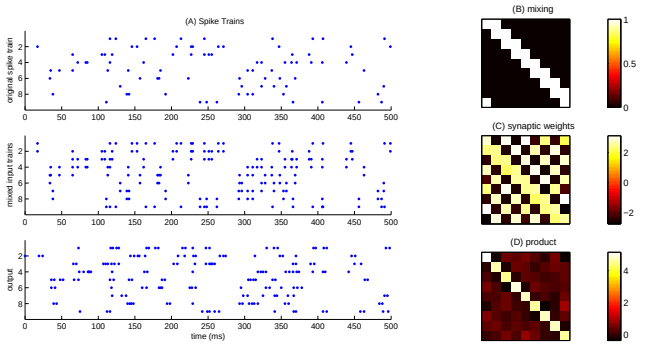

Figure 3. Unmixed spike trains. The input (center left) are 10 spike trains which are a mixture of 10 independent Poison processes (top left). The network unmixes the spike train to approximately recover the original. Bottom left panel show the recovered spike train. Notice that every original spike corresponds to a spike duple in the de-multiplexed output. The panels at the right show the mixing (top) the synaptic weight matrix after training (center) and the product of the two (bottom).

particle has already passed. Neurons with a preferred position which lie in the direction of the particle's motion are also inhibited – so as to reduce the overall firing rate of the output population – but to a much lesser extent.

This leads to a more efficient representation of particle position at any given time. Specifically, we found that linear Fisher information per spike, as estimated from the variance of the Local Optimal Linear Estimator (LOLE) (Shamir & Sompolinsky, 2001) applied to the entire spatio-temporal pattern of spikes, increased by nearly 30%. That is, with half as many spikes we were able to represent nearly two thirds of the input information about particle position. This is especially compelling since the learning rule was attempting to maximize information about spike timing and has no knowledge of particle position.

Spike Timing Dependent Plasticity

Finally, what about the spike-timing dependence of the observed learning? Does it match experimental results? The comparison is made in Figure 5, and the answer is no. There is a timing-dependent transition between depression and potentiation in our result in Figure 5B, but it is not a sharp transition like the experimental result in Figure 5A. In addition, it does not transition at zero (ie: when $t'_k - t_l = 0$), but at a time offset by the rise time of the EPSPs. The spike response model is inherently causal, a incoming spike can only affect a subsequent spike. The strength of this dependency is measured by the sensitivity matrix $\mathbf{T}$. The learning rule derived here modifies this dependency. The contributions to our weight update are as a result inherently causal. Current models that result in non-causal (synaptic adaptation for late spikes with $\Delta t < 0$) currently assume a probabilistic effect due to noisy spike generation (Bohte & Mozer, 2005; Toyoizumi et al., 2005). However, non-determinism in the spike generating mechanism is probably not the reason for the mismatch between our result and empirically observed

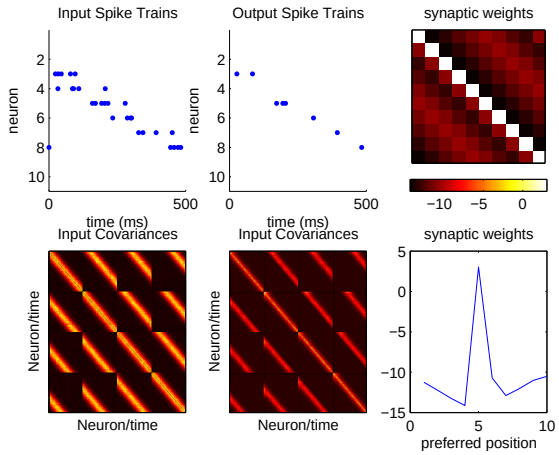

Figure 4. Population coding of the position of a particle moving on a periodic domain. (Top, left) The 10 input neurons respond by firing when the particle is within their receptive field. In this example the particle moves from the preferred position of the 3rd neuron to the preferred position of the 8th neuron within 500 ms. (Top, right) The synaptic weights after training with 1 h of data. Weight matrix is circulant due to the circular shift invariance of the problem. (Bottom, right) Weights show asymmetric lateral inhibition. (Top, center) The output spikes give a sparse representation of the particle trajectory omitting redundant spikes. (Bottom, left and center) Spatio-temporal correlation matrix for input and output neurons showing reduced correlation.

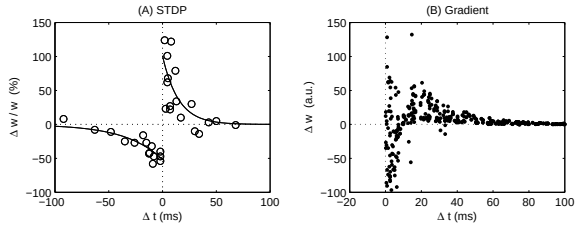

Figure 5. Dependence of synaptic modification on pre/post interspike interval. **Left (A):** Reproduced from Froemke & Dan, Nature (2002). Dependence of synaptic modification on pre/post interspike interval in cat L2/3 visual cortical pyramidal cells in slice. Naturalistic spike trains. Each point represents one experiment. **Right (B):** According to Equation (12) and (16). Each point corresponds to an spike pair between approximately 100 input and 100 output spikes.

STDP curves, as we discuss in the next section.

Discussion

In summary, we have started to explore, through thinking about timing sensitivity, how probabilistic learning of optimal information transfer can be achieved in spiking network models which are closer to known physiology. This holds the appealing promise of bringing network theories of representation and spike coding closer to biophysical theories of dendritic computation, through the infusion of ideas from unsupervised machine learning. We developed the theory for

a single-layer network of spiking neurons. With the derived learning rule, we were able to extract independent point processes from dependent mixtures, analogously to Independent Component Analysis (ICA) algorithms which do the same thing for real-valued processes.

Our results are relevant in terms of neural coding theory because they are the first to show how to use spikes to build a density model of spikes. Also, since arguably the most suggestive learning-based models of early cortical computation have been achieved with analogous rate-based network models (Olshausen & Field, 1997; Bell & Sejnowski, 1997), it is reasonable to expect similar advances in our understanding of the coding of sensory inputs when the formalism is extended to spiking neurons.

Our results and formalism may well turn out to be of general use for finding independent processes in data which is event-like, as opposed to continuous analog processes. Many complex systems and communication systems are of this nature, not just those in the brain. As our network is a probability density model, it or something similar could prove useful in the modeling of multivariate point-processes, a problem of more general interest in statistics.

We believe we have also taken a step to close the gap between known physiology and abstract learning principles. Physiological variables such as the shapes of EPSPs and post-spike repolarisation curves, even the placement of synapses in dendrites, are viewed here as parameters of a probabilistic model that could also be learned. In principle, anything that happens between input spikes and output spikes could adapt to fit the statistics of the input spikes. Another physiological variable, the voltage slope at threshold, has an important probabilistic interpretation in this framework. It controls the sensitivity to input timings (see (10)) and it is thus inversely proportional to the probability of the input spikes.

Despite these points, the complexity and non-locality of our learning rule and the difference between its synaptic changes and those observed in Spike Timing-Dependent Plasticity (STDP), force us to conclude that it cannot be considered as a candidate learning rule for real spiking neurons. In addition, a closer look at the physiological literature on STDP (Dan & Poo, 2004) reveals the importance, in the plasticity calculation occurring at post-synaptic densities, of action potentials back-propagating from the cell body to the synapse. This feedback within a neuron is completely missing from the model described here. Since our model is causal (see Figure 5B), and STDP is acausal (see Figure 5A), it is natural to conclude that the inclusion of this feedback within the neuron is necessary in order to account for the acausal Long Term Depression (LTD) part of the STDP effect (the left part of Figure 5A).

Acknowledgments

We are grateful for discussions with Nihat Ay, Michael Eisele, Hong Hui Yu, Surya Ganguli, Sophie Denève, Fabian Theis and Arunava Banerjee. AJB thanks Redwood colleagues for many such discussions.

References

Anthony, J. B., & Terrence, J. S. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*(6), 1129–1159.

Banerjee, A. (2001). On the phase-space dynamics of systems of spiking neurons. *Neural Computation, 13,* 161-225.

Bell, A. J., & Parra, L. C. (2005). Maximizing sensitivity in a spiking network. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems 17.* Cambridge, MA: MIT Press.

Bell, A. J., & Sejnowski, T. J. (1997). The 'independent' components of natural images are edge-filters. *Vision Research, 37,* 3327-3338.

Bohte, S. M., & Mozer, M. C. (2005). Reducing spike train variability: A computational theory of spike-timing dependent plasticity. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems 17* (p. 201-208). Cambridge, MA: MIT Press.

Dan, Y., & Poo, M. M. (2004). Spike timing-dependent plasticity of neural circuits. *Neuron, 44,* 23-30.

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

Olshausen, B., & Field, D. F. (1997). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature, 381,* 607-609.

Papoulis, A. (1991). *Probability, random variables, and stochastic processes*. New York: McGraw-Hill.

Shamir, M., & Sompolinsky, H. (2001). Correlation codes in neuronal networks. In *Advances in neural information processing systems 14.* Cambridge, MA: MIT Press.

Shriki, O., Sompolinsky, H., & Lee, D. D. (2000, November). An information maximization approach to overcomplete and recurrent representations. In *12th conference on neural information processing systems (nips 2000)* (p. 87-93). Cambridge, MA: MIT Press.

Toyoizumi, T., Pfister, J.-P., Aihara, K., & Gerstner, W. (2005). Spike-timing dependent plasticity and mutual information maximization for a spiking neuron model. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems 17* (p. 1409-1416). Cambridge, MA: MIT Press.

Appendix A
Transformation of probability density for over-complete and mixed variables

To establish Equation (1) we will proceed in three steps. First we will derive the transformation of a joint density function of continuous variables for the over-complete case assuming an invertible mapping. Then we will correct the expression for the case of a non-invertible transformation. Finally we will include discrete variables to have a transformation of a joint distribution of discrete and continuous variables.

In the first step we give an alternate derivation of the result in (Shriki et al., 2000) for over-complete mappings. A

mapping, $\mathbf{y} = \mathbf{f}(\mathbf{x})$, will transform a joint density $p(\mathbf{x})$ of the input $\mathbf{x}$ to a joint density $p(\mathbf{y})$ of the output $\mathbf{y}$ as

$$p(\mathbf{y}) = \int d\mathbf{x}\, p(\mathbf{y}|\mathbf{x}) p(\mathbf{x}) \,. \tag{17}$$

Since the mapping is deterministic the conditional density is given by

$$p(\mathbf{y}|\mathbf{x}) = \delta(\mathbf{y} - f(\mathbf{x})) \,. \tag{18}$$

First note that for a transformation which preserves dimensions, $\dim(\mathbf{y}) = \dim(\mathbf{x})$, the Kronecker delta distribution around a solution, $\mathbf{y}_o = \mathbf{f}(\mathbf{x}_o)$, is transformed as

$$\delta(\mathbf{y} - \mathbf{y}_o) = \left| \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \right|^{-1} \delta(\mathbf{x} - \mathbf{x}_o) \,, \tag{19}$$

where $|\frac{\partial \mathbf{y}}{\partial \mathbf{x}}| = \mathbf{J}$ is the Jacobian matrix evaluated at $\mathbf{x}_o$. To extend this to the non-square over-complete case, i.e. $\dim(\mathbf{y}) \geq \dim(\mathbf{x})$, define $\mathbf{y}^{\|}$ and $\mathbf{y}_{\perp}$ as the orientations at $\mathbf{y}_o$ that are parallel and orthogonal to the column space of $\mathbf{J}$ respectively. Specifically, $\mathbf{y}^{\|} = (\mathbf{J}^T\mathbf{J})^{-1/2}\mathbf{J}^T\mathbf{y}$, and similarly for $\mathbf{y}^{\perp}$ using the orthogonal space $\mathbf{J}^{\perp}$. Here the inner product $\mathbf{J}^T\mathbf{y}$ computes the projection into the parallel space while $(\mathbf{J}^T\mathbf{J})^{-1/2}$ is the normalization required to preserve scaling (unit Jacobian determinant, $|\partial(\mathbf{y}^{\|}, \mathbf{y}^{\perp})/\partial\mathbf{y}| = 1$). With this we can write

$$\begin{aligned}
\delta(\mathbf{y} - \mathbf{y}_o) &= 1\delta(\mathbf{y}^{\|} - \mathbf{y}_o^{\|})\delta(\mathbf{y}^{\perp} - \mathbf{y}_o^{\perp}) && (20)\\
&= \left| \frac{\partial \mathbf{y}^{\|}}{\partial \mathbf{x}} \right|^{-1} \delta(\mathbf{x} - \mathbf{x}_o)\delta(\mathbf{y}^{\perp} - \mathbf{y}_o^{\perp}) && (21)\\
&= \left| (\mathbf{J}^T\mathbf{J})^{-1/2}\mathbf{J}^T \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \right|^{-1} \delta(\mathbf{x} - \mathbf{x}_o)\delta(\mathbf{y}^{\perp} - \mathbf{y}_o^{\perp}) && (22)\\
&= \left| \mathbf{J}^T\mathbf{J} \right|^{-1/2} \delta(\mathbf{x} - \mathbf{x}_o)\delta(\mathbf{y}^{\perp} - \mathbf{y}_o^{\perp}) && (23)\\
&\propto \left| \mathbf{J}^T\mathbf{J} \right|^{-1/2} \delta(\mathbf{x} - \mathbf{x}_o) \,, && (24)
\end{aligned}$$

where we have applied (19) to the square transformations $\mathbf{y} \to (\mathbf{y}^{\|}, \mathbf{y}^{\perp})$ and again to $\mathbf{x} \to \mathbf{y}_{\|}$. Note that (23) reduces to (19) if the transformation is square and is therefore the generalization to the non-square over-complete case. By combining Equations (17), (18), and (24) and executing the integral in $\mathbf{x}$ it follows (as in (Shriki et al., 2000)) that

$$p(\mathbf{y}) \propto \left| \mathbf{J}^T\mathbf{J} \right|^{-1/2} p(\mathbf{x}(\mathbf{y})) \,. \tag{25}$$

In the integration we have assumed that the transformation was invertible so that $\mathbf{x}(\mathbf{y})$ is uniquely defined. In case that there are multiple inputs that can generate the same output $\mathbf{y}$ the integral in (17) reduces to a sum over those solutions (Papoulis, 1991). Denoting the set of solutions with $S(\mathbf{y}) = \{\mathbf{x}|\mathbf{y} = \mathbf{f}(\mathbf{x})\}$ we write

$$p(\mathbf{y}) \propto \sum_{\mathbf{x} \in S(\mathbf{y})} \left| \mathbf{J}^T\mathbf{J} \right|^{-1/2} p(\mathbf{x}) \,. \tag{26}$$

In this paper we are interested in a mapping with continuous and discrete variables. Adopting now again the notation of the main text this mapping is: $(\mathbf{t}, \mathbf{i}) \to (\mathbf{t}', \mathbf{i}')$. Equation (26) is equally valid if all probabilities are conditioned on some discrete event, for instance the occurrence of some discrete values $\mathbf{i}, \mathbf{i}'$. Therefore,

$$\begin{aligned}
p(\mathbf{t}', \mathbf{i}') &= \sum_{\mathbf{i}} p(\mathbf{t}'|\mathbf{i}, \mathbf{i}') p(\mathbf{i}, \mathbf{i}') && (27)\\
&= \sum_{\mathbf{i}} \sum_{\mathbf{t} \in S_{\mathbf{t}}(\mathbf{t}', \mathbf{i}')} \left| \mathbf{T}^T\mathbf{T} \right|^{-1/2} p(\mathbf{t}|\mathbf{i}, \mathbf{i}') p(\mathbf{i}, \mathbf{i}') && (28)\\
&= \sum_{\mathbf{i}} \sum_{\mathbf{t} \in S_{\mathbf{t}}(\mathbf{t}', \mathbf{i}')} \left| \mathbf{T}^T\mathbf{T} \right|^{-1/2} p(\mathbf{i}'|\mathbf{t}, \mathbf{i}) p(\mathbf{t}, \mathbf{i}) && (29)\\
&= \sum_{\mathbf{i} \in S_{\mathbf{i}}(\mathbf{i}, \mathbf{i}')} \sum_{\mathbf{t} \in S_{\mathbf{t}}(\mathbf{t}', \mathbf{i}')} \left| \mathbf{T}^T\mathbf{T} \right|^{-1/2} p(\mathbf{t}, \mathbf{i}) \,. && (30)
\end{aligned}$$

Here $S_{\mathbf{t}}(\mathbf{i}, \mathbf{i}')$ and $S_{\mathbf{i}}(\mathbf{i}, \mathbf{i}')$ are the set of solutions to $\mathbf{t}' = \mathbf{t}'(\mathbf{t}, \mathbf{i})$ and $\mathbf{i}' = \mathbf{i}'(\mathbf{t}, \mathbf{i})$ respectively. For (30) we used the fact that for a deterministic mapping, $p(\mathbf{i}'|\mathbf{t}, \mathbf{i}) = 1$ if $\mathbf{i}'$ is the result of the map to input $(\mathbf{t}, \mathbf{i})$, and 0 otherwise.

Appendix B
Independent Poisson processes

Here we simplify the joint density $q(\mathbf{t}', \mathbf{i}')$ for a set of independent Poisson processes. For a Poisson point process the times of events are uniformly distributed, hence

$$q(\mathbf{t}', \mathbf{i}') \propto q(\mathbf{i}') \,. \tag{31}$$

We now introduce a new variable, $\mathbf{n}'$, which indicates the number of spikes that occurred in each neuron during the observation time window. For any distribution we can write

$$q(\mathbf{i}') = \sum_{\mathbf{n}'} q(\mathbf{i}'|\mathbf{n}') q(\mathbf{n}') \,. \tag{32}$$

The spike count is deterministically dependent on $\mathbf{i}'$ and hence this sum only includes the term with $\mathbf{n}' = \mathbf{n}'(\mathbf{i}')$ since otherwise $q(\mathbf{i}'|\mathbf{n}' \neq \mathbf{n}'(\mathbf{i}')) = 0$. Furthermore, when the spike counts are obtained from independent Poisson processes, the index vector is uniformly distributed so that:

$$q(\mathbf{i}') = q(\mathbf{i}'|\mathbf{n}'(\mathbf{i}'))q(\mathbf{n}'(\mathbf{i}')) \propto q(\mathbf{n}'(\mathbf{i}')) = \prod_i q(n_i') \,. \tag{33}$$

Combining these Equations we obtain (4) in the main text.

Appendix C
Spike time sensitivity in the SRM

Here we give full details of the gradient ascent learning rule for Gerstner's Spike Response Model (Gerstner & Kistler, 2002), a generalization of the standard Integrate-and-Fire model. In this formulation the effect of a pre-synaptic spike at time t_l in neuron j_l on the membrane potential at time t is described by a post-synaptic potential or spike response

function $R(t-\hat{t}_i, t-t_l)$, which may also depend on the time of the most recent spike $\hat{t}_i$ in post-synaptic neuron i.

This response function is then weighted by the synaptic strength w_{ij}, which may represent either an excitatory or inhibitory synapses as determined by the sign of w_{ij}. In addition to the effects of $\hat{t}$ on R, refractoriness is also incorporated through an additive hyper-polarizing term, $\eta(t-\hat{t}_i)$. Thus, total membrane potential of neuron i is given by

$$u_i(t) = \eta_i(t-\hat{t}_i) + \sum_l w_{ij_l} R(t-\hat{t}_i, t-t_l)\,. \tag{34}$$

We have ignored here possible contributions from external currents which can easily be included without modifying the following derivations. The output firing times t'_k and indices i'_k are defined as the ordered set of times and indices for which $u_i(t)$ reaches firing threshold from below. In the presence of a dynamic threshold, $\vartheta(t-\hat{t}_i)$, the output spike times and indices are defined implicitly by the ordered set of solutions to

$$t'_k = t \quad : \quad \vartheta(t-t'_s) = u_{i'_k}(t) = \eta(t-t'_s) + \sum_l w_{i'_k j_l} R(t-t'_s, t-t_l)\,, \quad \frac{du_{i'_k}(t)}{dt} > 0\,. \tag{35}$$

Note that we have replaced $\hat{t}_i$ with t'_s, where $s = s(k)$ is the index of output spike preceding t'_k on the same neuron. This results from our indexing scheme which identifies t'_k as the time of the k-th output spike regardless of which neuron generated it. We find double indice notation cumbersome and would like to omit therefore the explicit reference to pre and post-synaptic index i and j as they result implicitly from spike index k and l. The threshold condition which defines post-synaptic spike times, t'_k, can be written in this simplified notation as

$$t'_k : \vartheta(t'_k - t'_s) = u_k = \eta(t'_k - t'_s) + \sum_l W_{kl} R(t'_k - t'_s, t'_k - t_l)\,, \quad \frac{du_k}{dt'_k} > 0\,, \tag{36}$$

where we also abbreviate, $u_k = u_{i'_k}(t'_k)$, and use the capital W_{kl} to indicate that this indexing scheme affects the synaptic weights as well. See Section for an explanation of the exact relationship between w_{ij} and W_{kl}.

Regardless, for this general model $\mathbf{T}_{kl}$ is given by

$$\begin{aligned} \mathbf{T}_{kl} = \frac{dt'_k}{dt_l} &= -\left(\frac{\partial u_k}{\partial t'_k} - \frac{\partial \vartheta_k}{\partial t'_k}\right)^{-1} \frac{\partial u_k}{\partial t_l} \\ &\quad -\left(\frac{\partial u_k}{\partial t'_k} - \frac{\partial \vartheta_k}{\partial t'_k}\right)^{-1} \frac{\partial u_k}{\partial t'_s}\frac{dt'_s}{dt_l}\,. \end{aligned} \tag{37}$$

For stereotyped $R(\tau,t)$, $\eta(t)$, and $\vartheta(t)$ we now define

$$\begin{aligned} \dot{R}_{kl} &= \frac{dR}{dt}(t'_k - t'_s, t'_k - t_l) && (38)\\ \tilde{R}_{kl} &= \frac{dR}{d\tau}(t'_k - t'_s, t'_k - t_l) && (39)\\ \dot{\eta}_k &= \frac{d\eta}{dt}(t'_k - t'_s) && (40)\\ \dot{\vartheta}_k &= \frac{d\vartheta}{dt}(t'_k - t'_s) && (41)\\ \dot{u}_k &= \dot{\eta}_k + \sum_l W_{kl}\dot{R}_{kl} + \sum_l W_{kl}\tilde{R}_{kl}\,, && (42) \end{aligned}$$

so that we may represent the recursion relationship of Equation (37) as

$$\begin{aligned} \mathbf{T}_{kl} &= \frac{W_{kl}\dot{R}_{kl}}{\dot{\eta}_k - \dot{\vartheta}_k + \sum_c W_{kc}\dot{R}_{kc} + \sum_c W_{kc}\tilde{R}_{kc}} \\ &\quad + \frac{\dot{\eta}_k - \dot{\vartheta}_k + \sum_c W_{kc}\tilde{R}_{kc}}{\dot{\eta}_k - \dot{\vartheta}_k + \sum_c W_{kc}\dot{R}_{kc} + \sum_c W_{kc}\tilde{R}_{kc}}\mathbf{T}_{sl}\,. \end{aligned} \tag{43}$$

In principle, this implicit expression for the matrix $\mathbf{T}$ may be solved by iteration or by simply inverting the square matrix associated with the second term. Unfortunately, the complex dependence of s on k causes this procedure to lead to a rather complicated expression for the learning rule. This issue can be avoided when the neurons have a low firing rate (compared to their refractory period), or simply, when they have a weak refractoriness. A low firing rate is not an uncommon assumption in theoretical modes (Gerstner & Kistler, 2002), and in particular, it is a good assumption for our simulations. Both of these assumptions imply that $\dot{\eta}_k$, $\dot{\vartheta}_k$ and $\tilde{R}_{kl}$ are all small compared to $\dot{R}_{kl}$. When this is the case, the second term of (43) may be neglected and the derivative of this equation with respect to W_{kl} is approximated by

$$\begin{aligned} \frac{\partial \mathbf{T}_{ab}}{\partial W_{kl}} &= \frac{\partial}{\partial W_{kl}}\left[\frac{W_{ab}\dot{R}_{ab}}{\dot{u}_a - \dot{\vartheta}_a}\right] \\ &= \delta_{ak}\delta_{bl}\frac{\dot{R}_{ab}}{\dot{u}_a - \dot{\vartheta}_a} - \frac{W_{ab}\dot{R}_{ab}\delta_{ak}\dot{R}_{al}}{\left(\dot{u}_a - \dot{\vartheta}_a\right)^2} \\ &= \delta_{ak}\mathbf{T}_{ab}\left[\frac{\delta_{bl}}{W_{ab}} - \frac{\mathbf{T}_{al}}{W_{al}}\right]. \end{aligned} \tag{44}$$

Therefore, since

$$\mathbf{T}^{\#T}_{ab} \equiv \frac{1}{2}\frac{d}{d\mathbf{T}_{ab}}\log|\mathbf{T}^T\mathbf{T}| = \left[\mathbf{T}(\mathbf{T}^T\mathbf{T})^{-1}\right]_{ab}\,, \tag{45}$$

we may conclude that

$$\begin{aligned} \frac{1}{2}\frac{\partial \log|\mathbf{T}^T\mathbf{T}|}{\partial W_{kl}} &= \sum_{ab}[\mathbf{T}^{\#T}]_{ab}\delta_{ak}\mathbf{T}_{ab}\left[\frac{\delta_{bl}}{w_{ab}} - \frac{\mathbf{T}_{al}}{W_{al}}\right] && (46)\\ &= \frac{\mathbf{T}_{kl}}{W_{kl}}\left([\mathbf{T}^{\#T}]_{kl} - \sum_b [\mathbf{T}^{\#T}]_{kb}\mathbf{T}_{kb}\right) && (47)\\ &= \frac{\mathbf{T}_{kl}}{W_{kl}}\left([\mathbf{T}^{\#}]_{lk} - [\mathbf{T}\mathbf{T}^{\#}]_{kk}\right). && (48) \end{aligned}$$