A general error-based spike-timing dependent learning rule for the Neural Engineering Framework

Trevor Bekolay

Monday, May 17, 2010

Abstract

Previous attempts at integrating spike-timing dependent plasticity rules in the NEF have met with little success. This project proposes a spike-timing dependent plasticity rule that uses local information to learn transformations between populations of neurons. The rule is implemented and tested on a simple one-dimensional communication channel, and is compared to a similar rate-based learning rule.

1 Introduction

Principle 2 of the neural engineering framework (NEF) describes an analytical method of determining connection weight matrices between populations of neurons. [1] In doing so, it renders moot the majority of the practical uses of learning in a neurobiological simulation. However, as noted in section 9.5 of [1], learning can help fine-tune models generated with the NEF, and can highlight the strengths and weaknesses of the approach. Further, as a function observed in neurobiological systems, learning is a function that a biologically plausible neural architecture must be able to perform.

Integrating learning in the NEF is done in a straightforward manner directly analogous to biological systems. The ω matrix, which represents synaptic connection weights, is manipulated by a learning rule. Changes in synaptic strength are the basis of the leading theories explaining biological learning and memory. Learning rules that have been implemented in the NEF to this point have been done at the rate level, despite experimental literature that shows that precise spike times play an important role in learning.

This paper aims to implement a learning rule that operates at the level of spikes, taking into account the precise spike times emitted by individual neurons. More ambitiously, it aims to bring the explanatory power of the NEF to a spike-timing dependent learning rule that operates mechanistically, without regard to how the changes in synaptic strength affect what the system is representing.

This paper is organized according to the method described in chapter 1 of [1]: section 2 describes the system being used to test the learning rule, section 3 further specifies the design of the system, and section 4 describes how the system was implemented and presents the results of running the simulation. Section 5 will reflect on those results and propose future work.

2 System description

Much like chapter 9 of [1], this paper focuses on theoretical ideas – how to integrate the spike-timing dependent learning rule – and applies it to a simple “toy” problem. For this reason, the system does not draw on experimental data. The system used to test the learning rule is a simple scalar communication channel, with a recurrent error signal generated by taking the difference between the input and output populations (see figure 1). The system is not based on any one biological system, though error signals are known to be present in many brain systems.

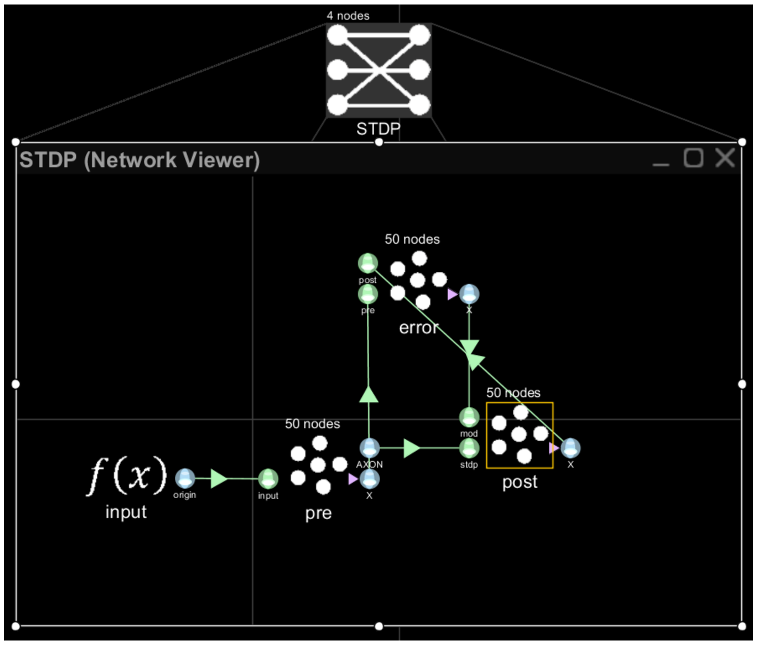


Figure 1: The system used to test the learning rule.

The communication channel is made up of two populations of 50 neurons created with the same properties – the default settings used by Nengo's NEF Ensemble Factory. The input population is called 'pre' and the output population is called 'post'. An 'error' population, also composed of 50 neurons created with the default settings of Nengo's NEF Ensemble Factory, represents the difference between the value encoded by 'pre' and the value encoded by 'post'.

For reference, by default the NEF Ensemble Factory creates leaky-integrate-and-fire (LIF) neurons with $\tau^{RC} = 0.02$, $\tau^{ref} = 0.002$, maximum firing rate uniformly distributed between 200 and 400 Hz, and x-intercept uniformly distributed between -0.9 and 0.9. The ensemble is tuned to represent values over the domain of -1 to 1.

An input function, called 'input', projects to 'pre'. The system is tested with both a sine wave and Gaussian white noise as input functions. Both 'pre' and 'post' project to the 'error' population, which projects back to the 'post' population. Finally, the 'pre' population projects to the 'post' population, and it is this projection that the learning rule is applied to.

The projection between 'pre' and 'post' is intended to be a communication channel, so the learning rule attempts to learn connection weights such that the value encoded by 'post' is the same as the value encoded by 'pre'.

2.1 The learning rule

The learning rule used in this work is a combination of the rate-based learning rule described by MacNeil & Eliasmith [2] and the STDP rule described by Pfister & Gerstner [6]. It is a general rule that only uses local information to learn a set of connection weights – that is, it is *Hebbian*. One of the pieces of local information is an error signal supplied by recurrent connections.

In [2], MacNeil & Eliasmith describe a rule inspired by retinal slip information in the neural integrator. Their rule can be expressed as:

$$\Delta\omega_{ij} = \kappa\alpha_j\tilde{\phi_j}ea_i$$

where:

- κ is a constant learning rate
- α_j is the gain associated with neuron j (j indexes the output population)
- $\tilde{\phi_j}$ is the encoding vector associated with neuron j
- e is the error (difference between current and desired output)
- a_i is the activity of the input neuron i.

This rule operates on a rate-based level; to operate in spikes, the obvious change from activities to sums of spikes would result in the following rule:

$$\Delta\omega_{ij}(t) = \sum_n \kappa\alpha_j\tilde{\phi_j}e\delta(t - t_n)$$

where $\delta(t - t_n)$ is a spike function.

However, this rule did not work when attempted at the outset of this project.

One reason the spike-based rule did not work may lie in the underlying principle behind spike-timing dependent plasticity (STDP), that the precise time of presynaptic and postsynaptic spikes determines how the synaptic strength changes.

In [6], Pfister & Gerstner describe a rule designed to fit data obtained by certain specialized plasticity experiments. They extend the traditional idea of STDP, that a presynaptic spike followed closely by a postsynaptic spike potentiates a synapse and the reverse depresses it, by introducing two additional rules involved with triplets of spikes. Specifically, they found that adding an additional depression with pre-post-pre triplets and additional potentiation with post-pre-post triplets produced a model that fit experimental data well. This rule was chosen because previous implementations of a pair-based STDP rule have not met with favourable results when integrated into the NEF.

Unlike the rate-based rule above, which would be applied on each timestep to all elements in the ω matrix, the STDP rule is intended to be applied when a spike occurs. When a presynaptic spike occurs, the rule is:

$$\Delta\omega_{ij}(t^{pre}) = -e^{\frac{-(t^{pre} - t^{post_1})}{\tau_-}}\left[A_2^- + A_3^- e^{\frac{-(t^{pre} - t^{pre_2})}{\tau_x}}\right]$$

where:

- t^{pre} is the time that the current presynaptic spike occurred
- t^{post_1} is the time that the last postsynaptic spike occurred
- t^{pre_2} is the time that the last presynaptic spike occurred
- A_2^- and A_3^- are constants that control how much to weight the pair-based rule and the triplet-based rule respectively
- τ_- and τ_x are constants that control the shape of the decaying exponential for the weight given to the pair-based rule and the triplet-based rule respectively

When a postsynaptic spike occurs, the rule is:

$$\Delta\omega_{ij}(t^{post}) = e^{\frac{-(t^{post}-t^{pre_1})}{\tau_+}} \left[A_2^+ + A_3^+ e^{\frac{-(t^{post}-t^{post_2})}{\tau_y}} \right].$$

The variables should be clear based on the variables used in the presynaptic rule.

While the rate-based rule does not take into account the relative spike times, the STDP rule does not take into account how a change in synaptic strength changes the value represented by an ensemble of neurons. For example, if the same rule is applied to two pairs of neurons that are identical except for the encoding vector of the output neuron, which is [-1] for one neuron and [1] for the other, then there will be no net effect on the decoded value in the output population, no matter how much the connection weights change.

We can put the two rules together by using the STDP rule in place of a_i in the original rate-based rule. This may introduce a scaling issue, but that is easily remedied by changing κ, which likely would have to be changed in a spike-based implementation anyway.

The rule for a presynaptic spike is then:

$$\Delta\omega_{ij}(t^{pre}) = \kappa\alpha_j\tilde{\phi_j}e\left(-e^{\frac{-t(^{pre}-t^{post_1})}{\tau_-}} \left[A_2^- + A_3^- e^{\frac{-(t^{pre}-t^{pre_2})}{\tau_x}} \right] \right).$$

The rule for a postsynaptic spike is:

$$\Delta\omega_{ij}(t^{post}) = \kappa\alpha_j\tilde{\phi_j}e\left(e^{\frac{-(t^{post}-t^{pre_1})}{\tau_+}} \left[A_2^+ + A_3^+ e^{\frac{-(t^{post}-t^{post_2})}{\tau_y}} \right] \right).$$

Putting the two rules together gives us a general learning rule that is spike-timing dependent. It takes into account the precise timings of presynaptic and postsynaptic spikes, and effects change in accordance with how the NEF encodes and decodes information.

3 Design specification

As a system designed specifically as a minimal system for testing the learning rule, none of the variables are constrained by any particular real-world limitations.

However, since we are dealing with manipulating the connection weight matrix, it is important to note that, while the matrix is analogous to the synaptic strengths of real synapses, there are important differences. In particular, the ω matrix accepts any value, with no constraints. There are a number of areas in which this does not match real-world observations.

The most glaring real world constraint is that biological neurons are either inhibitory or excitatory – that is, the connection weight matrix should contain only positive values or only negative values. In section 6.4 in [1] and in [5] this issue is addressed, and a method for changing populations with positive and negative connection weight matrices to a population with only positive connection weights and introducing a inhibitory population of "interneurons" is proposed. Since it is possible to change from the mixed system to one with segregated excitatory and inhibitory populations, we will ignore this real-world limitation, and leave analysis of the rule under those conditions as future work.

Another real-world limitation that may play a part is the relative strength of one synapse compared to another synapse on the same dendritic branch, or on the same neuron. One experimental study showed that synapses increase in strength through discrete steps based on the addition or removal of AMPA receptors in the synapse. [3] If this is the case, there are likely to be limitations on the range of values that could be present in the connection weight matrix. Approaches to handling this limitation could include enforcing a maximum

Parameter	A_2^+	A_3^+	A_2^-	A_3^-	τ_+	τ_-	τ_x	τ_y
Value	8.8×10^{11}	5.3×10^{-2}	6.6×10^{-3}	3.1×10^{-3}	16.8 ms	33.7 ms	714 ms	40 ms

Table 1: STDP parameters from [6]

synaptic strength, or normalizing the ω matrix. The implementation of the rate-based learning rule described previously uses Oja's rule to normalize synaptic strength (see [4] for details), but our spike-based learning rule will not be normalized.

4 Implementation

The system was modelled in Nengo using python scripts to generate and run the systems. The same network was tested with sine wave and Gaussian white noise inputs, using the rate-based rule (from [2]) and the spike-timing dependent rule proposed in section 2. The parameters, A_2^+, τ_x, etc., are the values used for the visual cortex data set in [6] (summarized in Table 1). Connection weights between the 'pre' and 'post' populations are initially randomized, with the random value for each synapse multiplied by the encoder of the postsynaptic neuron.[1]

Initial tests were largely unsuccessful, and determining the cause was difficult given the tools available in Nengo. For that reason, two additional graphs were added to the Interactive Plots feature in Nengo.

The first graph is a grid designed to visualize the current values stored in the ω matrix (see figure 2 (left)). The existing code used to display voltage grids was extended by differentiating between negative and positive values with colours (blue and red respectively), because connection weights can be both negative and positive.

The second graph is a line graph representing the connection weight between a presynaptic and postsynaptic neuron, overlaid on a spike raster displaying the spikes of the two neurons (see figure 2 (right)).

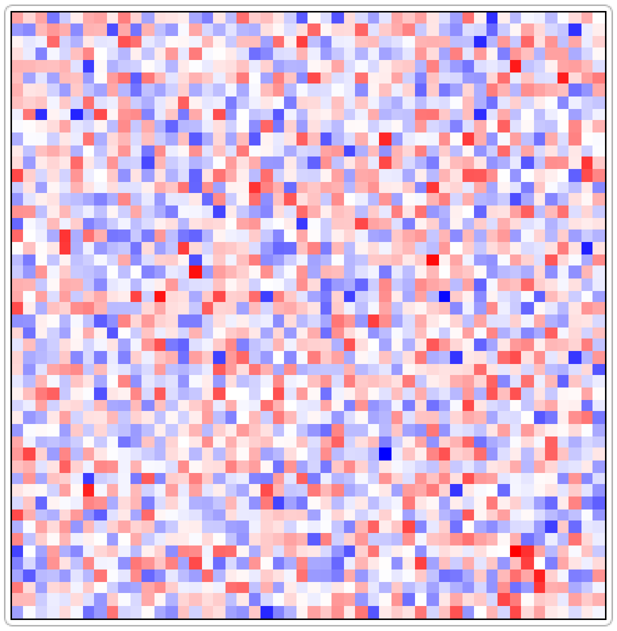
The second graph revealed that connection weights were changing inappropriately; this turned out to be due to a bug in the implementation of plasticity rules in Nengo. The same PlasticityRule object was being applied to each of the neurons in the output population. The object tracks the time of the previous postsynaptic spike, so if any of the fifty neuron in 'post' were firing, then the time of the previous postsynaptic spike would be updated for all neurons, even if they had not spiked. After fixing this bug, the general spike-timing dependent rule started working. [2]

Simulations were run to compare the efficacy of the rate-based rule compared to the spike-based rule. Runs of 10 seconds were done for the sine wave input and the Gaussian white noise input. Results from the simulations done in spikes are filtered by convolving the raw data with a decaying exponential with $\tau = 0.01$. The interesting sections of the runs are shown in figures 3 to 6. Immediately obvious is that both rules were able to learn the communication channel well for both inputs.

To objectively compare the learning rules, we calculated the root mean squared error (RMSE) between the input population and output population's representations of values along the entire radius that the populations can represent. This gives us a good idea of how well the channel is communicating the signal across the entire radius that it can represent. Each system was simulated 10 times and the resulting RMSE values averaged. Bootstrapped 95% confidence intervals were computed and plotted. The results are shown in figure 7 and 8. For the Gaussian white noise input, the rate-based and spike-based rules do similarly well throughout the 10 second run time. For the sine wave input, the spike-based rule starts off worse than the

[1] This is partly because connection weights were initially constrained to not change signs. This constraint was later lifted.

[2] The bug was fixed by saving the spike times of each output neuron separately, and then passing the index of the current neuron to various functions. This fix is not a robust one, and the implementation will have to be reviewed before it is committed to the general Nengo codebase.



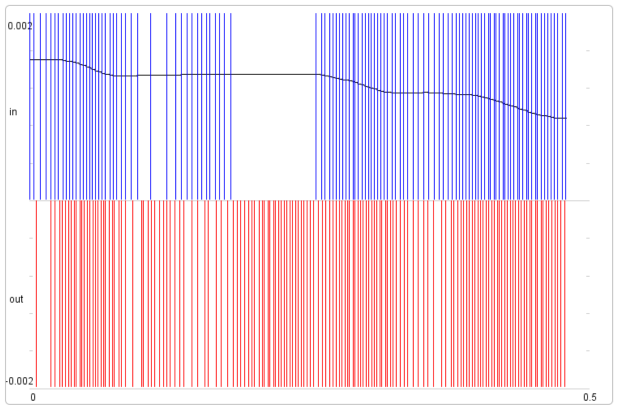


Figure 2: (Left) Visualization of the current connections weights – blue represents a negative weight, red positive. (Right) Visualization of the connection weight between a single pair of neurons, overlaid on a spike raster of the two neurons.

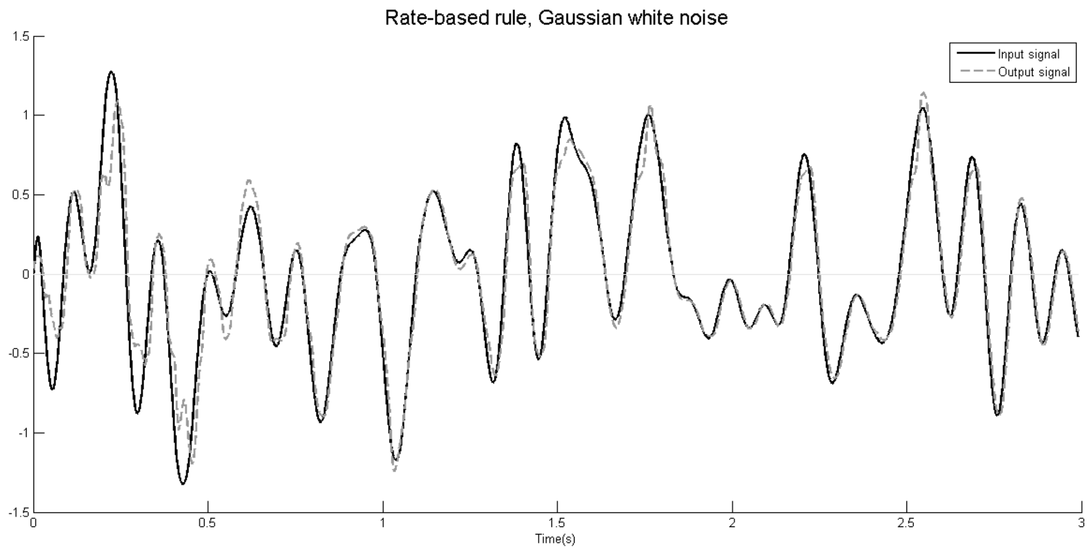


Figure 3: 3 seconds of the run of Gaussian white noise using the rate-based rule.

rate-based rule for the first 0.5 seconds, then does consistently better than the rate based rule.

There is, then, a difference between the performance of the two rules. To further examine this difference, we can examine the connection weight matrices before and after running a simulation. Figures 9 and 10 show the connection weight matrices before and after learning for the spike-based and rate-based rules, respectively.[3] Note that the presynaptic neurons change along the horizontal axis, and the postsynaptic neurons change along the vertical axis (i.e. all connections projecting to the first postsynaptic neuron are

[3]Gaussian noise input was used to create these graphs, though similar results were found with sine wave input.

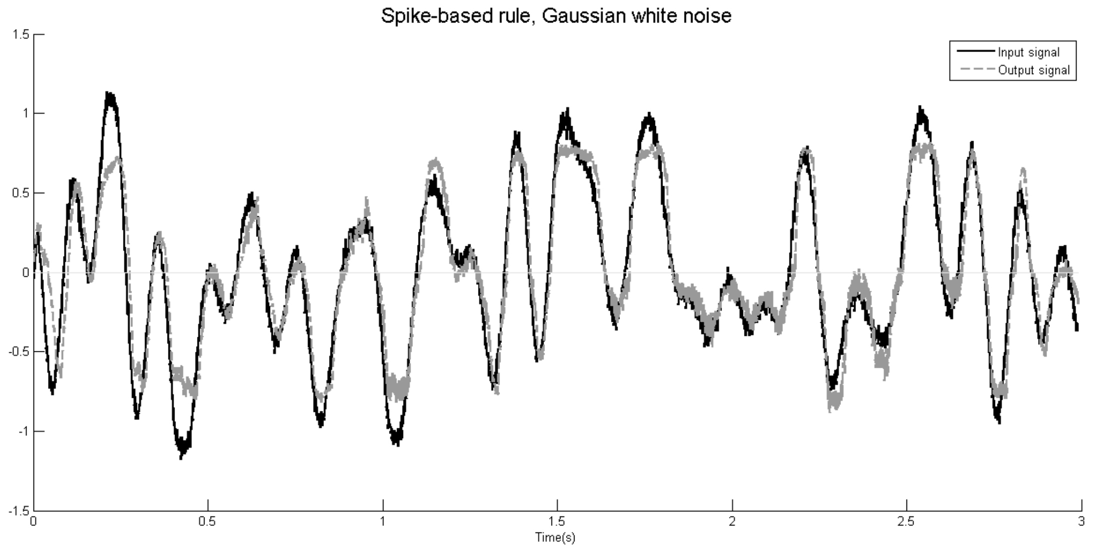


Figure 4: 3 seconds of the run of Gaussian white noise using the spike-based rule.

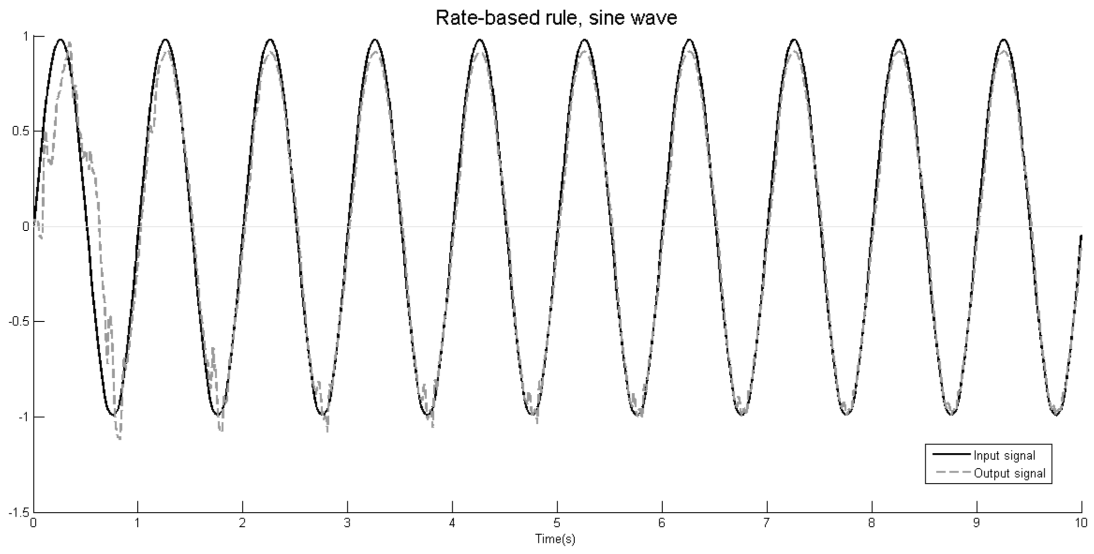


Figure 5: 10 seconds of the run of a sine wave using the rate-based rule.

represented in the first row, and all connections projecting from the first presynaptic neuron are represented in the first column).

The first observation is that the connection weight matrices after learning are completely different, despite using a similar rule, and performing similarly well. Second, the connection weight matrix after learning with the rate-based rule is much sparser. Third, elements in columns are clearly affected in a similar way with the rate-based rule. Since columns represent projections from presynaptic neurons, this means that when the rate-based rule changes the influence of one input neuron, it does so across all of the output neurons.

While some of these changes are due to the rate-based rule using Oja normalization, the most significant difference is that the spike-based rule operates on each synapse independently, while the rate-based rule changes groups of synapses in similar ways.

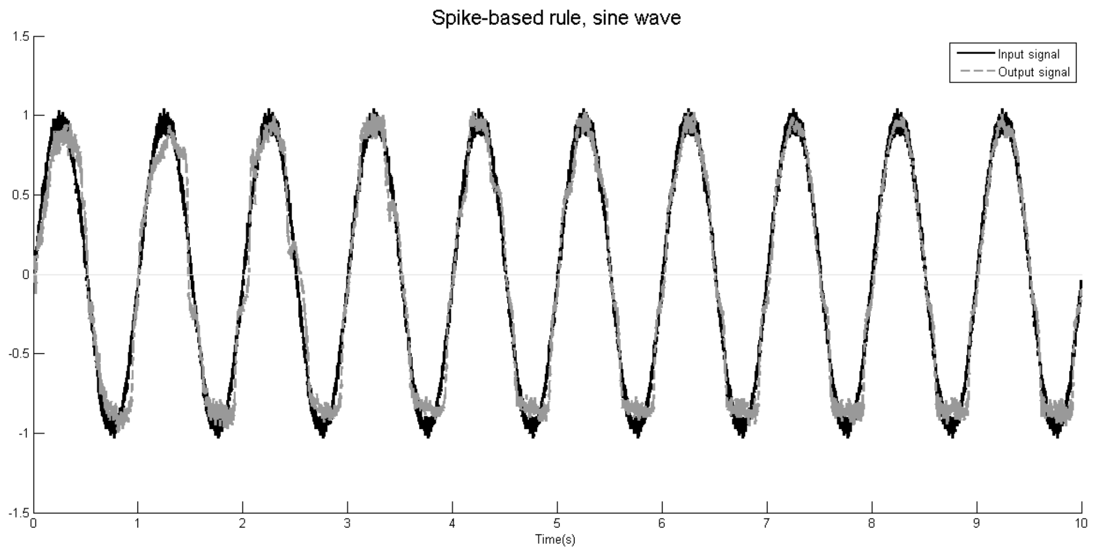


Figure 6: 10 seconds of the run of a sine wave using the spike-based rule.

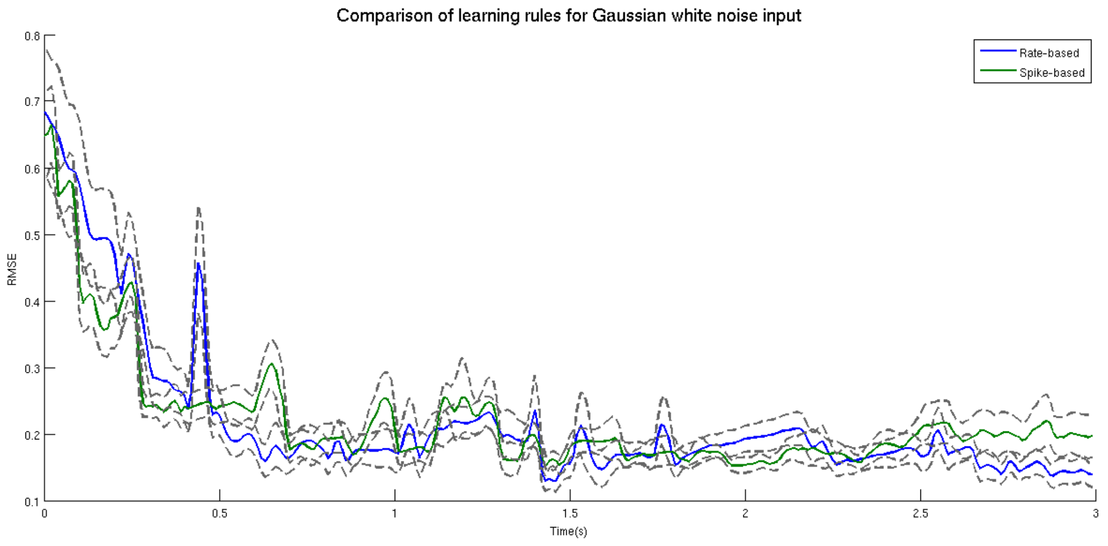


Figure 7: Comparing the rate-based and spike-based learning rules on Gaussian white noise input, averaged over 10 iterations each.

To see why this is, consider one timestep of simulation. For the rate-based rule $\Delta\omega_{ij} = \kappa\alpha_j\tilde{\phi}_j e a_i$, each synapse is changed (unless the input activity or error terms are zero). When the activity of an presynaptic neuron is low compared to other presynaptic neurons, the connection weight for that presynaptic neuron **at each postsynaptic neuron** will decrease at a similar rate (scaled by $\alpha_j\tilde{\phi}_j$). For the spike-based rule, even with Oja normalization, if either of the postsynaptic or presynaptic neuron associated with a connection is not spiking, then the connection weight will not change at all. While the weights of active connections – those whose presynaptic and postsynaptic cells are frequently firing – change rapidly, connections where either neuron is not firing – or they are firing at entirely different times – do not change. Many of the connections on the right grid in figure 9, then, are simply unchanged from their initial randomized value because the spike times of the presynaptic and postsynaptic cells are not correlated. While the rate-based rule produces sparse connection weight matrices in the end, the spike-based rule is *applied* sparsely, affecting

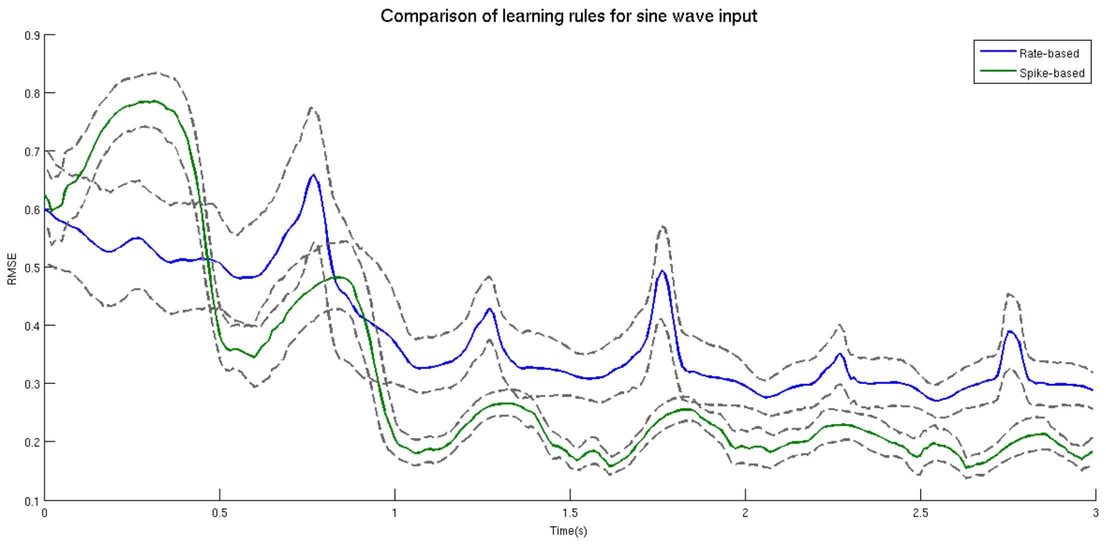


Figure 8: Comparing the rate-based and spike-based learning rules on sine wave input, averaged over 10 iterations each.

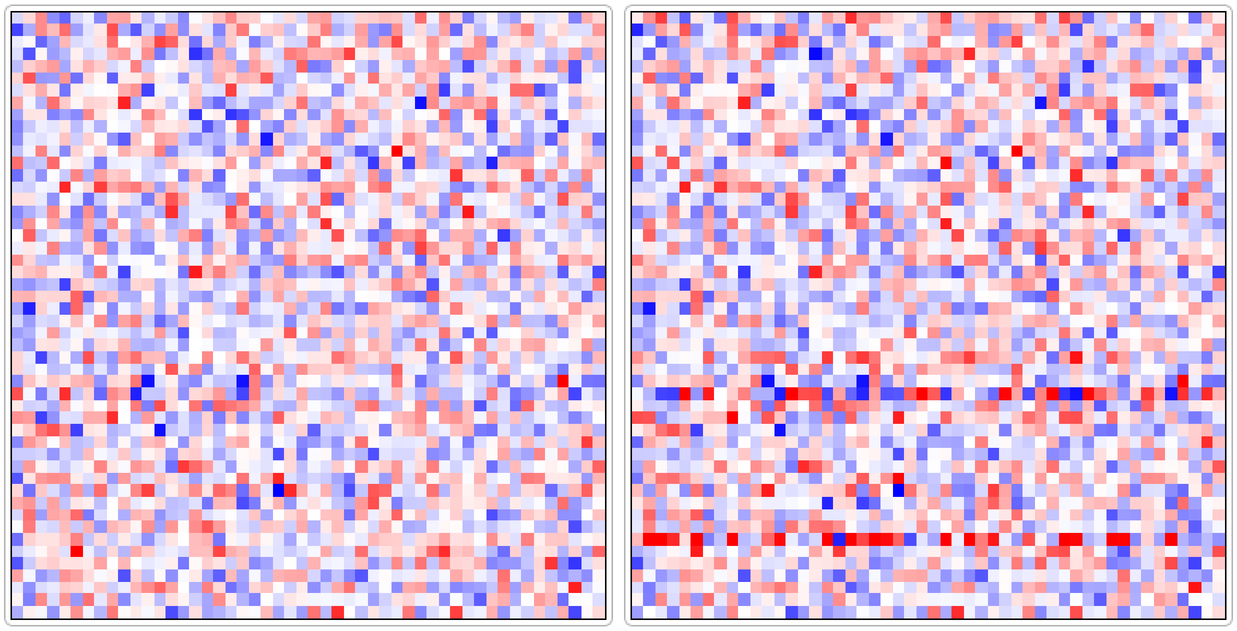


Figure 9: Visualization of connection weight matrix for the spike-based rule before (Left) and after (Right) a 3 second run using Gaussian white noise input.

only a small subset of connection weights. This additional flexibility may be why the spike-based rule is able to perform better on the sine wave input.

4.1 The importance of precise spike-timing

While the parameters supplied by Pfister & Gerstner in [6] learned the communication channel well, we wondered what effect changing the parameters would have. In most cases, the change was minor, but it

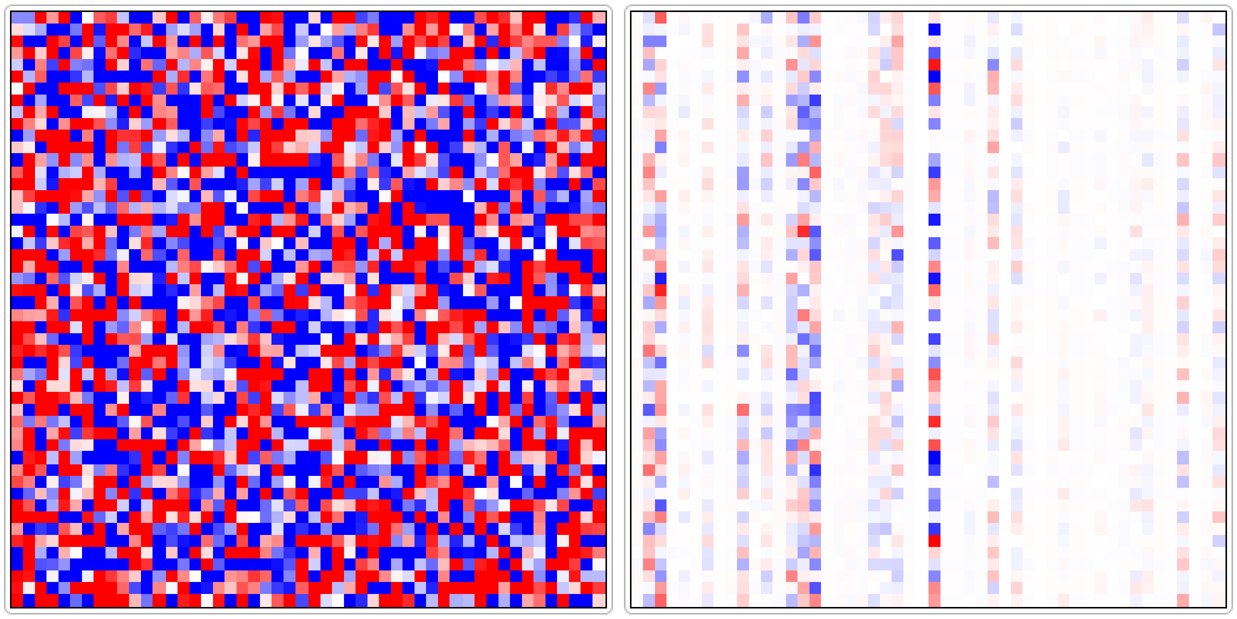


Figure 10: Visualization of connection weight matrix for the rate-based rule before (Left) and after (Right) a 3 second run using Gaussian white noise input.

was observed that changing A_2^- and A_3^- such that they were larger than A_2^+ and A_3^+ caused the rule to fail entirely. Further investigation revealed that using only the presynaptic rule learned the communication channel properly but flipped on the horizontal axis; that is, the rule scaled the the input by -1 rather than 1 as desired. Making the presynaptic rule positive (i.e. removing the negative sign on the first exponential term) allowed the rule to learn the communication channel as expected, even in the absence of the postsynaptic spike rule. The communication channel learned similarly well using only the presynaptic rule.

Experiments were then run to determine the efficacy of the spike-based rule using the parameters from the paper, but with a positive presynaptic rule (so that the spike-timing term in the rule is always positive). An example and comparison of RMSE values is included as figures 11 to 14.

In general, the rule using all positive terms performs at least as well as the original spike-based rule. Of particular interest is figure 13, in which the rule using all positive terms does consistently better than the original spike-based rule.

Experiments were then run to determine the relative contribution of each element of the rule. Four tests were run, in each setting all of A_2^-, A_3^-, A_2^+ and A_3^+ to zero except for one term which is set to the arbitrary amount 5×10^{-3}. The results are included as figures 15 to 24.

Recall that A_2^- represents the contribution of the presynaptic pair rule, which is applied when a presynaptic spike occurs closely after a postsynaptic spike, and A_3^- represents the contribution of the presynaptic triplet rule, which is applied when a presynaptic spike occurs closely after a postsynaptic spike that occurred closely after a presynaptic spike. A_2^+ and A_3^+ represent the contributions of the corresponding postsynaptic rules.

All four tests yielded good results, with every test learning the communication channel well. Looking at the comparisons of RMSE in figures 23 and 24 reveals that using only the presynaptic rule can actually improve performance, though a rigorous exploration of other factors, such as the learning rate, has not been performed, so these results should not be considered conclusive.

The general idea, however, is that the order of the pre and post spikes do not matter when integrating spike-timing dependent rules in the NEF; all that matters is that the spike-times of a presynaptic neuron

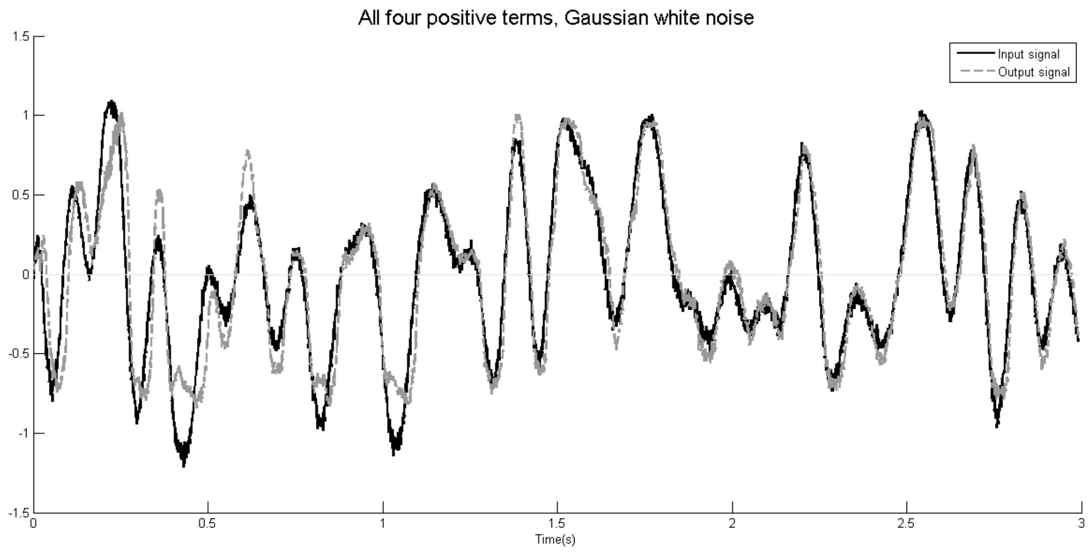


Figure 11: 3 seconds of the run of Gaussian white noise using the all four terms contributing positively to the spike-based rule.

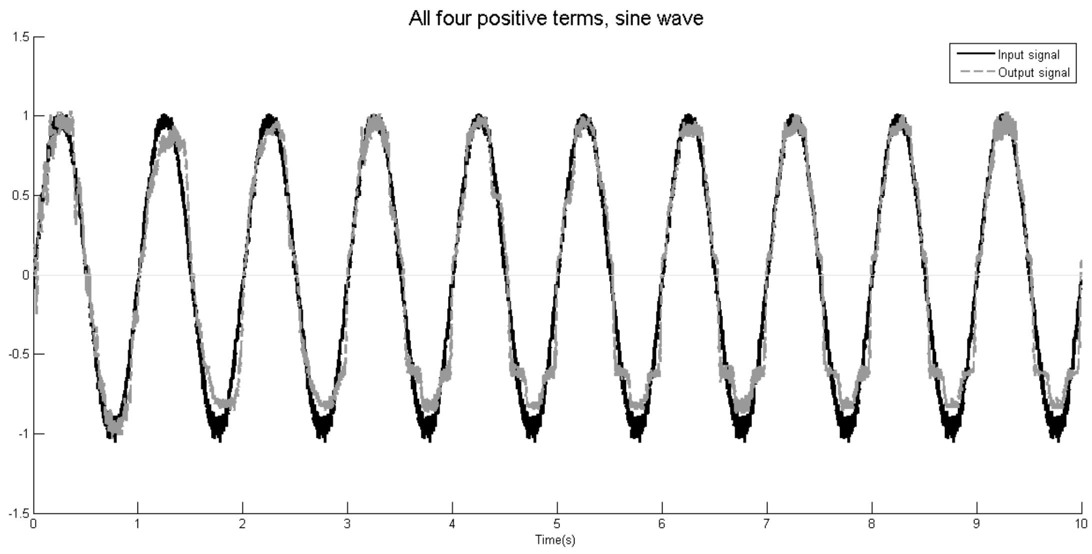


Figure 12: 10 seconds of the run of the sine wave using the all four terms contributing positively to the spike-based rule

are highly correlated with the spike-times of a postsynaptic neuron.

5 Discussion and future work

The surprising result that a spike-timing dependent learning rule can work, but does not depend on order, exposes a difference between simulations created with the NEF and real biological systems. STDP learning rules are characterized by synaptic depression when a presynaptic spike occurs directly after a postsynaptic spike, and yet in the NEF the same rule can apply whether the presynaptic spike comes directly before or

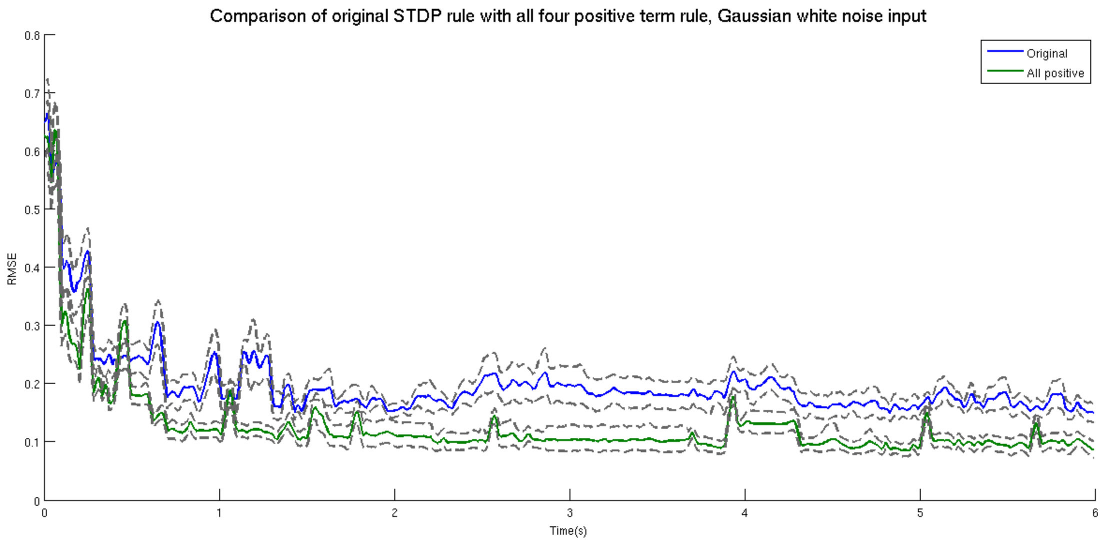


Figure 13: Comparing the original and all positive spike-based learning rules on Gaussian white noise input, averaged over 10 iterations each.

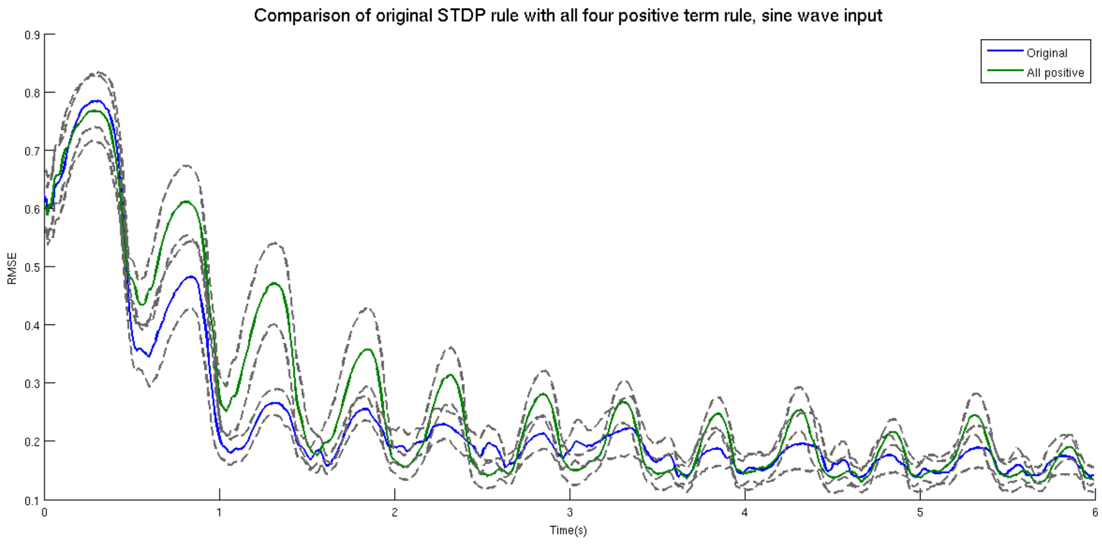


Figure 14: Comparing the original and all positive spike-based learning rules on sine wave input, averaged over 10 iterations each.

directly after a postsynaptic spike. That the STDP rule described in [6] worked was due to the positive postsynaptic rule having parameters that were significantly larger than the negative presynaptic rule.

In light of this, the rules discussed in section 4.1, which gave results as good as or better than the rate-based rule and the original spike-based rule, should be described as spike-timing based implementations of the generalized learning rule from [2] rather than "STDP," as they do not follow the traditional STDP rules.

Further, this also suggests that the main contribution of this project is perhaps the bug that was uncovered and fixed in Nengo, described at the beginning of section 4. The results gathered after fixing that bug showed that a variety of spike-timing based rules, with a wide range of parameters, were able to learn the communication channel. It would be an illuminating exercise to implement the rule mentioned at the

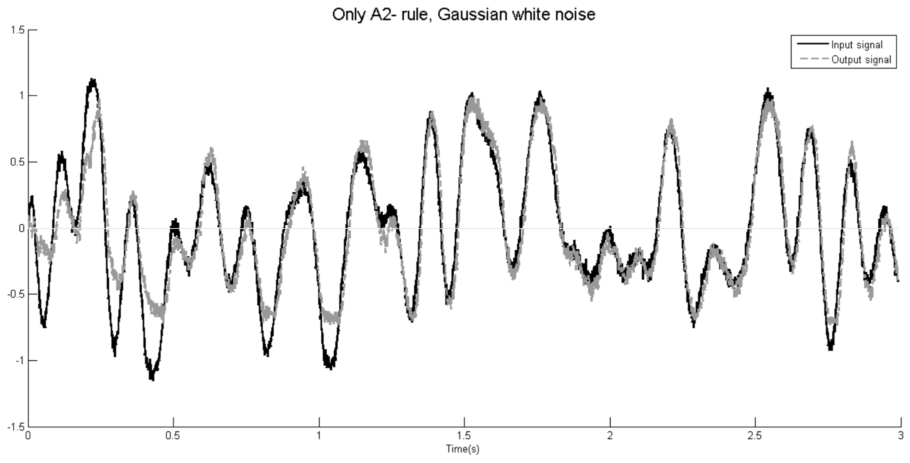


Figure 15: 3 seconds of the run of Gaussian white noise using the A_2^- spike-based rule only.

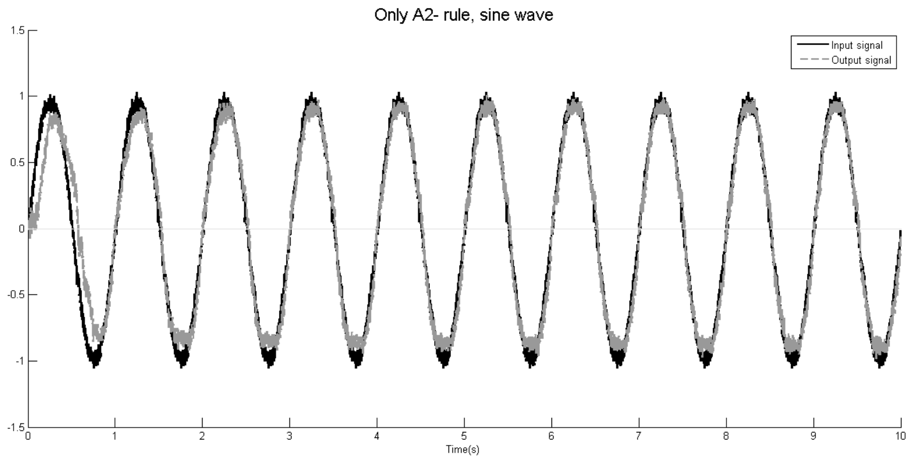


Figure 16: 10 seconds of the run of the sine wave using the A_2^- spike-based rule only.

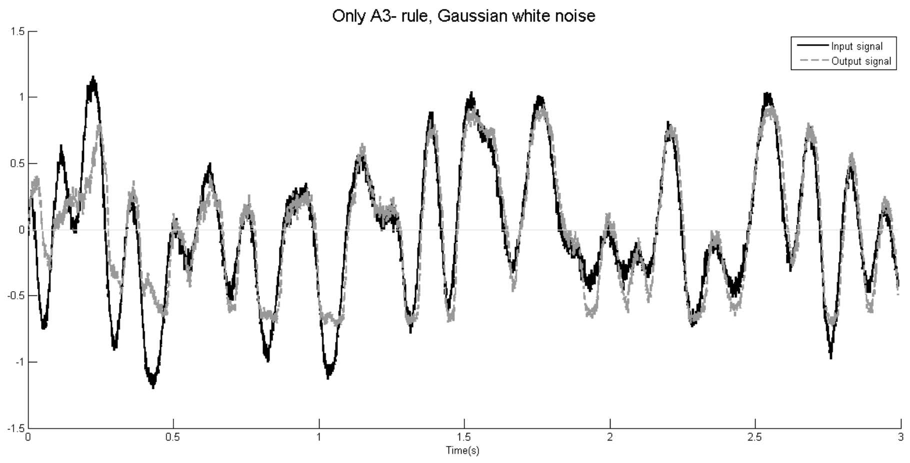


Figure 17: 3 seconds of the run of Gaussian white noise using the A_3^- spike-based rule only.

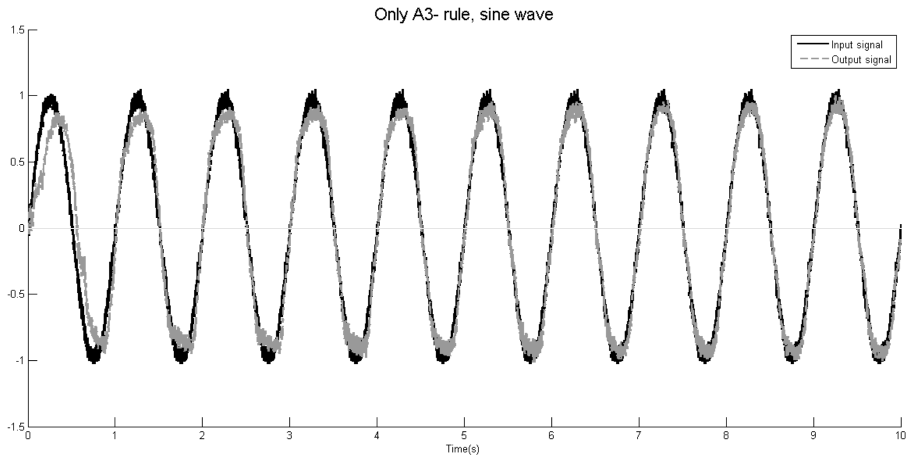


Figure 18: 10 seconds of the run of the sine wave using the A_3^- spike-based rule only.

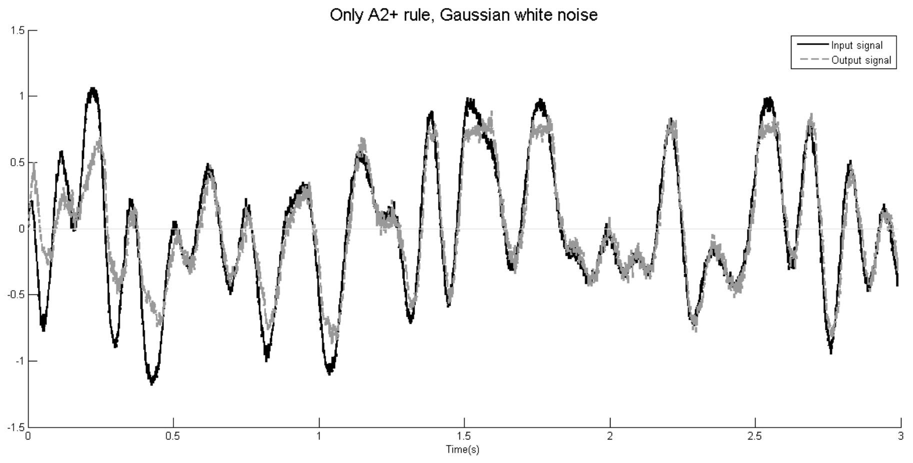


Figure 19: 3 seconds of the run of Gaussian white noise using the A_2^+ spike-based rule only.

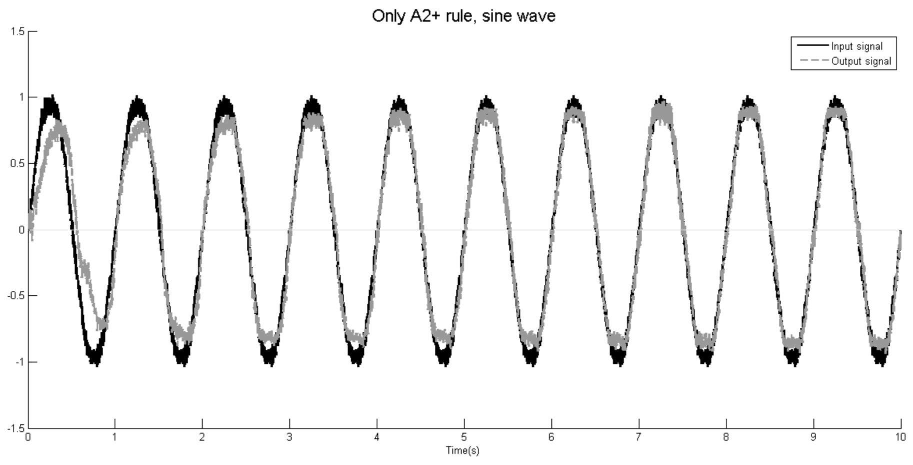


Figure 20: 10 seconds of the run of the sine wave using the A_2^+ spike-based rule only.

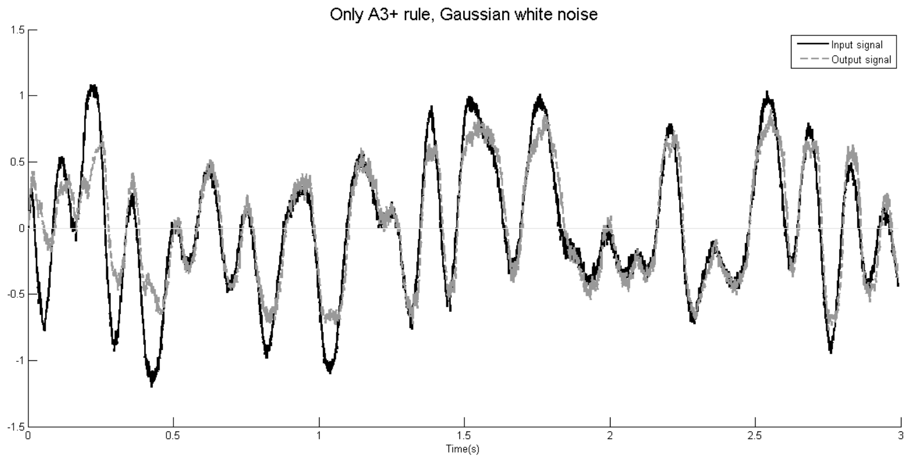


Figure 21: 3 seconds of the run of Gaussian white noise using the A_3^+ spike-based rule only.

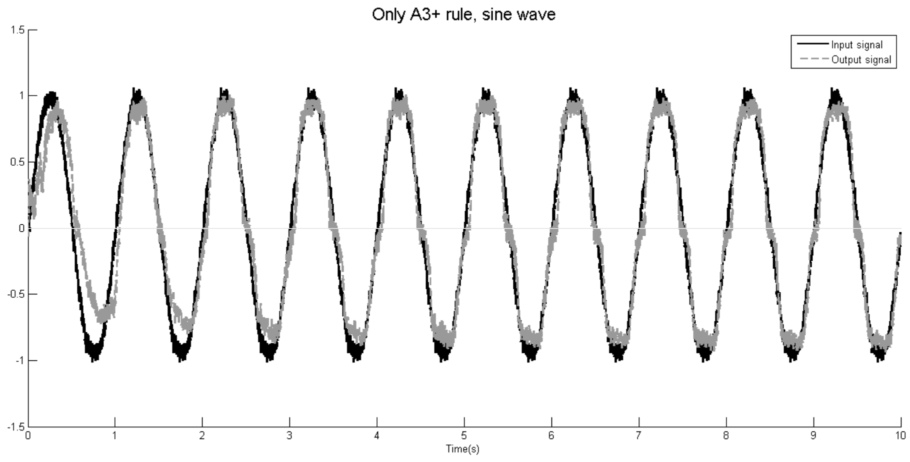


Figure 22: 10 seconds of the run of the sine wave using the A_3^+ spike-based rule only.

beginning of section 2.1, as fixing this bug may have made this more straightforward rule successful.[4]

Regardless, implementing a successful spike-timing dependent learning rule opens up a wide variety of possible future work.

First, the experiments outlined in this project were all done on a toy problem that learned a one dimensional communication channel. More complicated systems must be experimented with – systems encoding many dimensions with complex dynamics, and so on. At the moment, the changes made to Nengo to allow the spike-timing rules to work are not robust or scalable to high dimensions; making them so should be the top priority.

Further study into the parameters described in the model may reveal more interesting properties about the NEF or about the learning rule. What are the ranges of values that will produce favourable results in various situations? In particular, while changing the amplitude variables was explored in this paper, the time constants were not changed. Examining extreme values for these time constants, and looking at the kinds of values found in biological experiments, may help refine this rule or create others.

[4]For clarity, the rule being discussed is $\Delta\omega_{ij}(t) = \sum_n \kappa\alpha_j\tilde{\phi_j}e\delta(t-t_n)$.

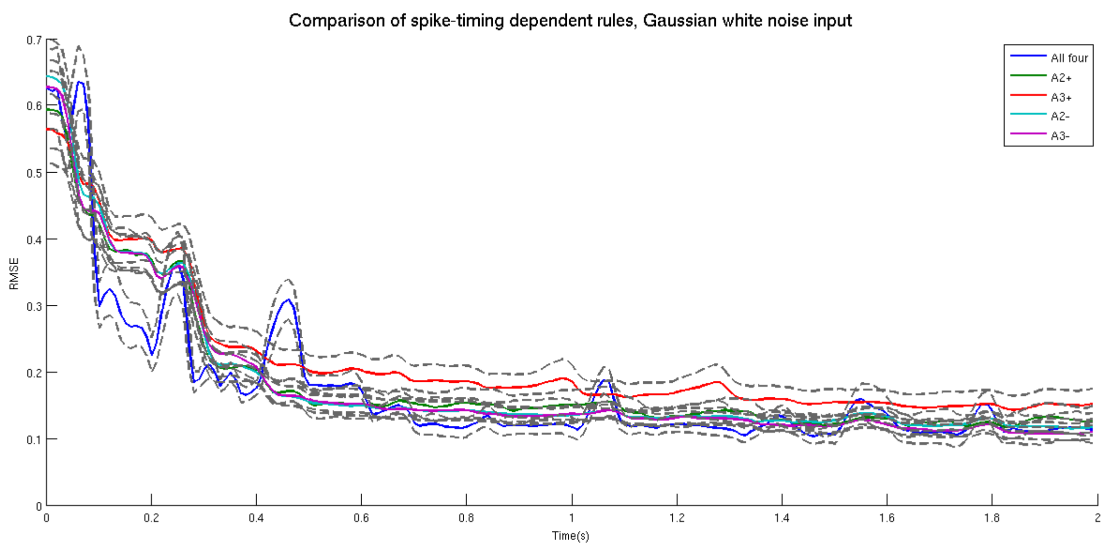


Figure 23: Comparing the various terms' contributions to the spike-based learning rule on Gaussian white noise input, averaged over 10 iterations each.

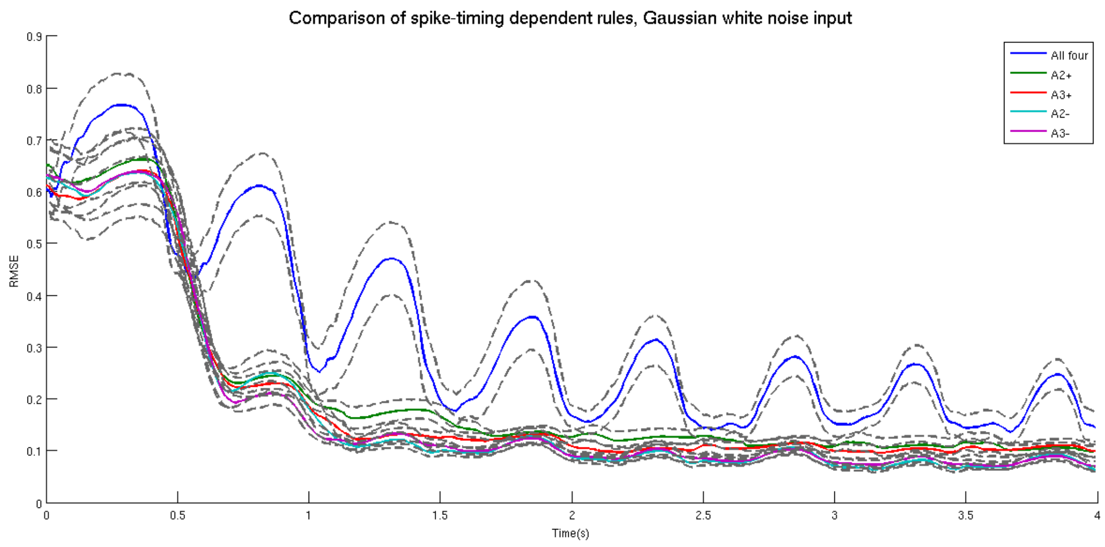


Figure 24: Comparing the various terms' contributions to the spike-based learning rule on sine wave input, averaged over 10 iterations each.

Biological plausibility is one of the main concerns of the NEF, and the plausibility of this learning rule has not been adequately explored. In addition to a thorough review of neurobiological literature, this rule should be tested in a situations where neural populations are constrained to be either excitatory or inhibitory, and connection weight matrices are either completely positive or negative.

A more general review of error signals in the brain should also be performed. The role of interneurons and the reward system in generating these signals is predicted to be an interesting area to investigate.

Now that a spike-based learning rule has been integrated in the NEF, it opens the door to experimenting with other spike-based learning rules. Existing literature has put forth many rules that have been applied in a variety of models; examining how they operate in the NEF may give insight to the NEF and its relation to other models.

References

[1] Chris Eliasmith and Charles H Anderson. *Neural engineering: computation, representation and dynamics in neurobiological systems.* MIT Press, Cambridge, Mass, USA, 2003.

[2] David Macneil and Chris Eliasmith. Fine-tuning and stability of recurrent neural networks. *Neuron (Preprint)*, 2010.

[3] Johanna M Montgomery and Daniel V Madison. Discrete synaptic states define a major mechanism of synapse plasticity. *Trends in neurosciences*, 27(12):744–50, December 2004.

[4] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, November 1982.

[5] Christopher Parisien, Charles H Anderson, and Chris Eliasmith. Solving the problem of negative synaptic weights in cortical models. *Neural computation*, 20(6):1473–94, June 2008.

[6] Jean-Pascal Pfister and Wulfram Gerstner. Triplets of spikes in a model of spike timing-dependent plasticity. *The Journal of neuroscience: the official journal of the Society for Neuroscience*, 26(38):9673–82, September 2006.