# Learning Temporal Clusters with Synaptic Facilitation and Lateral Inhibition

Chris L. Baker and Aaron P. Shon and Rajesh P.N. Rao

*Department of Computer Science and Engineering*

*University of Washington*

*Seattle, WA 98195-2350*

---

**Abstract**

Short-term synaptic plasticity has been proposed as a way for cortical neurons to process temporal information. We present a model network that uses short-term plasticity to implement a temporal clustering algorithm. The model's facilitory synapses learn temporal signals drawn from mixtures of nonlinear processes. Units in the model correspond to populations of cortical pyramidal cells arranged in columns; each column consists of neurons with similar spatiotemporal receptive fields. Clustering is based on mutual inhibition similar to Kohonen's SOMs. A generalized expectation maximization (GEM) algorithm, guaranteed to increase model likelihood with each iteration, learns the synaptic parameters.

*Key words:* dynamic synapses, facilitation, mixture models, temporal filtering

---

*Email address:* {`clbaker,aaron,rao`}`@cs.washington.edu` (Chris L. Baker and Aaron P. Shon and Rajesh P.N. Rao).

## 1 Introduction

Dynamic synapses in the cortex have been linked to processing of temporal information because of their nonlinear filtering properties [5] and their sensitivity to frequency changes in input [10]. In this paper we study how temporal processing in synchronously firing neurons with dynamic synapses, modulated by lateral inhibition between populations, contributes to learning temporal clusters in input. We use a population model of neurons with facilitory synapses based on the model in [10]. This and other related models were shown to have rich nonlinear spatiotemporal filtering properties in a supervised learning context in [5,6]. However, it is unclear how such supervised learning algorithms could be implemented in the brain. A possible mechanism for generating training signals for individual neuronal populations comes from the self-organizing map (SOM) model [4]. This model uses a neighborhood function that determines the receptive field of a population. Lateral inhibition limits activation to spatially contiguous neighborhoods for each input. Together, these features determine how the population parameters get updated.

In our model, we link the SOM's biologically plausible rules for local, competitive learning with a commonly used statistical algorithm called generalized expectation maximization (GEM). Under mild assumptions, the algorithm provably increases the likelihood of the model with each iteration [7]. Section 2 introduces the dynamic synapse model and a supervised training algorithm. Section 3 describes the mixture model and an unsupervised training algorithm (GEM) for estimating maximum likelihood model parameters. Section 4 presents simulation results on learning clusters of artificial time series.

## 2 Dynamic synapse model

Our population model of neurons with dynamic synapses is based on [10]. Use of population models of neurons with dynamic synapses is justified in in [5] and [9]. The computational properties of dynamic synapses compare favorably [5,6] with other models, such as [2]. As shown in [5], facilitory synapses have rich temporal filtering properties, and in feedforward networks can approximate any nonlinear filter whose response can be described using a Volterra series.

In our simulations, dynamic synapses receive real valued inputs corresponding to firing rates and update their synaptic weights at discrete time steps. Let $x(t)$ be the real-valued presynaptic input at time $t$. Let $\Theta$ be a set of hyperparameters that determine the temporal behavior of a synapse, in our model consisting of scalars $\alpha$, $\beta$, and $\mu$. The dynamic update of a synaptic weight $w$ is defined by the equation:

$$w(t+1) = w(t) + \alpha\Big(\mu - w(t)\Big) + \beta x(t). \tag{1}$$

In our model, the parameter $\beta$ is constrained to be positive to simulate synaptic facilitation. Equations (2) and (3) show the differential equation for the value of a dynamic synaptic weight and its solution, respectively:

$$\frac{dw(t)}{dt} = \alpha\Big(\mu - w(t)\Big) + \beta x(t), \tag{2}$$

$$w(t) = \mu + \beta \int_0^\infty e^{-\alpha\tau} x(t-\tau) d\tau. \tag{3}$$

Equation (3) shows that our model of a dynamic synapse implements a temporal filter with fading memory over previous inputs. The hyperparameter $\alpha$ determines the decay rate of the temporal filter, $\beta$ determines the amount of facilitation of the filter, and $\mu$ determines the maximum synaptic conductance.

Units in the model receive multidimensional input signals that change over time. Postsynaptic response depends on both the magnitude of present inputs and fading memory over past inputs. Let $M$ be the dimension of the input to a single unit. Let $\mathbf{x}(t)$ be an input vector of dimension $M \times 1$ at time $t$. Let $\mathbf{w}(t)$ be a vector of dynamic synaptic weights of dimension $M \times 1$ at time $t$. The real-valued output $y(t)$ of a unit for input $\mathbf{x}(t)$ is:

$$y(t) = \sigma\Big(\mathbf{w}(t)^{\mathrm{T}}\mathbf{x}(t)\Big), \tag{4}$$

where $\sigma$ is the sigmoid function, defined as $\sigma(x) = 1/(1 + \exp(-x))$. Let $L$ be the number of paired input/output matrices given as supervised training data. Let $T$ be the number of time steps for inputs (assumed to be constant across all inputs). Let $\mathbf{X}_l$ be the $l$th matrix of inputs, of dimension $M \times T$, for $l \in \{1 \ldots L\}$. Let $\mathbf{y}_l$ be a vector of real valued outputs of dimension $T \times 1$ for input $\mathbf{X}_l$, and let $\mathbf{d}_l$ be a vector of dimension $T \times 1$ of supervised training values for input $\mathbf{X}_l$. Let $\mathbf{e}_l = (\mathbf{d}_l - \mathbf{y}_l)$ be a vector of error values. Then:

$$SSE = \sum_{l=1}^{L} \mathbf{e}_l^{\mathrm{T}}\mathbf{e}_l \tag{5}$$

is the sum squared error of the output of a neuron over all inputs. The *SSE* can be minimized in a variety of ways, but we use conjugate gradient minimization for its computational efficiency. Minimizing the *SSE* with conjugate gradient requires computing gradients and Hessians of *SSE* with respect to each of the hyperparameters $\alpha$, $\beta$, and $\mu$. Fig. 1 demonstrates supervised training on a sample two dimensional input.

## 3 Learning spatiotemporal clusters with dynamic synapses

Cortical columns contain excitatory, recurrent, local connections, and mutually inhibitory long-range connections. This physiology motivates Kohonen's SOM model [4]. The SOM model spatially clusters its input data using lateral inhibition and the notion of spatially contiguous neighborhoods in which synaptic modification occurs.

In our model, the mean firing rates of nearby cortical columns comprise a distribution over spatiotemporal processes observed by sensory cortices. Our model assumes that each input sample is independently generated by one of several nonlinear stochastic processes, and infers which process is most likely to have generated each input. Our mean-field cortical model therefore performs spatiotemporal clustering, where nearby cortical columns are assumed to be enervated by afferents from a single, spatially contiguous region of sensory cortex. Within each spatial region of nearby columns, individual columns respond to sensory firing patterns generated by particular stochastic processes. When a spatiotemporal input pattern arrives, nearby columns compete for activation. The higher a column's activation, the greater the probability that the stochastic process it encodes produced the input pattern.

As a practical example, consider the task of identifying the velocities of rapidly-moving objects in a monkey's visual field. For a single spatial neighborhood in the monkey's retinotopic map, an object might be perceived as a single moving dark point. Movement of the point (and thus the object) at different speeds is generated by a different temporal sequence (or process). We propose that a different cortical column classifies each temporal sequence. During learning, lateral inhibition encourages competition for synaptic modification across columns. After learning, lateral inhibition causes competitive normalization of activity levels, ensuring that activities of columns within a single spatial neighborhood encode a probability distribution (Fig. 2(c)). Similar clustering could take place in, for example, auditory cortex, for classifying speech signals.

A model related to ours that investigates temporal clustering using dynamic synapses and lateral inhibition appears in [8]. This model clusters neuronal spike trains using a spike timing dependent learning rule based on feedback gradients from hard lateral inhibition. Unlike the model in [8], our temporal filtering units correspond to populations of neurons (cortical columns) rather than individual neurons. Thus, competition in our model occurs between populations rather than between single neurons fully connected to the input layer by dynamic synapses. In addition, the lateral inhibition in our model does not limit activation to one unit, but instead computes a probability distribution over the activation of all units. These features allow our model to represent different classes of temporal patterns than the model in [8].

Furthermore, our model admits a desirable probabilistic interpretation. Our spatiotemporal clustering algorithm is a variant of expectation maximization (EM) [3] called generalized EM (GEM) [7] that uses an approximate M step. EM is an iterative algorithm where each iteration consists of two steps. Let $C$ be the number of cortical columns, and let $\Theta_c^i$ denote the hyperparameters of column $c$ on iteration $i$ of the algorithm. The E step computes the function $Q(\Theta|\Theta^i)$, defined as the expected log-likelihood of the model given $\Theta^i$:

$$Q(\Theta|\Theta^i) \overset{def}{=} E\Big[\log\Big(p(\{\mathbf{X}_l\}_1^L|\Theta)\Big)\Big|\Theta^i\Big]. \tag{6}$$

During the M step, each column tunes its hyperparameters to maximize activation in response to patterns that are likely to have been generated by the stochastic process the column encodes. To maximize the activation of a column when the response of its dynamic synapses most closely resembles the input pattern, weight vectors $\mathbf{w}_c$ for each column $c \in \{1 \dots C\}$ are tuned to maximize the normalized dot product between the weights and the input patterns they select for most strongly.

Let $p(c|\mathbf{X}_l, \Theta_c^i)$ be the likelihood that column $c \in \{1 \dots C\}$ selects for a particular spatiotemporal input pattern $\mathbf{X}_l$, for $l \in \{1 \dots L\}$. Each input pattern is assumed independent of the other patterns and corrupted by zero-mean Gaussian noise. For a Gaussian error likelihood with uniform variance, $Q(\Theta|\Theta^i)$ takes the form:

$$\begin{aligned} Q(\Theta|\Theta^i) &= \sum_{c=1}^{C}\sum_{l=1}^{L}\sum_{t=1}^{T} p\Big(c\Big|\mathbf{X}_l, \Theta_c^i\Big) \log\Big(p\Big(\mathbf{x}_l(t)\Big|\Theta_c^i\Big)\Big) \\ &\propto -\sum_{c=1}^{C}\sum_{l=1}^{L}\sum_{t=1}^{T} p\Big(c\Big|\mathbf{X}_l, \Theta_c^i\Big)\Big(\Big(\mathbf{x}_l(t) - \mathbf{w}_c(t)\Big)^{\mathrm{T}}\Big(\mathbf{x}_l(t) - \mathbf{w}_c(t)\Big) + \mathit{const}\Big). \quad (7) \end{aligned}$$

The M step of the algorithm uses conjugate gradient maximization of $Q(\Theta|\Theta^i)$ with respect to $\Theta$ to maximize the likelihood of the model parameters for the next iteration. Similar to the supervised learning case, this requires computing gradients and Hessians of $Q(\Theta|\Theta^i)$ with respect to the hyperparameters $\alpha$, $\beta$, and $\mu$. The parameters for each column can be optimized independently. For $c \in \{1 \dots C\}$:

$$\Theta_c^{i+1} = \operatorname*{argmin}_{\Theta} \sum_{l=1}^{L}\sum_{t=1}^{T} p\Big(c\Big|\mathbf{X}_l, \Theta_c^i\Big)\Big(\Big(\mathbf{x}_l(t) - \mathbf{w}_c(t)\Big)^{\mathrm{T}}\Big(\mathbf{x}_l(t) - \mathbf{w}_c(t)\Big)\Big). \quad (8)$$

From equation (8) it is apparent that maximizing the likelihood of the mixture model is equivalent to minimizing a weighted sum-squared difference between weights and inputs for all data points. The estimate $\Theta^{i+1}$ is not a global maximum of $Q(\Theta|\Theta^i)$, but instead a local one found by conjugate gradient. Because the M step is not exact, our algorithm performs generalized EM, which is still guaranteed to converge to a local maximum in the likelihood function [7]. Note that a prior probability on the activation of each column $c$ could be added; here we assume a uniform prior over activation values for all columns. A nonuniform prior distribution might correspond, e.g., to a "top-down" activation signal arriving from distant association cortices or prefrontal cortex. A nonuniform prior may also select for uncorrelated or independent activations among cortical columns, as in PCA or ICA.

## 4 Results

Fig. 1 demonstrates how a population with two inputs can be trained in a supervised manner using conjugate gradient to map multi-dimensional inputs to desired outputs. In this example, the network was trained on set of 75 input/output pairs, and cross-validated on set of 500 input/output pairs. The sequence length of each input/output pair was 100. The MSE per sequence of the network on the cross-validation set was $6.7 \times 10^{-4}$ after convergence, showing that the algorithm learns the input-output mapping with high accuracy.

Further simulation results demonstrate how our unsupervised clustering algorithm can differentiate between noisy 3-dimensional visual stimuli moving at different speeds. The simulations use two model cortical columns; after learning, each column encodes a different trajectory of motion. The algorithm was presented a training set of 100 3-dimensional data points, each drawn from one of two sources and corrupted by noise. To test discriminative and generative generalization, 200 3-dimensional data points, drawn from the same sources as the training set, were used for cross validation. Fig. 2(d) shows that after 5 iterations, the algorithm converges to a local maximum in likelihood space, and learning is completed. Fig. 2(c) shows that the algorithm correctly classifies each trajectory from the testing set. Figs. 2(a) and 2(b) show the weights for a sample trajectory from the cross validation set. The weights of column 2 accurately track the input, while the weights of column 1 do not, and for this instance, the algorithm selects column 2. For inputs that column 1 selects, the weights of column 1 track the input while the weights of column 2 do not. These results demonstrate that the algorithm increases the likelihood of the model with each iteration by robustly learning a correct assignment of inputs to cortical columns in the presence of sensory noise.

## 5 Conclusion

The brain undoubtedly employs plasticity on several timescales to process temporal signals. Dynamic synapses are promising candidates for classifying and predicting quickly fluctuating time series. We have demonstrated a population-level model for how dynamic synapses can cluster temporal input sequences. Learning in the model can be performed in a supervised or in an unsupervised manner. Future work will involve expanding our framework to handle long time courses with many clusters, with data drawn from several different sensory modalities such as audition or vision. Methods for further representing and exploiting statistical regularities in sensory data such as sparseness and higher order dynamics will be explored.

## References

[1] L. F. Abbott, J. A. Varela, K. Sen, S. B. Nelson, Synaptic depression and cortical gain control, Science 17 (20).

[2] A. D. Back, A. C. Tsoi, A simplified gradient algorithm for IIR synapse multilayer perceptrons, Neural Computation 5 (456-462).

[3] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. Royal Statistical Soc. Series B 39 (1) (1977) 1–38.

[4] T. Kohonen, Self-Organizing Maps, Springer, 1997.

[5] W. Maass, E. D. Sontag, Neural systems as nonlinear filters, Neural Computation 12 (1743-1772).

[6] T. Natschläger, W. Maass, E. D. Sontag, A. Zador, Processing of time series by neural circuits with biologically realistic synaptic dynamics, in: T. K. Leen, T. G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, Vol. 13, 2000, pp. 145–151.

[7] R. M. Neal, G. E. Hinton, A view of the EM algorithm that justifies sparse, incremental, and other variants, in: M. I. Jordan (Ed.), Learning in Graphical Models, 1998.

[8] J. Storck, F. Jäkel, G. Deco, Temporal clustering with spiking neurons and dynamic synapses: towards technological applications, Neural Networks 14 (2001) 275–285.

[9] M. Tsodyks, K. Pawelzik, H. Markram, Neural networks with dynamic synapses, Neural Computation 10 (821-835).

[10] J. A. Varela, K. Sen, J. Gibson, J. Fost, L. F. Abbott, S. B. Nelson, A quantitative description of short-term plasticity at excitatory synapses in layer 2/3 of rat primary visual cortex, The Journal of Neuroscience 17 (20).

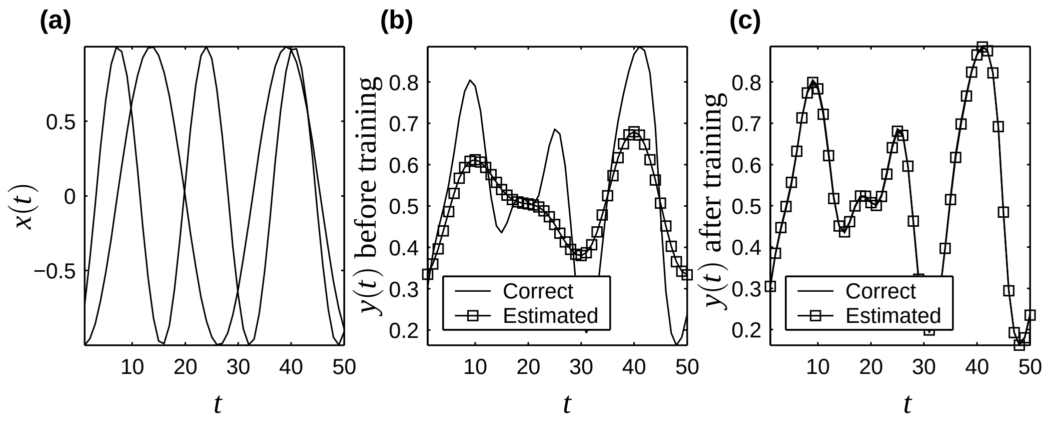

**Fig. 1: Supervised temporal learning with dynamic synapses.** (a) Subsequence of a two-dimensional input pattern from the cross validation set. (b) Before training: output of a population given the input from (a) compared with the desired output. (c) After training: output of a population given the input from (a) compared with the desired output. The learned output of the network accurately matches the desired output.

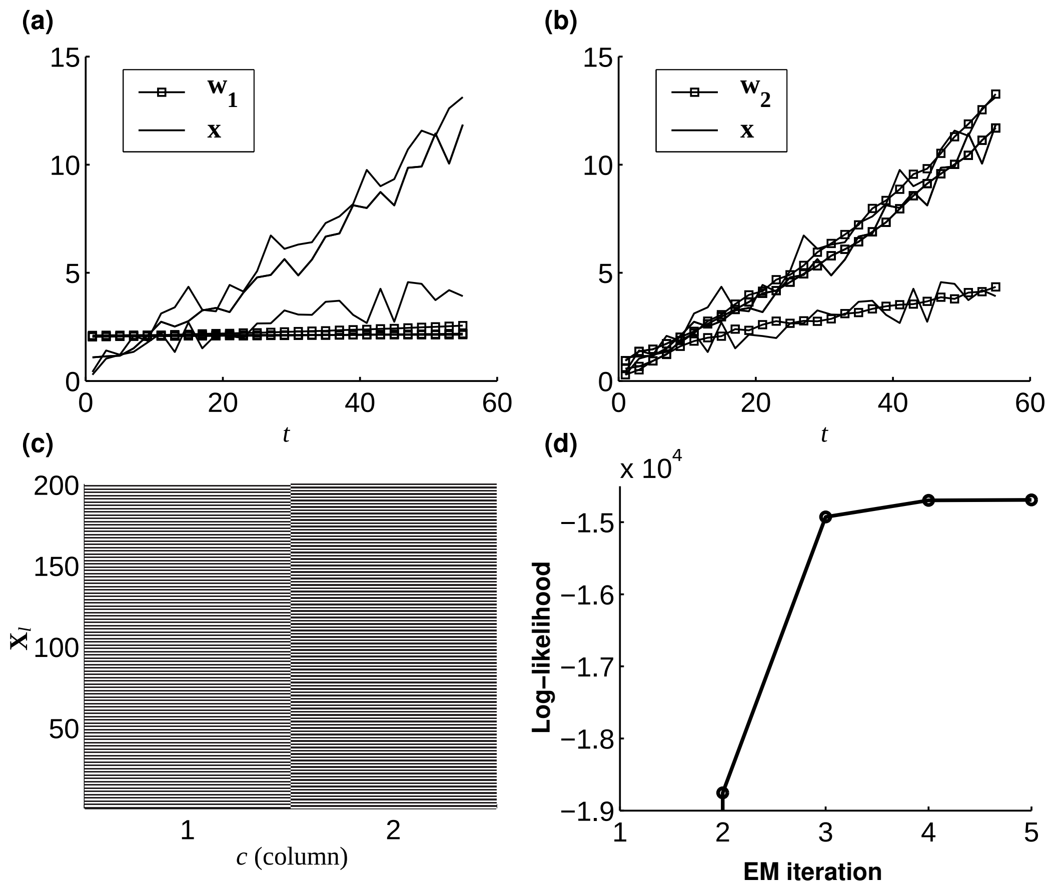

**Fig. 2: Unsupervised spatiotemporal clustering of inputs from a cross validation set.** (a) After training on a set of 100 inputs, weight vectors of the first column compared with input vectors over an example sequence from the testing set of 200 inputs. (b) Weight vectors of the second column compared with input vectors over the same sequence as in (a). The weights of the second column track the input, while the weights of the first column do not, indicating that the second column is more likely to code for the input. (c) After training, probability that a column selects for an input sequence from the cross validation set. The lighter the square, the higher the probability $p(c|\mathbf{X}_l, \Theta_c)$. 200 testing inputs were presented from two stochastic processes, and the algorithm correctly identified them (inputs from different sources were presented in alternating order; the checkered pattern represents a correct classification). (d) Log-likelihood of the generative mixture model increases with each iteration of training.