Think of a title later

Ansel Blumers and Ankan Ganguly

December 13, 2015

Abstract

Abstract goes here.

Contents

1 Introduction

Write an introduction here.

2 Methods

In this paper, we used the simulated Integrate and Burst model introduced in [1]. We then implemented STDP learning and Hebbian learning.

2.1 Integrate and Burst Model

Our integrate and burst model is an implementation of equations described in [1]. We provide a detailed description of those equations here, with more attention paid to the intuition behind these equations. There are N neurons connected in an all-to-all environment. Each neuron i bursts when its membrane potential, V_i, hits the threshold V_θ. We assume that every neuron bursts for T_{burst} time. While bursting, each neuron fires four times uniformly over the burst interval before resetting to V_{reset}.

When a neuron is not bursting, its potential is governed by a typical conductance based leaky integrate and fire model with built-in inhibition:

$$\tau_V \frac{\mathrm{d}V_i}{\mathrm{d}t} = -g^L(V - V^L) - g_i^E(V - V^E) - g_i^I(V - V^I) \tag{1}$$

The leak conductance, g^L, is assumed to be a homogeneous constant input, as is the leak potential V^L. Notice that in the absence of excitatory or inhibitory conductance, neuron i will tend to V^L. Therefore we can say that the leak potential is also equal to the rest potential.

The excitation potential, V^E, and the inhibition potential, V^I, act as upper and lower bounds on untethered (What was the word for IF potential plots without a firing threshold?) potential respectively.

The excitation conductance, g^L, is defined by the activity in neighboring neurons as well by external random stimulation:

$$g^L = Ws + W_0 b$$

Where W_{ij} is the strength of the synapse from neuron j to neuron i, W_0 is the conductance strength of external synapses and b is a Poisson random variable with frequency r_{in}. r_{in} was an input parameter. In [1] r_{in} was constant, but we chose to assign r_{in} a high value and anneal it to jump-start neural activity. s_i is the activation of neuron i. It is incremented each time neuron i fires, and it decays by

$$\frac{\mathrm{d}s}{\mathrm{d}t} = -\tau s$$

The inhibitory conductance, g^I, is defined as the sum of the adaptation inhibition, g^I_{ada}, and the global inhibition, g^I_{glob}. The adaptation inhibition is the internal inhibition generated by activation of a neuron. It is defined by the same equations as s_i, but with a constant multiplier A_a and a slower time-constant τ_{ada}.

2.2 Learning

The purpose of this paper is to replicate and test some of the claims made in [1]. One of the claims made is that the model used by [1] demonstrates how STDP can contribute to the synchronous regular synfiring chains exhibited in Zebra Finch (Where? Maybe cite second paper here?). We implemented STDP, but we also implemented a pure Hebbian learning rule to demonstrate that STDP learning is not necessary for the model to exhibit synchronous regular firing chains.

2.2.1 STDP Learning

We followed [1] precisely in our implementation of STDP learning. Define

$$K(t) = \begin{cases} e^{-t/\tau_{STDP}} & \text{if } t > 0 \\ -e^{-t/\tau STDP} & \text{if } t < 0 \\ 0 & \text{otherwise} \end{cases}$$

as the STDP kernel. For every pair of neurons i, j, let $t_i < t_j$ be two times when i and j fired respectively. Then According to STDP, the weight matrix element W_{ij} should increase proportional to $K(t_j - t_i)$. Notice that the change in W according to STDP is approximately anti-symmetric. [1] strengthened the nonlinearity of STDP by making STDP growth of a synapse proportional to the strength of that synapse (with an added factor so 0 weight synapses could grow). In particular, the paper defined:

$$\Delta_{ij}^{STDP}(t) = \left(\frac{W_{ij}(t-1)}{w_{max}} + 0.001\right) * \left(x_i(t)K(0)x_j(t) + \sum_{\tau=0}^{t}[x_i(t)K(\tau)x_j(t-\tau) - x_i(t-\tau)K(\tau)x_j(t)]\right) \tag{2}$$

Where $x_i(t)$ is a binary variable taking the value 1 if neuron i fired at time t. So, at each time t, $\Delta^{STDP}(t)$ is proportional to the change in W with respect to STDP. Our implementation assumed $K(t) \approx 0$ for $t > 4$ms.

However, the key innovation of this paper was to introduce a secondary source of competition. If the total strength of all synapses into or out of a given neuron exceed a soft limit W_{max} after STDP learning, then all such synapses experience long-term depression (LTD) proportional to the amount by which the soft limit is exceeded:

$$\theta_i^{col} = \left[\sum_{j=1}^{N}(W_{ij} + \eta\Delta_{ij}^{STDP}) - W_{max}\right]^+ \tag{3}$$

θ^{col} denotes the amount by which the soft limit has been exceeded [1]. Define θ^{row} similarly. Then,

$$\frac{\mathrm{d}W_{ij}}{\mathrm{d}t} = \eta\Delta_{ij}^{STDP} - \epsilon(\theta_i^{col} + \theta_j^{row}) \tag{4}$$

This updates weights according to STDP along with a penalty if the soft limit is exceeded. Finally, to ensure that STDP never diverges, we implement a hard limit. If any element $W_{ij}(t) > w_{max}$, where w_{max} is a constant parameter, we manually set $W_{ij}(t)$ to w_{max}.

[1] Our implementation was a little different and [1] defined θ^{col} differently, however this is the correct way to define θ^{col}. See the results and discussion for more details.

2.2.2 Hebbian Learning

Hebbian plasticity is a general family of learning rules which strengthen synapses between neurons which fire with high correlation. Hebbian learning rules are generally considered incomplete because while they explain how synapses are strengthened, they do not explain how synapses might be weakened.

One way to complete the Hebbian learning rule is with STDP. In STDP learning, highly correlated neurons experience a large change in synaptic strength, but the direction of the change (increase or decrease) depends on the relative timing of firing between the synapses.

However, the soft and hard limits imposed by the model also provide a way for synapse strength to decrease, so it should be possible to implement this model using Hebbian plasticity. To do this, we ran the exact same learning algorithm as implemented for STDP, but we modified the kernel to be

$$K(t) = \begin{cases} e^{-t/\tau_{Heb}} & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases}$$

With this new kernel,

$$\Delta_{ij}^{Heb}(t) = \left(\frac{W_{ij}(t-1)}{w_{max}} + 0.001\right) * \left(x_i(t)K(0)x_j(t) + \sum_{\tau=0}^{t} x_i(t)K(\tau)x_j(t-\tau)\right) \tag{5}$$

gives us a Hebbian learning rule. We implemented the soft and hard synaptic weight limits as in STDP learning.

2.3 Parameter Choices

Our implementation of the Integrate and Burst neuron network with STDP learning matched that used in [1] in many cases. However a few parameters were chosen differently. The table below provides a list of parameters we used:

Parameters	Values
dt	2×10^{-5}s
V^E	0V
V_θ	-0.05V
τ	0.004s
N	50
η	varies
A_a	9

Parameters	Values
τ_V	0.01 F/m^2
V^I	-0.07V
V_{reset}	-0.055V
r_{in}^{start}	10000Hz
w_{max}	0.14
ϵ	varies
$\tau_{STDP} = \tau_{Heb}$	0.02s

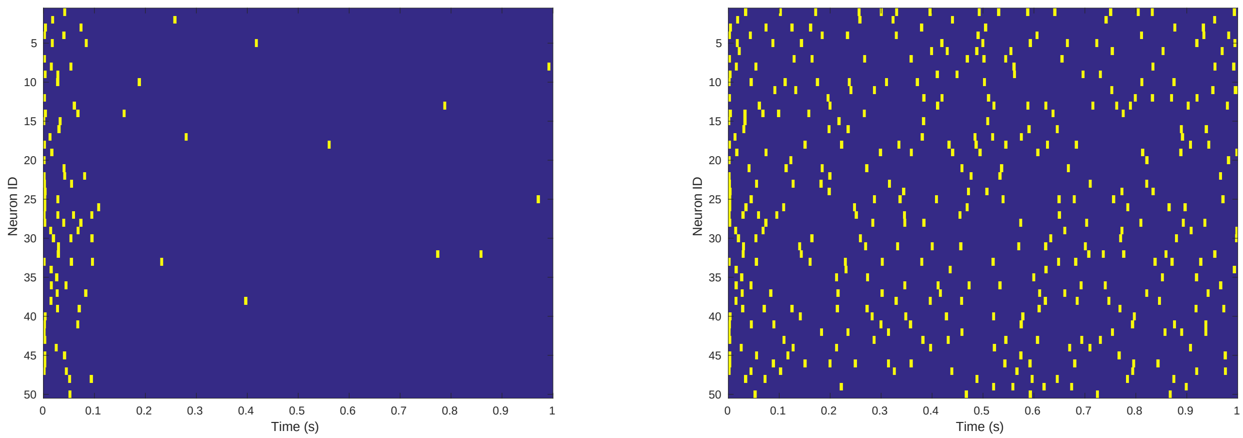
Parameters	Values
V^L	-0.06V
W_0	5S/m^2
T_{burst}	0.006s
r_{in}^{min}	4000Hz, 6000Hz
W_{max}	0.14
A_g	4
τ_{ada}	0.015s

3 Results

Introduce the big idea and what we got.

3.1 Parameter Tuning

- 4000 Hz doesn't work! (Burst Plot)

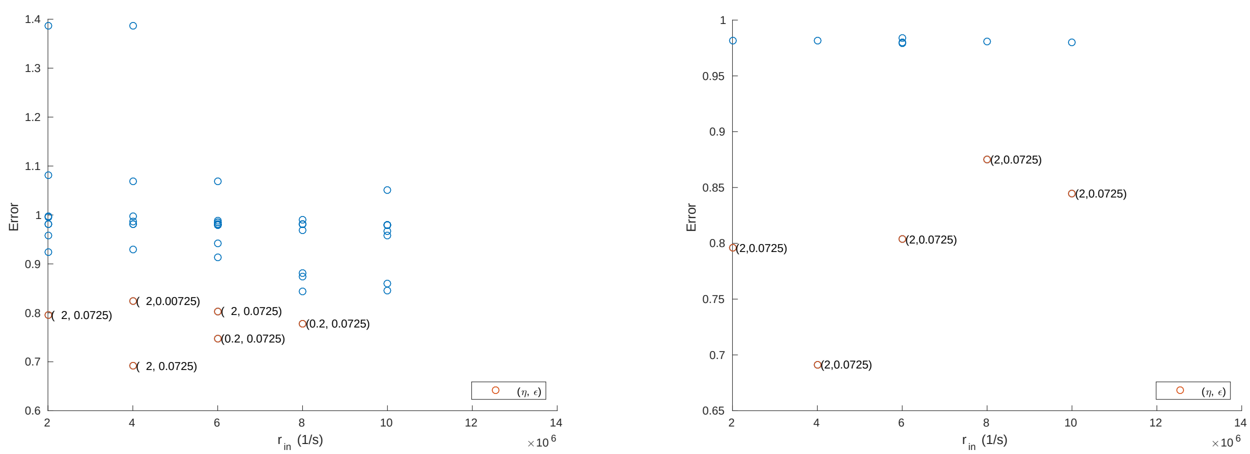


(a) 4000 Hz

(b) 6000 Hz

Figure 1: Compare the two

- Mention annealing and our choice of r_{in}, η and ϵ. Name the two data sets we refer to for the remainder of the paper. (Scatter Error Function)

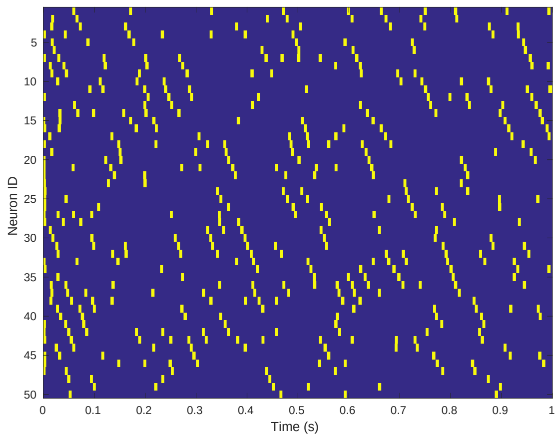


(a) Scatterplot of all errors

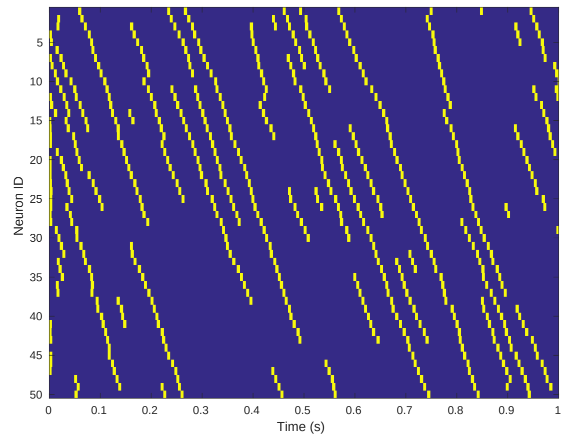
(b) Scatterplot of all errors with constant product

Figure 2: Compare the two

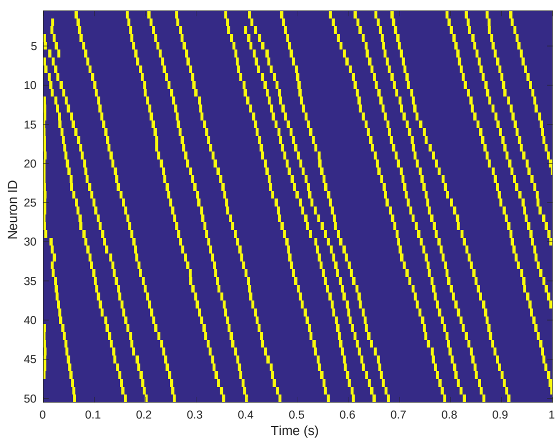
- Setting w_{max}. (Burst History)



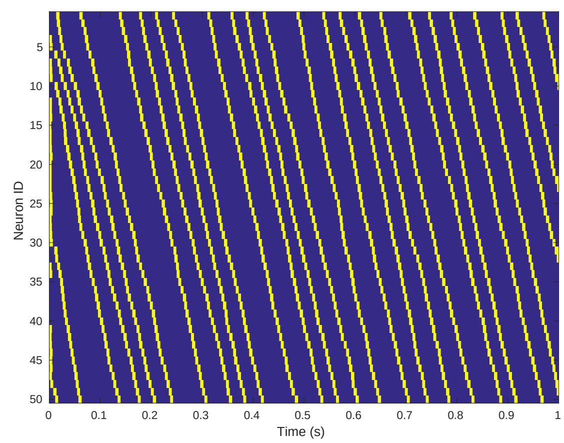
(a) Bursts without learning, $w_max = 0.14$.



(b) Bursts without learning, $w_max = 0.3$.



(c) Bursts without learning, $w_max = 0.5$.



(d) Bursts without learning, $w_max = 0.7$.

Figure 3: Compare the four

3.2 Convergence and Stability

- Demonstrate the stability of our IB model by showing the firing rate plot and how it splits according to r_{in}.

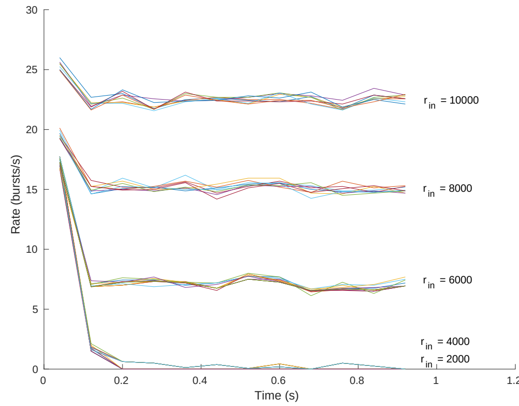
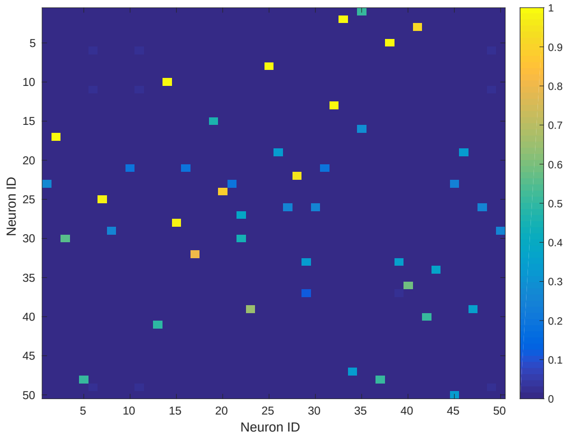
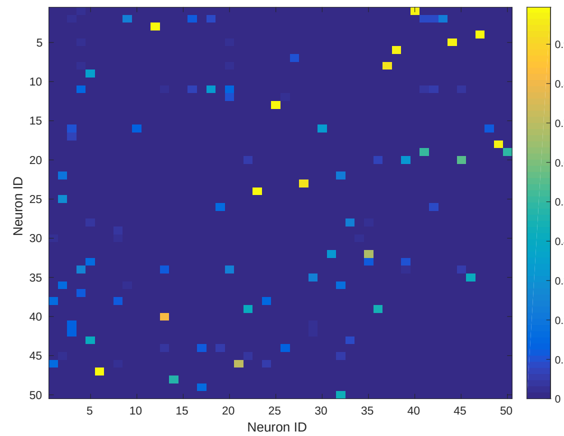


Figure 4: Caption will go here

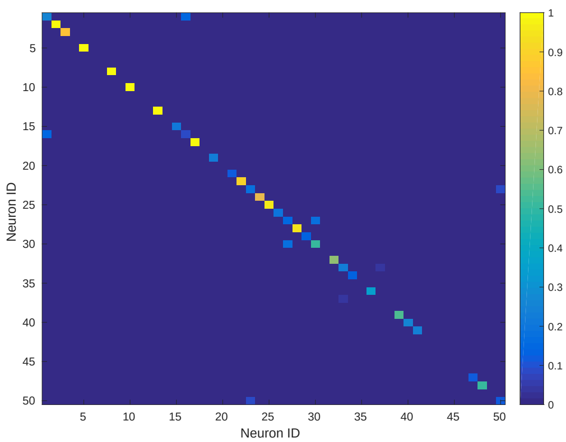
- Plot Weight and WW^T for 4000 and 6000 Hz to show some level of convergence.



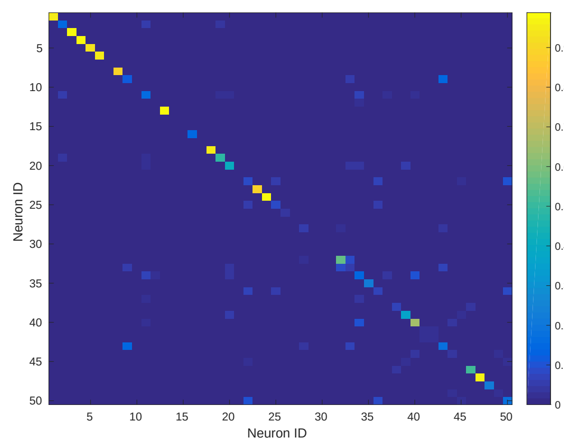
(a) Weight matrix of 4000Hz annealed



(b) Weight matrix of 6000Hz annealed



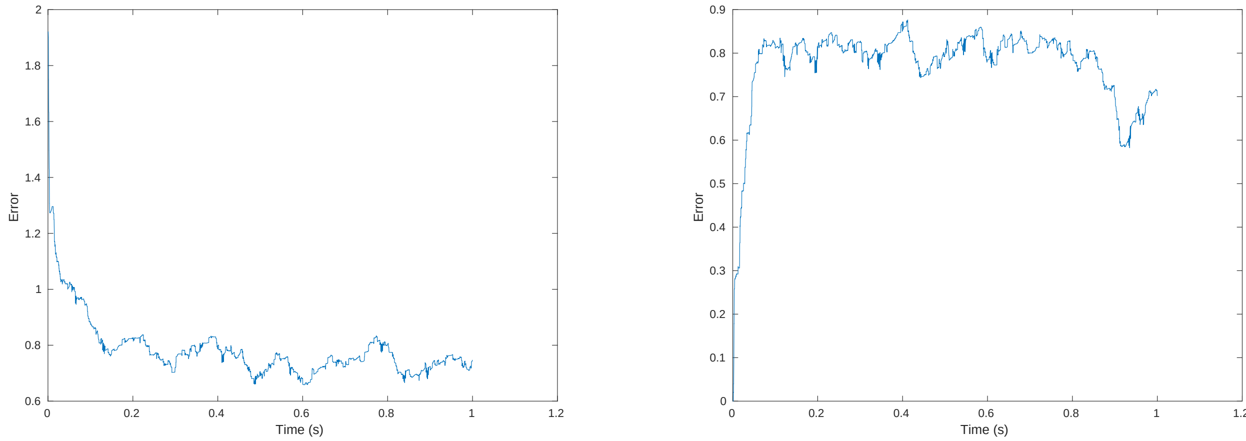
(c) WW^T of 4000Hz annealed



(d) WW^T of 6000Hz annealed

Figure 5: Compare the four

- Plot error function over time from normal and from permutation matrix



(a) Plot of error in weights over time starting from a full connection weight matrix.

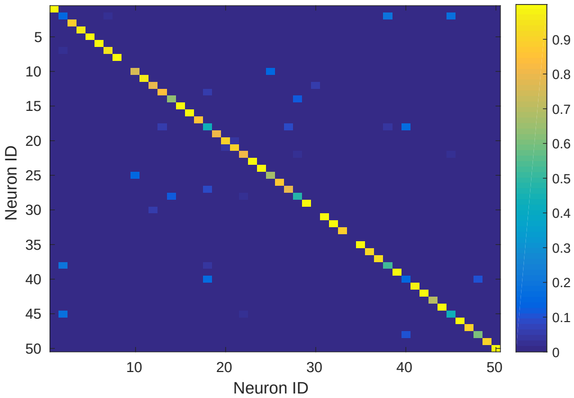
(b) Plot of error in weights over time starting from a permutation weight matrix.

Figure 6: Compare the two

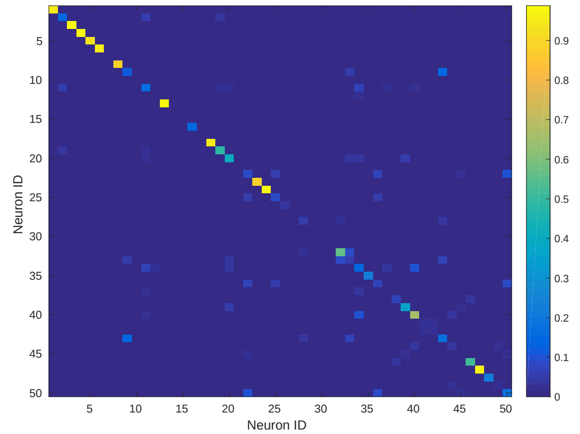
- Describe why the error function converges away from (mistake, see discussion).

3.3 Hebbian Learning versus STDP

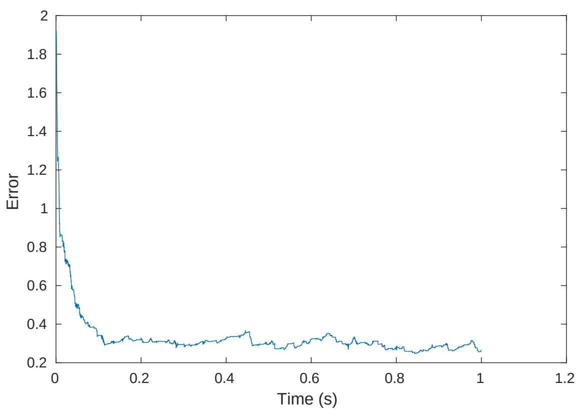
- Introduce the idea of the refutation.
- Give a theoretical description why the type of learning should be relatively unimportant.
- Compare plots (WW^T, Error vs Time, Burst History).



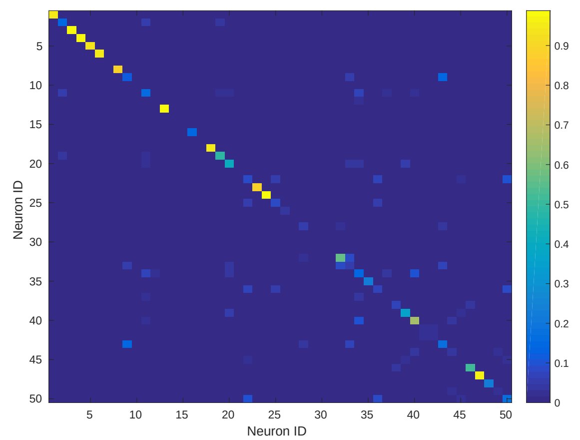
(a) Multiplied weight matrix with Hebbian learning



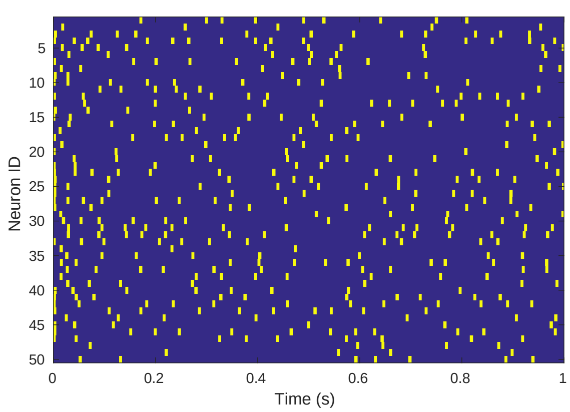
(b) Multiplied weight matrix with STDP learning



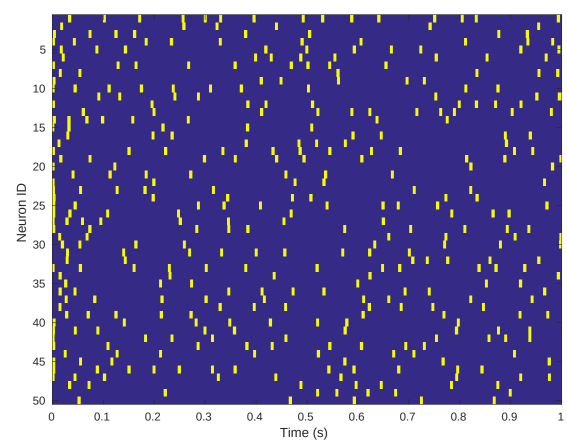
(c) Error over time of Hebbian plot



(d) Error over time of STDP plot



(e) Burst history of Hebbian plot



(f) Burst history of STDP plot

Figure 7: Compare STDP to Hebbian

4 Discussion

Further improvements that could be made to our model and where this research could be taken.

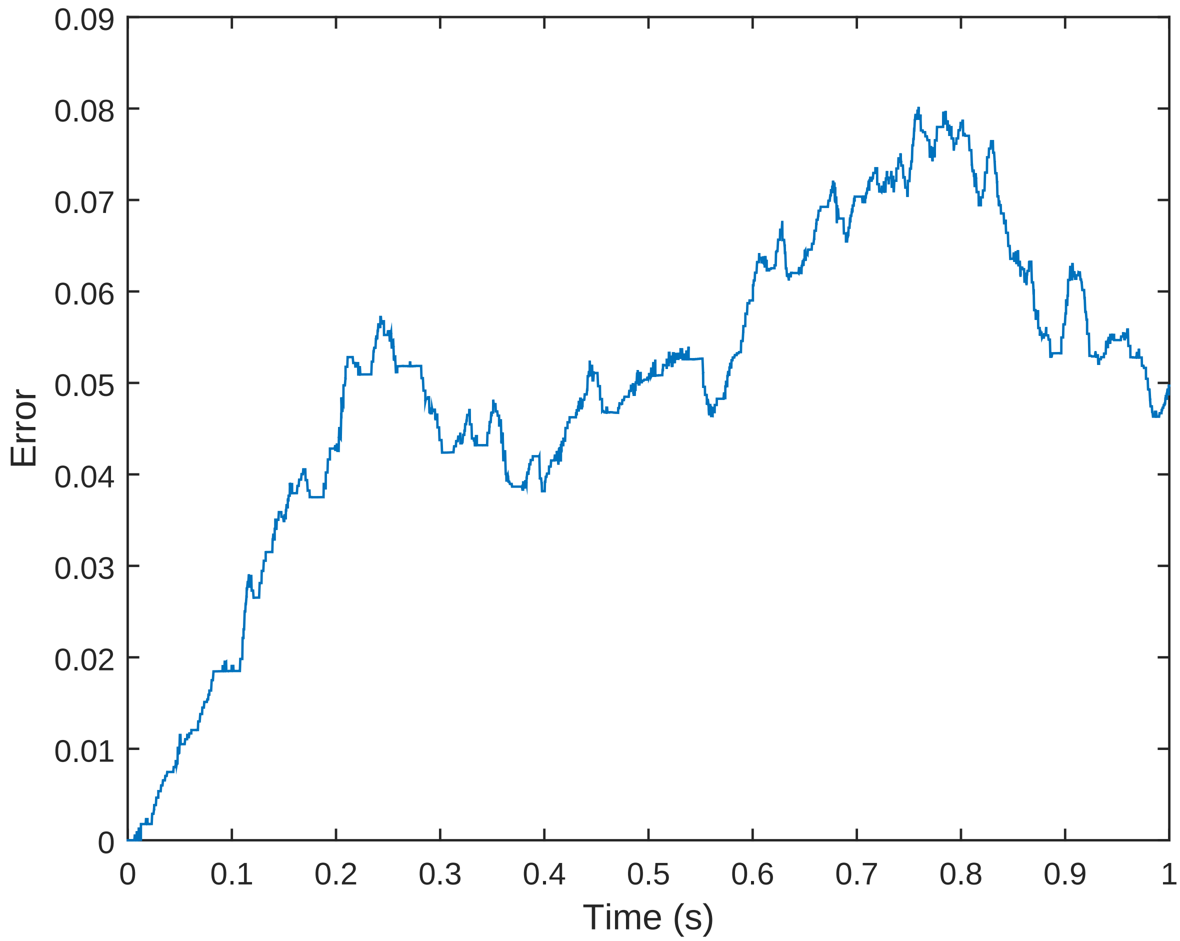


Figure 8: Here is a plot of error over time which was computed using the corrected algorithm presented in the methods section.

5 Summary

Quick summary of our results and everything.

References

[1] Ila R. Fiete, Walter Senn, Claude Z.H. Wang, and Richard H.R. Hahnloser. Spike-time-dependent plasticity and heterosynaptic competition organize networks to produce long scale-free sequences of neural activity. Neuron, 65(4):563 – 576, 2010.