university of groningen / faculty of science and engineering

Using memristors in a functional spiking-neuron model of activity-silent working memory

Joppe Wik Boekestijn (s2754215)

July 2, 2021

Internal Supervisor(s): Dr. Jelmer Borst (Artificial Intelligence, University of Groningen)
Second Internal Supervisor: Thomas Tiotto (Artificial Intelligence, University of Groningen)

Artificial Intelligence
University of Groningen, The Netherlands

university of groningen / faculty of science and engineering

Abstract

In this paper, a spiking-neuron model of human working memory by Pals et al. (2020) is adapted to use Nb-doped $SrTiO_3$ memristors in the underlying architecture. Memristors are promising devices in neuromorphic computing due to their ability to simulate synapses of artificial neuron networks in an efficient manner. The spiking-neuron model introduced by Pals et al. (2020) learns by means of short-term synaptic plasticity (STSP). In this mechanism neurons are adapted to incorporate a calcium and resources property. Here a novel learning rule, mSTSP, is introduced, where the calcium property is effectively programmed on memristors. The model performs a delayed-response task. Investigating the neural activity and performance of the model with the STSP or mSTSP learning rules shows remarkable similarities, meaning that memristors can be successfully used in a spiking-neuron model that has shown functional human behaviour. Shortcomings of the Nb-doped $SrTiO_3$ memristor prevent programming the entire spiking-neuron model on memristors. A promising new memristive device, the diffusive memristor, might prove to be ideal for realising the entire model on hardware directly.

1 Introduction

In recent years it has become clear that computers are using an ever-increasing number of resources. Up until 2011 advancements in the semiconductor industry followed 'Moore's law', which states that every second year the number of transistors doubles, for the same cost, in an integrated circuit (IC). Transistors, the building blocks of a computer, have had to become progressively smaller to enable tighter semiconductor integration. Nowadays, transistors are built on the scale of 5 nanometers, thus leading us into the quantum domain, where quantum effects such as quantum tunneling start to be relevant to the precision of computation (Bate, 1988). The end of 'Moore's law' comes at a time where the demand for computational resources is increasing exponentially. Currently, computers, data centers, and networks consume over 10% of global electricity (Mills, 2013).

The combination of the end of 'Moore's law', and the enormous resource usage, calls for a radical new way of computing, which may lower our ecological footprint. Luckily we have an example of a very efficient computational device nearby: namely, our brain. A human brain consumes only about 20 watts and is estimated to be competitive with a modern supercomputer in regards to the floating-point operations per second (FLOPS) it can carry out (Bostrom & Sandberg, 2008).

Traditionally, computers make use of billions of transistors as implementation of Boolean logic. Memristors, first discoverd by Chua (1971), are electrical components, and could be used to implement the same logic. Memristors are a promising device in neuromorphic computing – computing where information is processed in a way that mimics the brain's neural architecture. The synapses in an artificial neural network (ANN) can be realised on memristors. Typically, memristors and ANNs, are pretrained for a specific task, e.g. Yao et al. (2020), B. Li et al. (2014), Ebong & Mazumder (2011). However, if we would want to use memristors in a true cognitive computer, we would need

to create a flexible memory system, which would make the system capable of adapting to a variety of tasks. Since the brain is so efficient, we can use strategies that have evolved as an inspiration to create a cognitive computer.

One of the main memory systems in the brain is working memory (WM). WM is crucial for reasoning and the guidance of decision-making and behaviour, and therefore interesting for creating a cognitive computer. In this paper, memristors will be used in the underlying architecture of an ANN to perform a delayed-response task, which shows that memristors could be used as building blocks in a flexible memory system to perform a cognitive experiment.

The remainder of the introduction discusses transistors, memristors, and ANNs, followed by an introduction of activity-silent working memory. Lastly, an overview of the experiments conducted in this study is provided.

1.1 Transistors

Transistors are semiconductor devices used to amplify or switch electronic signals and electrical power. They are composed of at least three terminals: base (B), collector (C), and emitter (E). For illustrating purposes, a Negative-Positive-Negative (NPN) transistor is shown in Figure 1. The base, which can be thought of as a gate, is responsible for controlling whether current can flow freely through the transistor when power is applied. If the voltage of the base rises, the resistance between the collector and emitter decreases, allowing current to flow. In this state, where the transistor becomes saturated, the transistor is "ON". If the base voltage is low the resistance between the collector and emitter rises allowing no current to flow through the circuit. In this case, the transistor is switched "OFF".

There are some inherent problems with the transistor. Keeping the transistor "ON" requires maintaining it under tension. In the "OFF" state there is a small leakage of current. Since billions of transistors are used in everyday devices, the energy consumption given by normal operation in addition to the efficiencies can be quite demanding. The need for new computing devices becomes clear.

1.2 Memristors and ANNs

There has been an ongoing search for alternatives to the transistor. In the 1970s a new fundamental circuit element was discovered, which was named the memristor (Chua, 1971). A device exhibiting the theorised properties was only fabricated in 2008 at HP Labs (Strukov et al., 2008). Unlike the three-terminal transistor, the memristor only has an input and an output channel and is thus classified

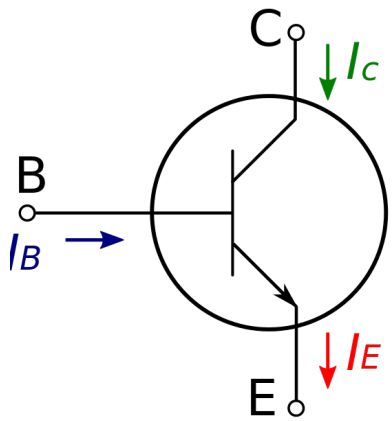


Figure 1: Diagram of NPN transistor. If the voltage of the base (B) increases, it allows the current between the collector (C) and the emitter (E) to flow freely (Moscote, 2013).

as a two-terminal device. The current flowing into the memristor directly changes the resistance of the material, without the need for an additional gate. In an ideal memristor, this resistance does not change when no current is passed through the circuit, thus acting as a "memory" of the past injected current, however, in a practical memristor we have to account for leakages. Specific resistances can be used to indicate an "ON" or "OFF" state for digital computing, but intermediate states can also be used to infer information, as we show in this paper. The fact that the state of the memristor changes due to the inherent properties of the material makes the memristors a very efficient analogue computation device.

1.3 Efficient computation using memristors

A limitation of traditional computing paradigms is the separation between memory and computation. The limited throughput between the central processing unit (CPU) and memory, also called the 'Von Neumann bottleneck', restricts the computational power. An alternative computing paradigm is an ANN, which demonstrates colocation of memory and computation. ANNs are comprised of many entities called neurons that are connected via synapses, as can be seen in Figure 2. The input to the network is transformed by the ANN into a certain output, e.g. an image is transformed into an image classification. If the ANN is used in a supervised fashion in a classification problem, the correct labels describing the data are used as the ground truth. The output is compared to the ground truth, and via a backwards pass, called back-propagation, the synaptic weights are adjusted in order to better represent the correct labels. Although we are not guaranteed that the ANN ends up at the perfect solution, also known as the global minimum, we are ensured that it moves towards one of the possible (sub-)optimal solutions (local minima) (Rumelhart et al., 1986). Consequently, the synaptic weights in the model learn to represent the desired input to output mapping, known as the

transformation function, effectively combining memory and computation.

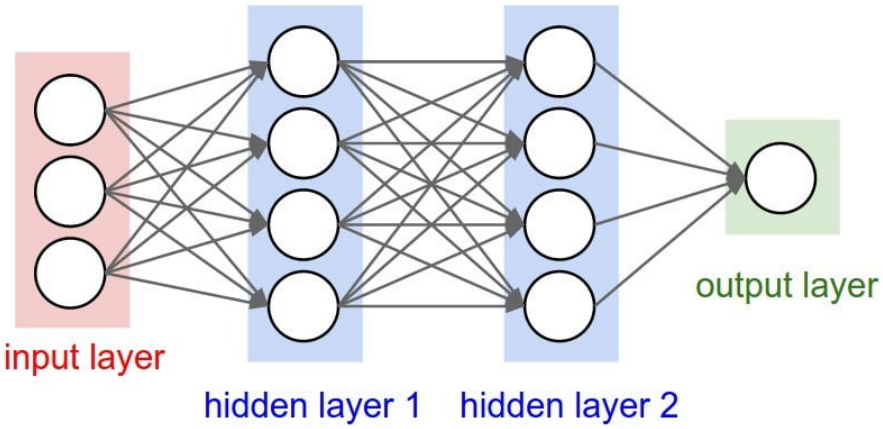


Figure 2: A schematic representation of an artificial neural network. The round circles are neurons, and the grey arrows represent synapses. In this specific schema the ANN consists of an input layer, two hidden layers, and an output layer (Karpathy, 2016).

ANNs are mostly computed on traditional Von Neumann architectures, that do not enjoy the benefits of coupled memory and computation. There is interest in using memristors within ANNs – as investigated by Thomas (2013), and Adhikari et al. (2012) – since combining multiple memristors into a crossbar array allows one to program ANNs directly on the memristors. The individual memristors can be used as weights in the network as introduced in e.g., Hu et al. (2012). Physical colocation of memory and computing can lead to an efficient ANN, which may be run natively on the computational architecture, instead of a Von Neumann architecture, where the ANN is effectively simulated.

Memristors have been used in different types of networks illustrate this benefit. Using memristors as connections in a long-short term memory (LSTM) network shows that the memristor is a promising low-power and low-latency hardware platform (C. Li et al., 2019). They are able to carry out the fundamental mathematical operations underpinning an ANN in an efficient manner, which is the ability to carry out parallel vector-multiplication. This follows from Ohm's law $I = VG$ and Kirchoff's current law $I_j = \sum I_i$. The input is voltage V, the output is current I, and the weight corresponds to the conductance G. By using the memristors in a crossbar array the vectors containing the properties of the individual memristors, which make up the connection weights, can be directly calculated in parallel using Ohm's law. Other applications of the memristor as connections in an ANN have been in edge detection, a notoriously resource-intensive algorithm (Prodromakis & Toumazou, 2011).

Since the resistance of the ideal memristor does not change when no current is passed through the material, the state of the memristor is 'saved' even when it is not activated. Due to no energy being needed to maintain its state, the memristor is energy efficient. In a practical setting, memristors have already been used in working random-access computer memory (ReRAM; Moon et al. 2019). A binary CNN accelerator on a digital ReRAM-crossbar has shown to be "296 times more energy-efficient than a high-end GPU" (Ni et al., 2017).

Over the years different types of memristors have been developed. This paper will use a simulation of Nb-doped $SrTiO_3$ memristors, which have been used to create a model that is a general function approximator (Tiotto & Goossens, 2020). The memristance properties of these memristors follow the power law, exhibiting exponential behavior. In addition, they show low reading currents at room temperature, which makes them ideal as neuromorphic hardware in everyday use. The nonlinear conductance dynamics ensure a large range of resistances (Brivio et al., 2018). The minimum and maximum resistance values are approached in a gradual manner due to soft-bounded behavior, guaranteeing slower learning and, symmetrically, longer memory retention. These effects may lead to a more robust learning performance (Frascaroli et al., 2018).

1.4 Working memory

In this research, the aforementioned Nb-doped $SrTiO_3$ memristors are used to simulate human working memory (WM). WM is a capacity-limited memory system that can hold information temporarily. WM plays an essential role in fluid intelligence: the ability to reason and solve problems in unique and novel situations. It is important for reasoning and the guidance of decision-making and behaviour (Diamond, 2013). Simulating WM by memristors would lead the way to a cognitive system capable of responding to a plethora of novel situations – just like our brain – while having a very efficient, low-energy underlying architecture.

It was always thought that when storing information in WM there had to be persistent firing of neurons, as found in Fuster & Alexander (1971). Interestingly, recent studies have shown that information can be stored in activity-silent states, i.e., without neuronal activity. This is of particular relevance since the combination of memristors and activity-silent states would present an even more efficient, low-energy, way of storing information. Mongillo et al. (2008) propose a model that exemplifies activity-silent states by a mechanism called short-term synaptic plasticity (STSP; Stevens & Wang, 1995). This model is engineered by Pals et al. (2020) is able to approximate human behavioural and neuronal data. In the STSP mechanism synapses can be temporarily strengthened (facilitated), due to a build-up of ions in calcium-mediated presynaptic channels. The residual calcium effectively leaves a 'synaptic trace' of the information in WM. When probed, the strengthened connections will facilitate the firing of certain neurons, leading to the stored information being expressed again. This presents a credible alternative mechanism to that of persistent firing.

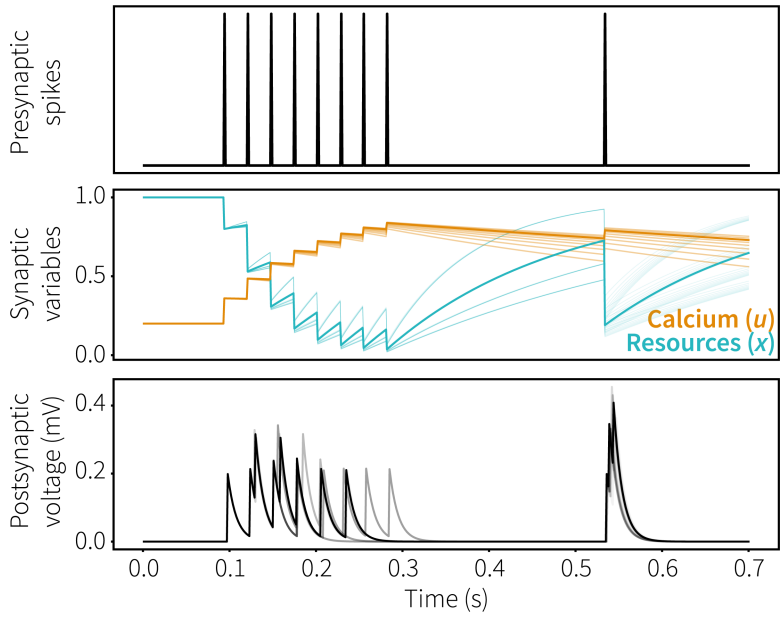


Figure 3: Short-term synaptic plasticity mechanism (STSP). Reprinted with permission from Pals et al. (2020).

The mechanism is further illustrated in Figure 3. The top panel shows spikes in the presynaptic neuron. In the middle panel, it is shown how the spikes lead to an increase of calcium (u) and a decrease of the resources (x) available to the presynaptic neuron. In the bottom panel, the resulting induced postsynaptic voltage is shown. After the first spike train, the resources are depleted. However, after the later spike, the postsynaptic voltage is much steeper than during the first spike train. This indicates that the presynaptic neuron has been facilitated due to the calcium build-up in the presynaptic channel.

1.5 Nengo

The model by Pals et al. (2020) is built in the Nengo framework (Bekolay et al., 2014). Nengo is a Python library designed to simulate large-scale neural models, using the Neural Engineering Framework (NEF) (Eliasmith, 2013). It has been used to create biologically plausible neural networks using unsupervised and supervised learning (e.g. Bekolay et al. (2013)). A notable project using Nengo is *Spaun*, a 2.5 million neuron model of the brain that is able to capture and demonstrate complex behavior on eight diverse tasks (Eliasmith et al., 2012).

NEF is in essence a neural compiler. If the properties of the neurons, the values to be represented, and the functions to be computed are specified, the compiler solves for the connection weights between components that will perform the desired functions. Typically these components are groups of neurons, called *ensembles*. An example of the inner workings of NEF when input is given to an ensemble is shown in Figure 4. Neurons in these *ensembles* are sensitive to a particular input, as shown in the left figure, where a specific input value x results in a higher firing rate (spikes/s). The middle figure shows the neural *spikes* in response to an input signal. The summation of these spikes over time constitutes the *decoded* output, as seen in the rightmost figure. In a simple example, the summation of spikes can be used to perform the identity function $f : x \rightarrow x$.

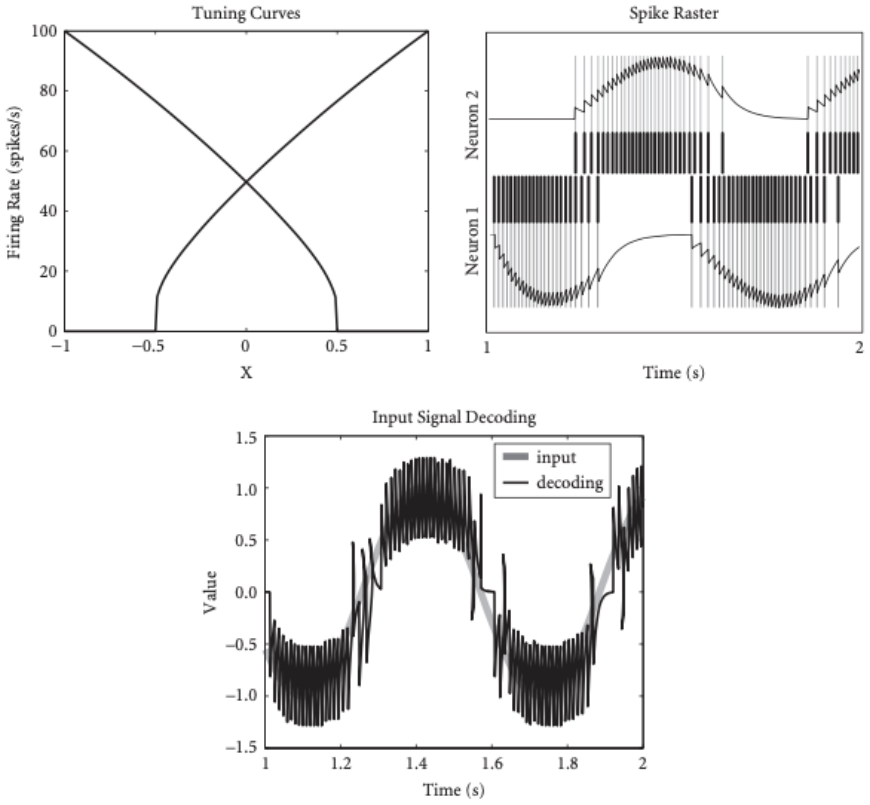


Figure 4: Inner workings of NEF illustrated by a response of an ensemble to an input signal. Leftmost figure: firing rate response of individual neurons to an input signal (tuning curve). Right figure: spikes in response to an input signal. Bottom figure: input signal and the decoded output of the ensemble (Eliasmith (2013), p.45)

1.6 Current study

The model engineered by Pals et al. (2020) provides an opportunity to use the resistance properties of the memristor in a flexible memory system. This entails simulating the memristance properties, modifying them via a learning rule, and using the conductance of the memristors as network parameters. In similar research, memristors have been utilized in a model employing spike-timing-dependent plasticity (STDP) (Serrano-Gotarredona et al., 2013). Here the strength of the synapse is dictated by the precise timing of individual pre-and/or post-synaptic spikes. Serrano-Gotarredona et al. (2013) show that there is no need for global synchronization, and learning happens asynchronously and on-line. This paves the way to other neuronal learning mechanisms based on memristors.

To summarize, in this research, a simulation of Nb-doped $SrTiO_3$ will be used in the architecture of a model by Pals et al. (2020) which has shown that information is maintained in activity-silent states. This model, created in Nengo, is used to perform a cognitive task, known as the delayed-response task. If the results were satisfactory this may lead to developing a flexible memory system based on memristors which could be an important step towards realising a true cognitive computer.

2 Methods

The model by Pals et al. (2020) shows that the STSP mechanism as proposed by Mongillo et al. (2008) results in effective, and functional human behavior, where both the behaviour as well as its neural representations are in agreement with human data. They integrated the STSP mechanism into a large-scale spiking-neuron model that can perform a delayed-response task.

In the following sections the experiment and model by Pals et al. (2020) will be explained in more details. Subsequent sections will focus on the learning rules in their experiment, as well as in this research. The last section of the Methods will delve into the current experiments.

2.1 Delayed-response task

As mentioned above, the model by Pals et al. (2020) is used to perform a delayed-response task. A delayed-response task is a task widely used for studying the maintenance of information in WM, where a briefly presented memory item is followed by a delay period, after which the items are queried. In this specific delayed-response task, as shown in Figure 5, two randomly oriented gratings are presented during each trial. After a fixation period, a cue is shown which indicates which of the gratings must be kept in memory. Then, an impulse is presented which acts as a high-contrast stimulus. This stimulus activates all synapses but due to its non-characteristic nature does not alter the memory representation. This helps to maintain the stimulus for a longer time, and the ensuing neural activity following the stimulus reveals what is currently held in an activity-silent state (Wolff et al., 2017). Lastly, a probe is presented which shows another visual grating. The participants are asked to indicate if the grating had moved clockwise or counter-clockwise.

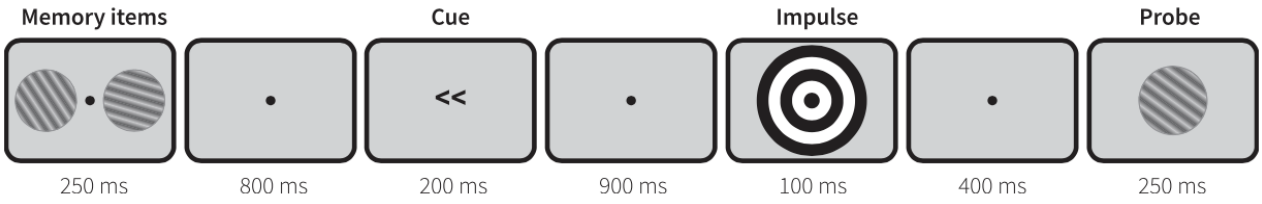


Figure 5: The retro-cue delayed-response task. Reprinted with permission from Pals et al. (2020).

Wolff et al. (2017) conducted experiments where human participants were asked to perform the delayed-response task. EEG data recorded during the task shows that indeed information was maintained in activity-silent states. Hereby the orientation of the grating was effectively kept in memory. The spiking-neuron model by Pals et al. (2020) performed the same task. Comparing the representation in the activity-silent states to the representation maintained by the spiking-neuron model

showed remarkable similarities.

2.2 Model architecture

In this research the same model architecture was used as in the model created by Pals et al. (2020). The following section focuses on its inner workings.

The spiking-neuron model consists of several groups of neurons (ensembles), as shown in Figure 6. The input is passed to the sensory ensemble, which in turn has a connection to the memory and comparison ensemble. The memory ensemble is critical for storing the presented input via a recurrent connection. This all-to-all connection between neurons is where the connection weights will be adjusted via a learning rule. To increase the biological plausibility some noise (X) is randomly drawn from a Gaussian distribution, $X \sim \mathcal{N}(\mu, \sigma^2)$, and inserted into the memory ensemble. Here μ is 0 and σ is 0.010. The comparison ensemble compares the output of the sensory ensemble, where the presented input has been encoded, with the output of the memory ensemble, which produces a neural representation of the orientations of the probe. Finally, the decision ensemble processes the different orientations of both the probe and the memory item, and decides if the memory item has been rotated clockwise, or counter-clockwise. The model parameters are attached in the Appendix (Table 22).

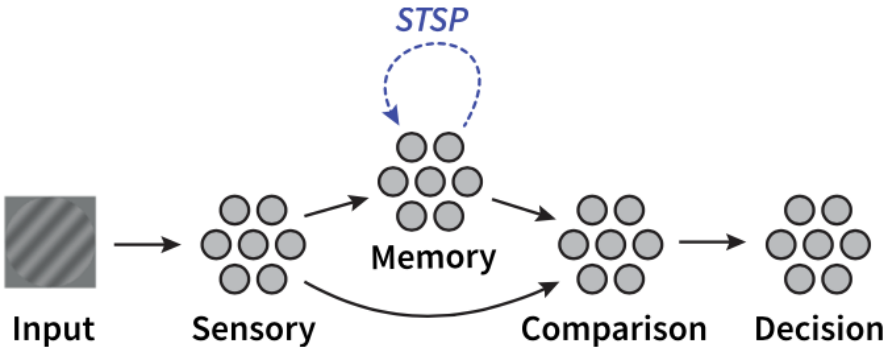


Figure 6: Model architecture. Reprinted with permission from Pals et al. (2020).

2.3 Learning rule

In the model by Pals et al. (2020), learning happens in the memory ensemble, using an all-to-all recurrent connection. The neurons in this ensemble are adapted leaky fire-and-integrate neurons, incorporating the STSP mechanism. Each neuron has resources (x) and calcium (u) and is updated

each timestep by the following equations:

$$x = x + (dt * \frac{(1-x)}{\tau_x} - u * x * \text{output}) \tag{1}$$

$$u = u + (dt * \frac{U-u}{\tau_u} + U * (1-u) * \text{output}) \tag{2}$$

Here τ_x and τ_u are the relaxation time of respectively the resources and the calcium. U is the starting value of the calcium (0.2). If the neuron has spiked; the *output* is 1000, if not, it is 0.

The connection weights are updated based on the calcium and resources of the model, by the following equation:

$$\Delta\omega_i = \frac{u_i x_i}{U_i} * \omega_{0,i} - \omega_i \tag{3}$$

Here $\omega_{0,i}$ is the initial weight, and ω_i is the current weight of connection i.

The recurrent connection uses decoded values as input to the ensemble. In a practical setting: if the ensemble were to represent the identity function ($f : x \to x$), and the input would be a constant value of 0.5, the recurrent connection would solve for the most fitting connection weights to reproduce the value 0.5 at each time step.

As mentioned in Section 1.3, memristors have successfully been used as the connection weights in ANNs. Consequently, we chose to implement a learning rule that directly alters the connection weights. The physical properties of the memristor used in this research make it difficult to incorporate learning rules that result in volatile connection weights, as will be explained more thoroughly later on. As a result of this we settled for a straightforward learning rule based on Hebbian learning, which was first introduced by Bienenstock et al. (1982).

2.4 BCM

The BCM update rule is a form of Hebbian learning, as the update depends on the pre- and post-synaptic activity:

$$\Delta\omega_{ij} = \alpha_i \alpha_j (\alpha_j - \Theta) * \mu \tag{4}$$

Here Θ is the modification threshold, which in this case is the average synaptic activity. If the post-synaptic activity, α_j, is greater than Θ, the synaptic weight ω_{ij} will be potentiated, in a process called *long-term potentiation* (LTP). Conversely, if α_j is smaller than Θ, the synapse will undergo *long-term depression* (LTD). The modification threshold reflects the expectation of a cell's activity. The idea is that cells that are driven above their average activity $E(\alpha_j)$ must play an important role so their afferent synapses become potentiated (Bekolay et al., 2013). In Nengo the modification

threshold is enabled by a low-pass filter, and Equation 1 is multiplied by the learning rate μ. A low-pass filter is used to filter-out high-frequency components, this can be seen as a type of averaging.

2.5 Memristor-based BCM

In order to use the BCM learning rule in conjunction with Nb-doped $SrTiO_3$ memristors, the BCM rule was adapted into a discretised version called mBCM, which operates under the constraints of the memristive material. The resistance of the memristors changes due to applied voltage. To simulate the memristors, the behavior is modelled in response to SET pulses (of +0.1V) (Tiotto & Goossens, 2020). The behavior can be described by an exponential equation:

$$R(n,V) = R_0 + R_1 n^{a+bV} \tag{5}$$

Here V represents the amplitude of the SET pulse, n the pulse number, R_0 the lowest value that the resistance could reach, and $R_0 + R_1$ the highest value. Given that only SET pulses of +0.1V are applied, we can calculate $V = \frac{1}{a+b^V}$. The estimated best fit for the memristor behavior is found by:

$$R(n) = 200 + 2.3 * 10^8 n^{-0.146} \tag{6}$$

Instead of directly altering the connection weights by Equation 4, the sign of the calculated connection weight change ($\Delta\omega_{ij}$) is used. If $\Delta\omega_{ij}$ is positive, a SET pulse of 0.1V is given to the simulated memristor. Tiotto et al. (2020) used a pair of memristors in their experiment. One of the memristor was initialized with a low resistance, and the other with a high resistance. Taking the difference in conductance between the two memristors, and pulsing them separately, allowed for positive and negative increases of the connection weights. Considering we only want to learn a pattern, and not unlearn it, we only have to increase the connection weights between neurons that are active due to the input pattern, and a single memristor suffices. The resulting weights are calculated by transforming the resistance values into conductances, multiplicated by the hyperparameter gain (γ). γ functions in the same way as the learning rate μ in Equation 4.

2.6 Current experiments

The goal of this research is to substitute the STSP learning mechanism in the model by Pals et al. (2020), from here on referred to as Pals model, with a memristor-based learning rule. In Pals model the neurons in the memory ensemble are adapted to contain a calcium and a resource property. The learning rule utilizes the levels of calcium and resources to update the connection weights. The recurrent connection in the memory ensemble consists of an all-to-all decoded neuron connection.

To create a model utilizing a memristor-based learning rule multiple experiments are devised. The first part of the experiments focuses on exploring the learning capabilities of BCM and mBCM

and its limitations. Subsequent experiments use an adapted STSP mechanism based on memristors. Lastly, a comparison is made between the results of STSP and the memristor-based STSP mechanism.

In order to compare the results of STSP and the memristor-based STSP mechanism, identical experiments are conducted as by Pals et al. (2020), from here on further referred to as Pals experiment. First the neural representations of the model is examined by simulating one trial of the delayed-response task. The spiking activity and contents of calcium and resources are recorded and visualized. Pals model shows that information is maintained in activity-silent states. To examine which orientations share the most similarity with the vector in the memory ensemble, the absolute normalized cosine similarity between the memory items and probe is calculated, averaged over 100 trials. During each trial the orientation of the memory item and probe are respectively 0°and 42°. Crucially, this is not the case in the full experiment, where the orientations of the memory item and probe are uniformly drawn from seven different angles (3°, 7°, 12°, 18°, 25°, 33°, 42°), both clockwise and counter-clockwise. The experiment shows that the neural connections representing the memory item are indeed facilitated. The full experiment consists of 30 sets of 1,344 trials, that match the 30 participants in the human experiment conducted by Wolff et al. (2017).

As mentioned above, the aforementioned experiments are repeated with the memristor-based STSP learning mechanism[1]. In this mechanism the resulting calcium is dependent on SET pulses to the memristors, and the gain (i.e. learning rate). The experimentally derived gain that shows the most resemblance to the original STSP mechanism is reported, and applied across the experiments.

Since the Pals model has shown to resemble the human data from Wolff et al. (2017), we compare the performance of the memristor-based STSP model with the STSP model directly. If both models show comparable behaviour, we can surmise that the memristor-based STSP model displays plausible functional human behaviour.

[1]Code for experiments provided at `https://github.com/JoppeBoekestijn/WM-memristors`

3 Results

As mentioned in Section 1.3, the literature has focused on ANNs where the connection weights are directly and successfully realised on memristors. For that reason the first goal of this research was to create a learning rule from which the connection weights would be realised on memristors. In the following sections the learning capabilities, which are essential for encoding the neural representation of the input grating, of BCM and the memristor-based mBCM will be investigated. Both the learning rules have shortcomings, ultimately resulting in a novel learning rule (mSTSP). This learning rule will be applied to the experiments conducted by Pals et al. (2020). The result of the experiments are compared to the results of Pals et al. (2020).

3.1 BCM learning

In the delayed-response task a memory item needs to be kept into memory. Thus the learning rule that we apply also needs to be able to learn, and reproduce a pattern. First, we use BCM in a basic spiking-neuron model. Instead of a recurrent connection, which is used in Pals model, we use a direct connection between two ensembles of 15 neurons, *pre*, and *post*, as depicted in Figure 7. A direct connection means that the neural activation is used in the connection, instead of the decoded values. The learning rate is set to $\mu = 5e-11$ (Equation 4). Input to the *pre* ensemble is a sine wave for 30 seconds. Input to the *post* ensemble is also a sine wave, but only for the first 15 seconds of the simulation, afterwards the input is disabled, as can be seen in the top panel of Figure 8. Both ensembles try to mimic their input patterns, essentially applying the identity function $f : x \rightarrow x$. The direct connection between the *pre* and *post* ensembles use the BCM learning rule. As the middle panel of Figure 8 shows, the *post* ensemble keeps outputting the sine wave, even after the input to that specific ensemble is disabled.

BCM potentiates active synapses and depresses non-active synapses. During the learning phases, the synapses that are being activated due to the input to the ensemble are therefore potentiated by increasing the connection weights between active neurons, in this case the connection weights between the neurons from the *pre* and *post* ensemble. If the original pattern is sufficiently learned, the ensemble will start to represent that pattern, even when the original pattern is no longer given as input to the model. The middle panel shows that after 15 seconds the post ensemble can recreate the sine wave, showing that BCM can be successfully applied to learn a pattern, in a spiking-neuron model with a direct neuron connection.

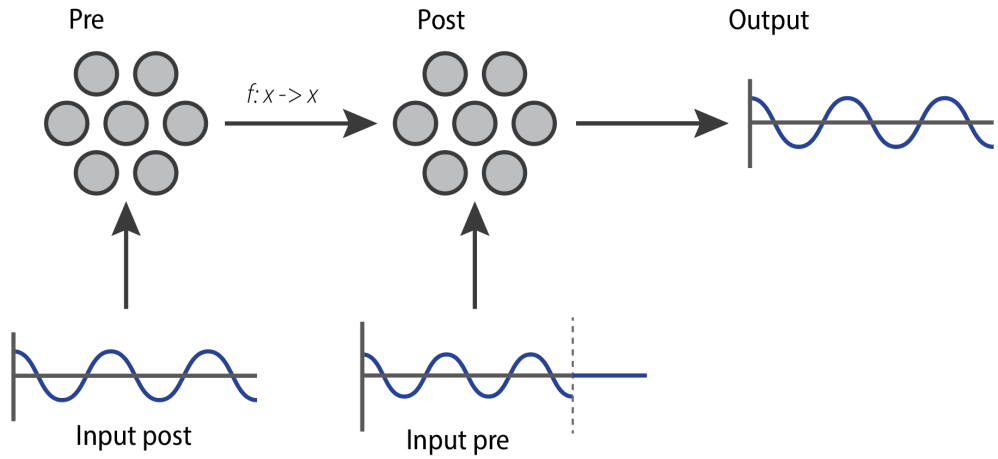


Figure 7: Model architecture. *Pre* and *post* ensembles both have a unique input, respectively: *input pre*, and *input post*. Both inputs are a sine wave, after $t = 15$ seconds *input post* becomes 0.0. The *post* ensembles learns to represent the sine wave, even when *input post* is disabled.

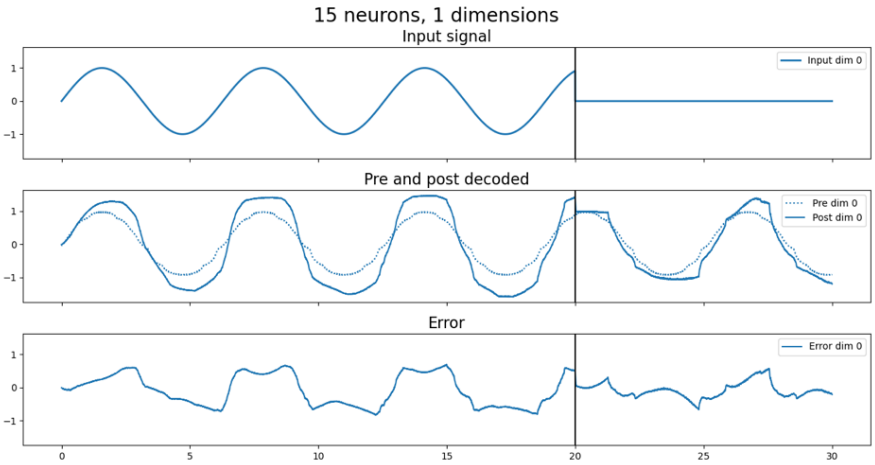


Figure 8: Nengo model learning to represent the sine wave. Top panel: input signal to the pre ensemble. Middle panel: decoded values from the pre and post ensemble. Bottom panel: error between the decoded output of the post ensemble and the sine wave.

3.2 Reactivation with recurrent connection

Instead of a direct connection, we explore the results of BCM using a recurrent connection, as was used in Pals model. In Pals experiment a cue is shown during the delayed-response task. This cue functions as a neuron-wide unspecific input that activates neurons, but crucially, does not alter the information kept into memory. Encoding the contents of neural activation during the cue then shows

the information that is kept into memory. This then shows if the correct memory item has been effectively learned. In order to investigate if information is successfully kept into memory using BCM, and can be triggered by a neuron-wide unspecific input, another experiments is devised, where the neuron-wide unspecific input is simulated by injected a small amount of current into all neurons, and thereby "reactivating" the neurons.

The idea of reactivation works as following. If sufficient current is injected into an ensemble some of its neurons will fire. Ideally, these neurons have already been active during the presentation of the memory item. Due to the increased connection weights between the neurons that make up the original pattern, the stimulation of some of these neurons results in the same spiking pattern appearing, which can be interpreted to find out what is kept in memory. However, injecting too much current might result in the unnecessary firing of neurons, and simultaneously might increase the connection weights between neurons that do not represent the presented memory pattern.

A non-specific population-wide current of 0.4 ensures that some neurons fire, and due to the increased connection weights, the original learned pattern also emerges, as shown in Figure 9. However, the original input of 0.5 does not manifest, which is represented by the grey line. Later sections will delve into the causes of this behavior.

The level of injected current is dependent on the number of neurons in the ensemble. As the number of n neurons grow, the number of connection grows with $n * n$. Due to the multitude of connections, a smaller current injection suffices to activate the original spiking pattern. Due to the higher complexity, the injected current in Pals model is much lower (0.02).

3.3 Persistent firing

One of the reasons why the input of 0.5 does not reemerge after reactivation, is because the connection weights have not been strengthened. However, with a recurrent connection, if the connection weights become too large, the neurons will just keep firing, which enforces the increase of the connection weights, thus leading to more firing. This behavior of persistent firing directly counters the idea of activity-silent memory, and hampers the ability to reproduce an input pattern. To avoid this the connection weights must not become too large during the presentation of memory items. A few ways to address this behavior are shown below, although they do not result in the sought-after learning behaviour.

1. BCM with max weights
2. Stop learning
3. Adaptive integrate-and-fire neurons (AdapLIF)

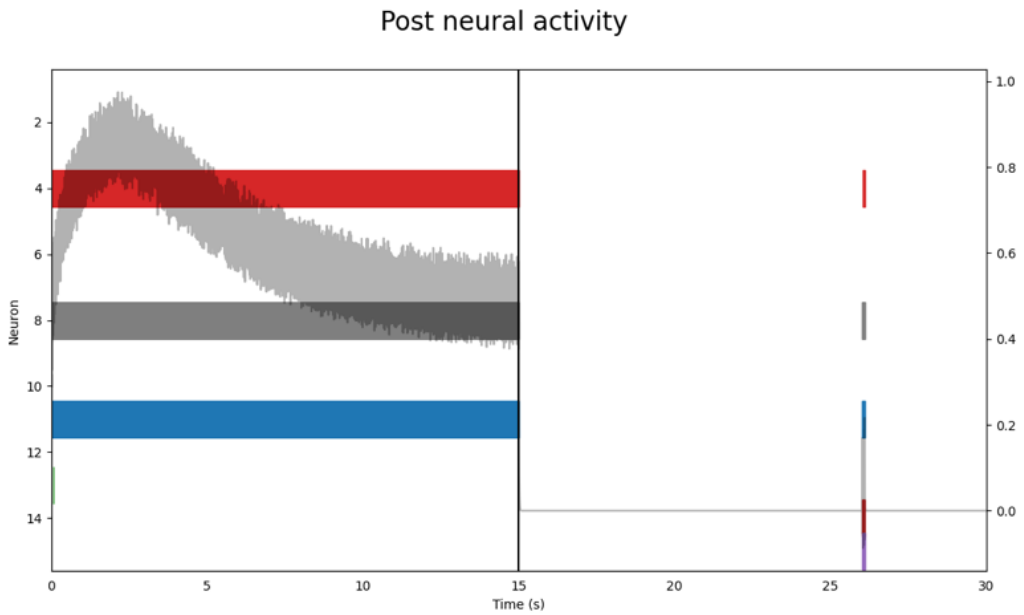


Figure 9: Spiking of the neurons in the *post* ensemble (colored). Decoded value from *post* ensemble (gray). Current injection of 0.4 at t=26.0

3.3.1 BCM with max weights

First, we tried to limit the connection weights to counter the persistent firing of the neurons. We adjusted BCM such that the connection weights could not be increased further than a predefined hyperparamaters λ. This makes the learning rate essentially obsolete: if the learning rate is large enough that the connection weights reach λ, increasing the learning rate further will only result in the connection weights becoming saturated sooner. The benefit of this approach is two-fold; firstly the connection weights can be defined beforehand, instead of having to test different learning rates to end up at the desired connection weights. Secondly, the spiking during the reactivation phase does not increase the connection weights further, making it less probable that the neurons start to persistently fire due to the higher connection weights.

3.3.2 Stop learning

Another way to limit the connection weights is to stop the learning after t seconds. Since we only want to learn the pattern during the memory item presentation phase, we can adjust the simulation such that it only increments the weights of the connections during that phase. In Nengo this is done by splitting the simulation into two parts, in which the first part encompasses the memory item presentation phase ($t = 0$ until $t = 0.25$), and the second part the remainder of the simulation ($t > 0.25$ until $t = 3$). In the second part, the learning rate is manually adjusted to 0, limiting the

connection weight update. This still means that the learning rate has to be carefully chosen such that the connection weights do not become too large during the memory item presentation phase, but the reactivation phase cannot result in persistent firing of the neurons. Although this approach stops the learning, and therefore the increase of the connection weights, still a too low learning rate will not reproduce the correct pattern, and a too high learning rate results in persistent firing.

3.3.3 AdapLIF

Adaptive integrate-and-fire neurons (AdapLIF) provide a different mechanism to limit the firing of neurons after being active for a longer period of time. By default ensembles in Nengo consist of leaky integrate-and-fire neurons (LIF). AdapLIF neurons extend the LIF neuron with an adaptation parameter. After prolonged constant firing, the adaptation state will increase, and in turn decrease the current injection to the neurons. In Nengo this is realised by decreasing the current injection in the neurons by:

$$\tau_n \frac{dn}{dt} = -n \tag{7}$$

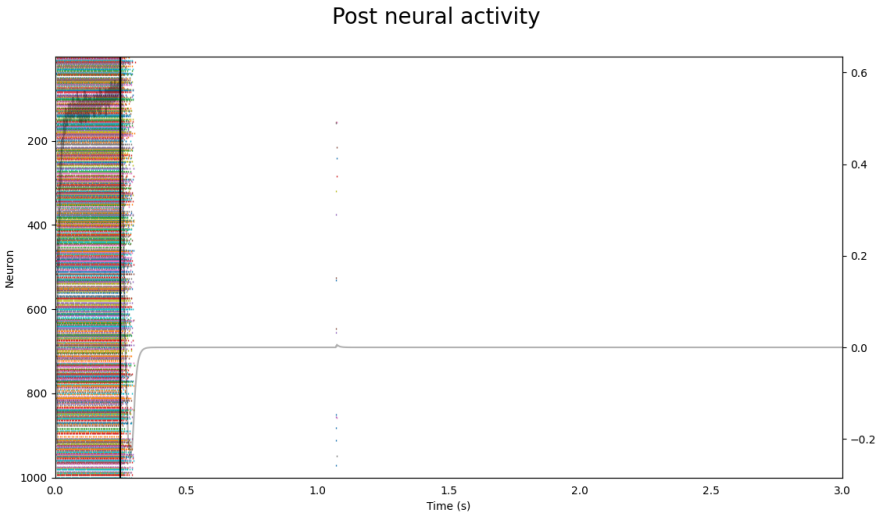
Here τ_n is the adaptation time constant and affects how quickly the adaptation state decays to zero in absence of spikes. When a neuron spikes, the adaptation state n is increased by inc_n. This mechanism might allow for a higher learning rate, without the ensuing persistent firing.

An experiment is conducted that matches Pals experiment in the number of neurons (1000) and uses an ensemble with a recurrent neuron all-to-all connection. The BCM learning rule is applied to illustrate the effect of AdapLIF. In the following steps the most fitting hyperparameters are selected:

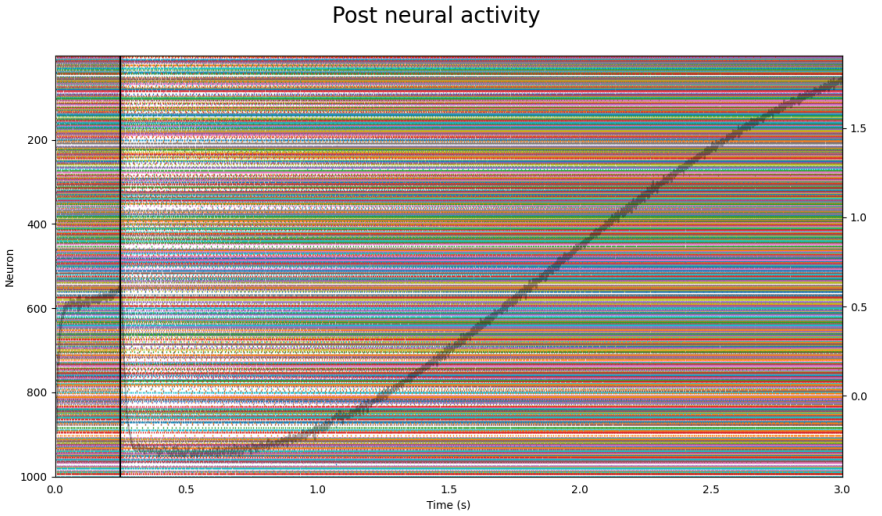
1. Find lowest learning rate that has persistent activation of neurons
2. Replace LIF by AdapLIF neurons
3. Iteratively increase the learning rate
4. Repeat step 3 until the highest learning rate where there is no persistent firing is found

It is crucial to increase the learning rate as much as possible, to ensure that the input pattern is sufficiently reinforced in the connection weights of the model. Preferably, the learned pattern reemerges during the reactivation phase. During the learning phase, the input node consistently injects the value 0.5 into an ensemble. In step (1) we find that the lowest learning rate that demonstrates persistent firing is $7e-13$ as illustrated in Figure 10b.

Substituting the LIF neurons for AdapLIF neurons allows for a higher learning rate. Using steps (3) and (4) we find the highest learning rate of BCM using AdapLIF is $1e-12$, the results are enclosed in the Appendix (20). Using AdapLIF neurons warrants a higher learning rate. Nonetheless,



(a) BCM with a learning rate of $6e-13$



(b) BCM with $7e-13$ learning rate

Figure 10: Neural activity (grey) and spikes per neuron (colored) during simulation.

although there is more activation during the reactivation phase, we do not see that the original input (0.5) emerge. Figure 21 in the Appendix shows that different values for the hyperparameters tau_n and inc_n deteriorate the performance as opposed to the default values ($tau_n = 0.1$ and $inc_n = 0.01$).

3.4 mBCM

A combination of the aforementioned techniques is used in conjunction with the mBCM learning rule. The same experiment as mentioned above is conducted with mBCM, with early stopping of learning after $t = 1.0$ seconds. Through steps 1 to 4, the appropriate hyperparameters for AdapLIF and the learning rate are discovered. Figure 11 shows the most fitting results with and without AdapLIF neurons. A subset of neurons does fire during the reactivation phase, but the input pattern (0.5) is not replicated.

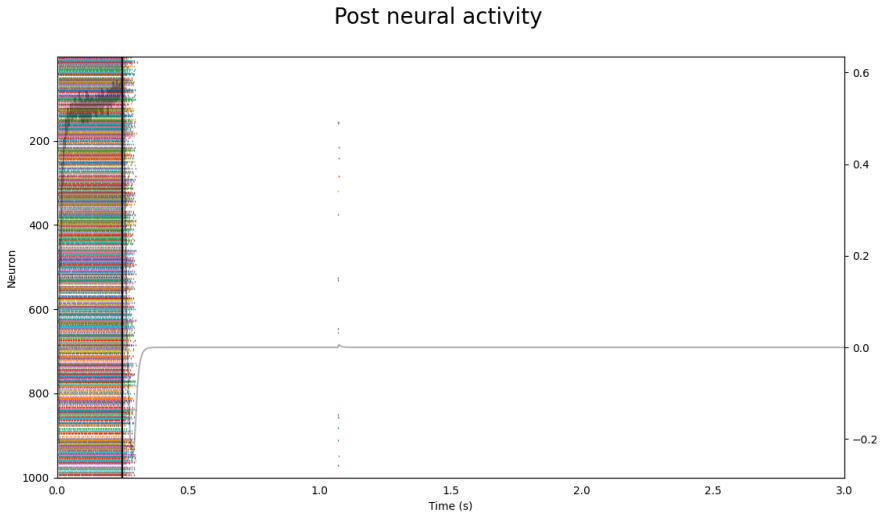
3.5 STSP

It seems that persistent firing has a detrimental effect on the ability of the ensemble to learn a pattern. All of the aforementioned techniques - AdapLIF, BCM with max weights, and early stopping - do not counter the problem of persistent firing with the BCM or mBCM learning rules. The same experiment as above is repeated with the STSP mechanism that was used in Pals model. Interestingly, as Figure 12 shows, this does result in the correct learned pattern to reemerge. The neurons that fire during the learning phase also become active during the reactivation phase. More interestingly, the grey line, which indicates the decoded output of the ensemble, becomes around 0.5 during the reactivation. This means that the neurons that are active in order to represent the input pattern have been successfully reinforced.

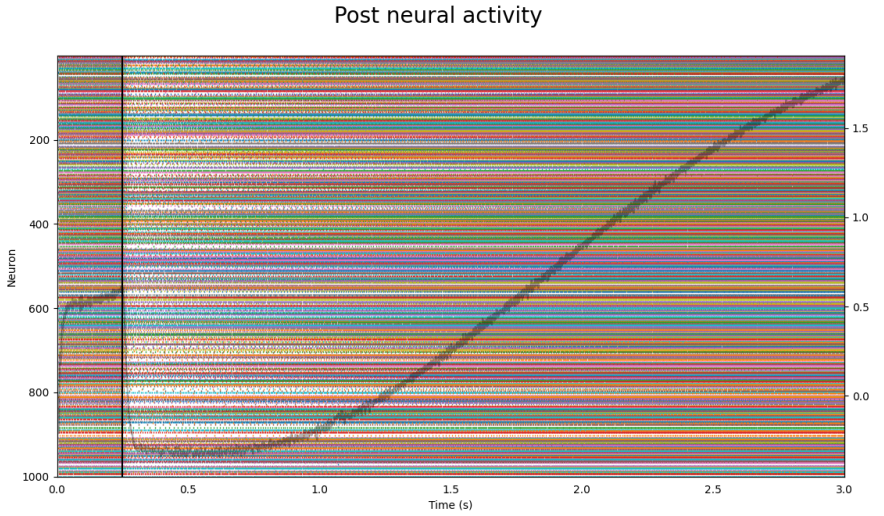
The learning rule impacts the connection weights. STSP and BCM exhibit different behaviour in updating the connection weights throughout the simulation. In Figure 13 the connection weight of an active neuron to itself is shown. The right panel shows that with BCM the connection weight keeps steadily increasing. This may lead to either persistent firing or relatively low connection weights. With STSP, as can be seen in the left panel, the connection weight is dependent on the calcium and resources of the neuron. Because the resources of an active neuron quickly deplete, the connection weight becomes almost zero when the neuron is active. When the neurons get reactivated at $t = 1.050$ seconds the relatively high connection weight of neurons that have been active result in their firing. A plausible explanation for the lack of persistent firing with STSP is that the resources of the neuron quickly become depleted, decreasing the connection weight, which can not induce persistent firing of the neurons.

3.6 Calcium mechanism with memristors

Seeing as the previous experiments showed issues with persistent firing, impeding the learning performance, while the STSP learning rule does produce satisfactory behaviour, we decided to adjust the STSP mechanism to incorporate memristors directly. Figure 14 shows the behavior of the cal-



(a) mBCM with a gain of 80 and AdapLIF neurons



(b) mBCM with a gain of 79 and AdapLIF neurons

Figure 11: Neural activity (grey) and spikes per neuron (colored) during simulation.

cium and resources of a single neuron in a working model using the STSP learning rule.

As will be explained in further sections, the current memristor model has its limitations. Due to these limitations, in combination with the fact that we only increase the resulting conductance of memristors by SET pulses, we only model the increasing part of the calcium mechanism on the memristors.

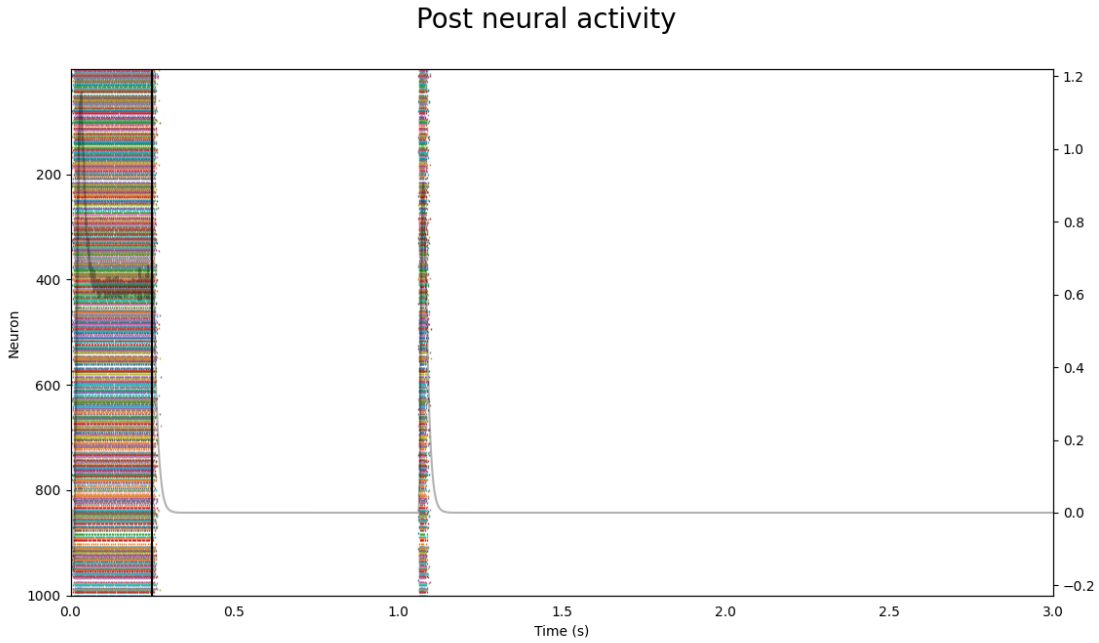


Figure 12: Neural activity (grey) and spikes per neuron (colored) during simulation using the STSP learning rule. During reactivation the input pattern reemerges.

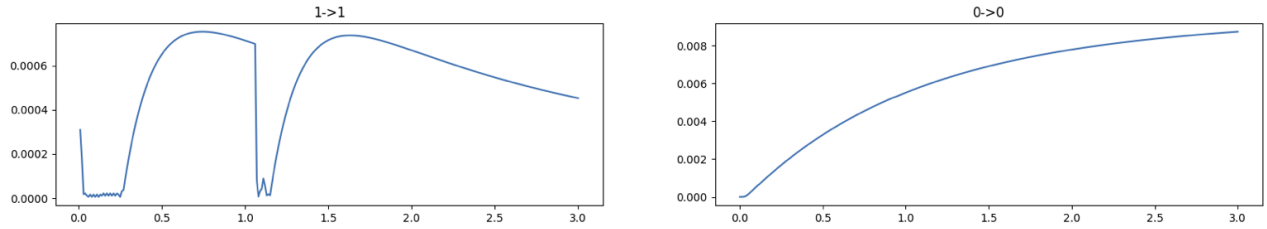
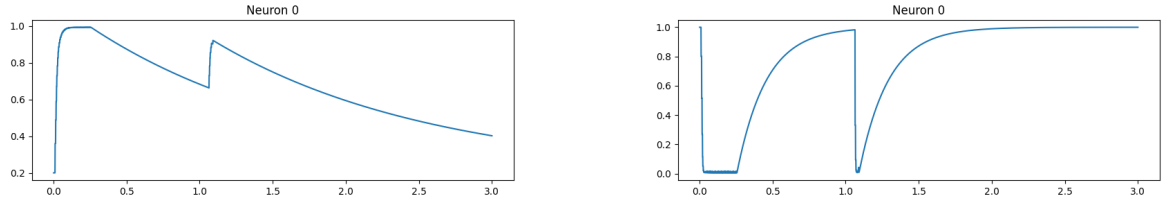


Figure 13: On the x-axis the time (t) in seconds. On the y-axis the connection weight. Connection weight during simulation of an active neuron to itself for left: STSP, right: BCM

The STSP learning rule with memristor-based calcium is further referred to as mSTSP. If the calcium update is positive, a SET pulse of $0.1V$ is given to the memristor. We do not account for negative updates of the calcium. Only the increasing part of Equation 2 is used, resulting in:

$$V(t) = \text{sign}(dt * (U * (1-u) * \text{output}) * 0.1 \tag{8}$$

The STSP connection weight update stays the same as in Equation 3. Likewise, the resources mechanism is not altered.



(a) Calcium (b) Resources

Figure 14: STSP. On the x-axis time (t) in seconds. On the y-axis the level of calcium and resources of neuron 0.

A SET pulse has a limited effect on the resistance of the Nb-doped $SrTiO_3$ memristors. This is ideal for representing small values, such as the connection weights in the previous experiments, but to incorporate the calcium mechanism the memristor must represent values in the range $[0.2, 1.0]$. Two ways to circumvent this problem is pulsing the memristors multiple times when the calcium increases (adaptive pulses), or adjusting the gain.

3.6.1 Adaptive pulses

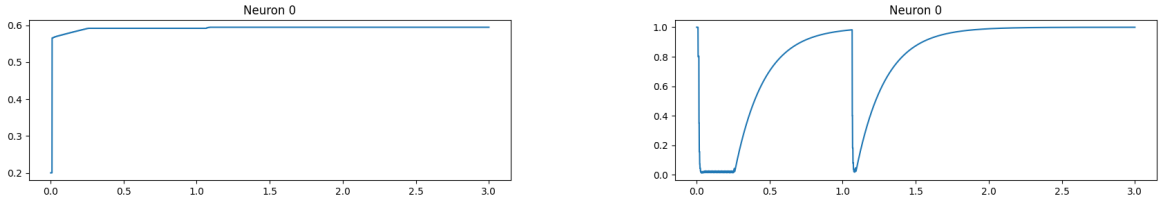
Instead of a single SET pulse, the memristor can be pulsed multiple times to increase the conductance. Different magnitudes of pulse levels are examined. The resulting maximum calcium of an active neuron is shown in Table 1. As the table shows the resulting calcium will only increase

Pulse levels	**Maximum calcium**
$10,000,000$	$1 * 10^{-5}$
$1,000,000$	$1.2 * 10^{-5}$
$100,000$	$8 * 10^{-6}$
$10,000$	$5 * 10^{-6}$
$1,000$	$3.5 * 10^{-6}$
100	$2.4 * 10^{-6}$
1	$1.2 * 10^{-6}$

Table 1: Number of SET pulses in combination with the maximum calcium of an active neuron

slightly even if a multitude of SET pulses are given to the memristor. The resistance of the memristor is initialized at $1 * 10^8$. Every SET pulse decreases the resistance until the lower bound of 200. Since we convert the resistance to conductance in the last step ($G = \frac{1}{R}$), the resulting output will

always be a small value, regardless of the resistance of the material.



(a) Calcium (b) Resources

Figure 15: mSTSP with $5 * 10^5$. On the x-axis time (t) in seconds. On the y-axis the level of calcium and resources of neuron 0.

3.6.2 Increasing gain

Alternatively, we increase the gain, essentially scaling the resulting conductance of the memristors. A SET pulse will still be given to the memristor if the calcium update is positive. The resistance is calculated according to Equation 6, which for illustrative purposes is also shown here.

$$R(n) = 200 + 2.3 * 10^8 n^{-0.146} \tag{9}$$

From the resistance follows the conductance ($\frac{1}{R} = G$). The resulting conductance is multiplied by the gain (γ), as was done with the mBCM learning rule. Figure 15 shows the calcium levels of a single active neuron when a gain of $5 * 10^5$ is used. After 1 second the calcium is around 0.6, which is the same as the calcium level of neuron 0 using the STSP learning rule (Figure 14). A gain of $8.3 * 10^5$ results in a calcium level of approximately 1 for an active neuron. This resembles the STSP calcium level when we only increase the calcium, and omit the decreasing term of the equation. Figures 23, 24, and 25 in the Appendix show respectively the STSP simulation, mSTSP with a gain of $5 * 10^5$, and mSTSP with a gain of $8.3 * 10^5$. The left panels of those figures show the mean calcium and resources over all neurons (1000). With the model using STSP, the calcium is increased during activity, and decreased during inactivity, while this is not the case with mSTSP. The spiking activity recorded in all three scenarios shows that the same neurons that fire during learning, are reactivated, thus the connection weights are correctly facilitated, see Table 2.

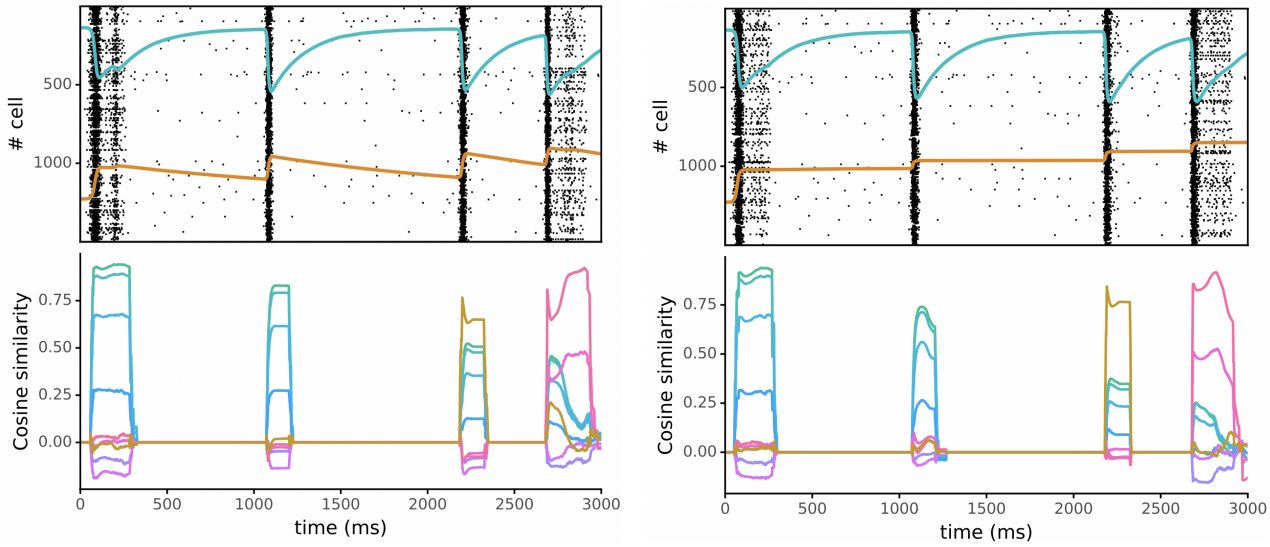
3.7 Neural activity

To examine the neural activity, the spiking activity and contents of the calcium and resources are measured during a single trial, as shown in the top panels of Figure 16a and Figure 16b. In this singular trial the memory item has an orientation of 0°. In order to gauge if the neural connections

Learning rule	**Learning** $(0-250ms)$	**Reactivation** $(1-1,070ms)$	**Combination**
STSP	497	504	497
mSTSP, gain: $5*10^5$	497	504	497
mSTSP, gain: $8.3*10^5$	497	504	497

Table 2: Number of unique neurons that spike at least once during learning (time period $0-250ms$) and reactivation (time period $1-1,070ms$). Last column shows the number of neurons that are both active during learning and during reactivation.

that are required for the learned orientation are indeed facilitated, the vector in the memory ensemble is compared with the ideal vector that would have produced the neural activity for the memory items with different orientations (bottom panels). For the model using mSTSP, a gain of $5*10^5$ is used, which produced the best matching results.



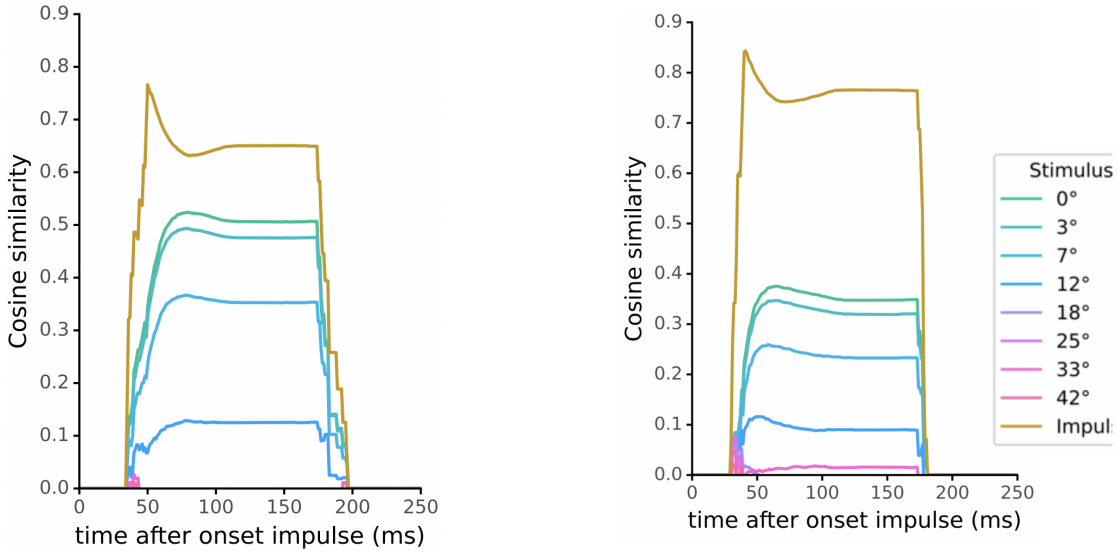
(a) STSP

(b) mSTSP with gain of $5*10^5$

Figure 16: Top: spiking activity during a single trial. Over time (ms) spiking (black dots). Mean calcium (orange), and resources (blue) over all neurons. Bottom: absolute normalized cosine similarity averaged over 100 trials. Orientation of memory item and probe are respectively 0°and 42°.

If we compare the spiking activity, in the top panels, we see similar patterns emerge. In contrast to the STSP model, we only pulse the memristors, solely allowing the calcium level to rise. This does not seem to influence the spiking patterns in any significant way. The marginal increase of the

memristors' conductances due to the limited effect of SET pulses on the conductance has the effect of approximating the calcium increase and decrease in the original STSP model. In both cases, the average calcium at the end of the trial is around 0.5.



(a) STSP

(b) mSTSP with gain of $5 * 10^5$

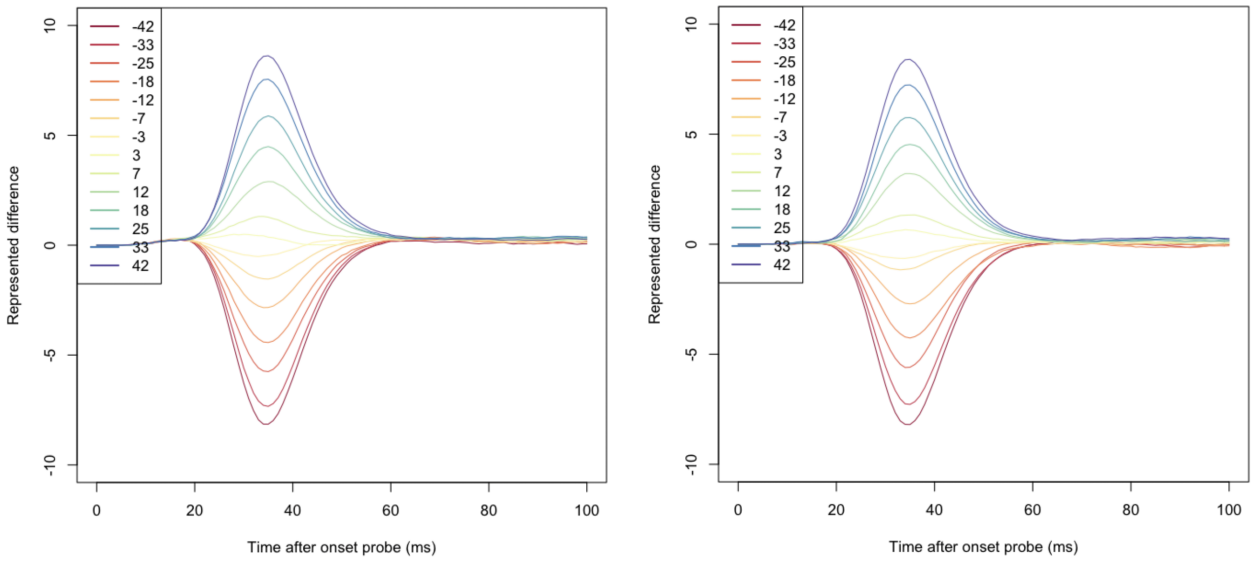
Figure 17: Cosine similarity of respectively STSP and mSTSP during impulse presentation.

The average absolute normalized cosine similarity during 100 trials also shows remarkably similar behaviour. Figure 17 focuses on the cosine similarity during the impulse presentation. As expected, the impulse is encoded into the model. However, the correct orientation, 0°, is dominant in both models, compared to the other orientations. This is the orientation of the memory item during this single trial, showing that indeed the correct orientation is stored in memory.

3.8 Performance

To ascertain if functional behavior arises, the memristor-based mSTSP model is applied to experiment 1 of Pals et al. (2020). 1,344 trials across 30 subjects are simulated in order to match the experiment conducted by Wolff et al. (2017).

The value in the decision ensemble represents the angular difference between the memory and probe orientation. Figure 18 shows the represented angular difference averaged across all trials. The left figure shows the STSP results, and the right figure the mSTSP results. An experimentally derived gain of $8.3 * 10^5$ has demonstrated the best-fitting results.



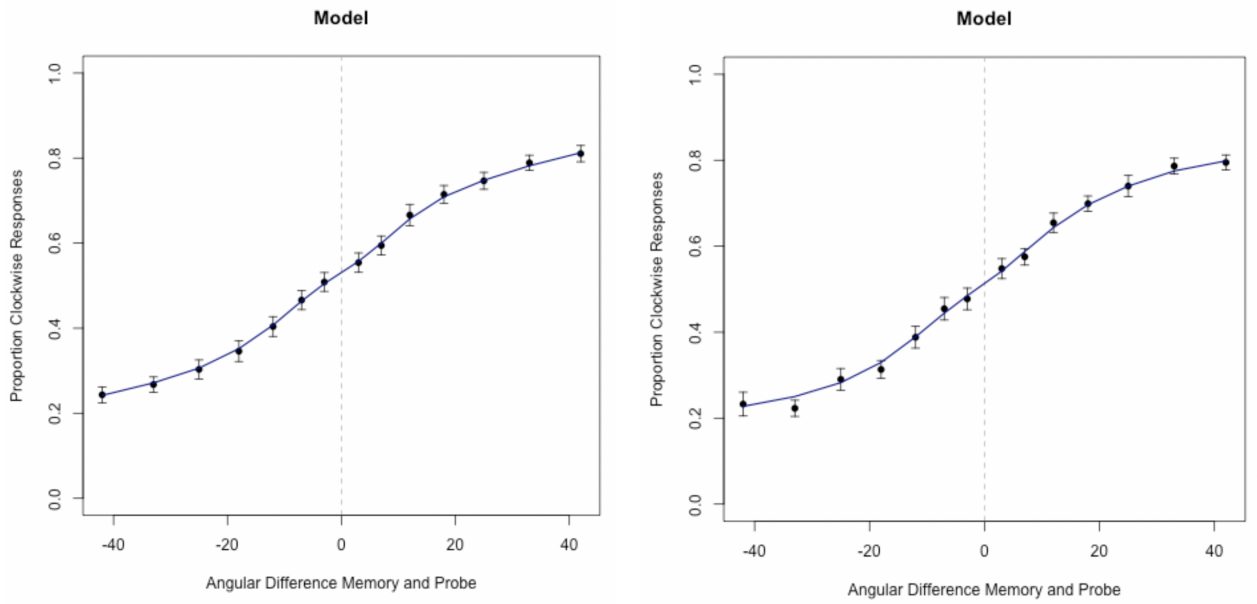
(a) STSP

(b) mSTSP with gain of $5 * 10^5$

Figure 18: Represented difference in the decision population.

The STSP model and the mSTSP model with a gain of $8.3 * 10^5$ have almost identical performance. Interestingly, a gain of $5 * 10^5$, which had the best match regarding the neural activity in the previous experiment, produces unsatisfactory results in the full experiment. Figure 26 in the Appendix shows that the mSTSP model with $5 * 10^5$ gain is inadequate in finding the differences between orientations of the memory item and probe.

To investigate whether the model correctly indicates that an memory item has been rotated clockwise or counter-clockwise relative to the probe, the proportion of clockwise response dependent on the angular difference of the memory item and probe is shown in Figure 19. An almost identical S-shaped curve is found with the mSTSP model as with the STSP model, meaning that both models are able to distinguish clockwise responses.



(a) STSP

(b) mSTSP with gain of $5 * 10^5$

Figure 19: Proportion of clockwise response dependent on the angular difference between the memory item and input grating.

4 Discussion

The goal of this research was to use simulated Nb-doped $SrTiO_3$ memristors in the underlying architecture of a spiking-neuron model, that is able to approximate data by Pals et al. (2020), which has shown to resemble functional human behaviour. We started by investigating the performance of different learning rules, with and without memristors. Eventually we created a novel learning rule, mSTSP, that uses the short-term synaptic plasticity mechanism but programs the calcium part on memristors. Then we conducted the same experiments as in Pals et al. (2020), and compared the results. Since the results are particularly similar, we conclude that simulated Nb-doped $SrTiO_3$ memristors can effectively be used to resemble the calcium part in a working model of WM.

We ascertained that learning rules where the connection weight steadily increases during activation, i.e. BCM, are not able to learn to represent input patterns in a recurrent all-to-all neuron connection. In order to successfully facilitate the neurons, the connection weights need to be sufficiently increased. This, however, leads to persistent firing, which does not conform to the idea of activity-silent states. The interaction between resources and calcium in the STSP model has the unapparent benefit of countering the persistent firing of neurons during activation, resulting in the desired behaviour.

The calcium mechanism is thought of being the main contributor to enabling short- and long-term plasticity. Findings in this paper show that the resource mechanism in this setup is a second very important factor for creating a working spiking-neuron model. Only the depletion of resources, and thus the decrease of connection weights, resulted in the ability to keep information in memory.

Due to the limited increase in conductance by SET pulses, the memristor update is constrained, and the calcium level is only slightly increased. The gain, therefore, becomes the determining factor, setting the calcium level artificially high. This effectively leads to active neurons with a "high" calcium level and inactive neurons with a "low" calcium level. Given that there is no hierarchy between which neurons need to fire first during representation, but only the pattern needs to be encoded in neural representation and be reinforced in the neurons, a model with binary calcium "high" and "low" calcium will still produce satisfactory behaviour.

Interestingly, we had to use two different gains to resemble the original STSP results. The calcium of a single active neuron was inspected in order to mimic calcium behaviour during the experiment. The gain of $5 * 10^5$ was chosen considering it enabled the calcium of an active neuron to rise to 0.6 up until the moment of reactivation, which matched the 0.6 value at that specific time in the STSP experiment, see Section 3.7. Nonetheless, the calcium in the STSP experiment becomes 1 when the to-be-learned pattern is presented. This is not the case for the memristor-based mSTSP model, where the calcium gradually increases to 0.6. Since the connection weight in the STSP learning rule is dependent on both the calcium and the resources, see Equation 3, a lower calcium

level during the learning phase results in a lower connection weight at the period of reactivation. Evidently, this will lead to less neural spiking. This may cause detrimental results in the full experiment, where 1,344 trials are conducted across 30 subjects. Increasing the gain to $8.3 * 10^5$ does replicate the findings of Pals et al. (2020). This ensured that the calcium level reaches around 1.0 during the learning phase, which means that the connection weights are comparably incremented. An alternative explanation for the difference in gain might be due to the fact that the resistance of the memristors is not actively cleared after each experiment.

In this research, we opted to model only the increasing part of the calcium with memristors, instead of the entire resources and calcium mechanism. The primary reason for this is that the memristor-based model used in this research only allows for small changes in the conductance. This behaviour is not sufficient in order to mimic the volatile dynamics of the resources mechanism, where the resources are de- and repleted abruptly.

Likewise the decreasing part of the original calcium mechanics is strenuous to implement on the current memristors, given that only SET pulses are given to a single memristor. Dual memristors with different initialized resistances, where one memristor is pulsed to incorporate positive changes, while the other memristor is pulsed for negative changes, might be able to implement the full calcium mechanics.

A promising new type of memristor that could be used to model the entire STSP model is the diffusive memristor. The Ag-in-oxide memristor demonstrates diffusive dynamics (Wang et al., 2017), which relates to the movement of nanoparticles across the material. These memristors are based on metal atom diffusion and spontaneous nanoparticle formation. Under zero electrical bias the Ag atoms regroup, essentially returning the resistance and conductance to a baseline after stimulation. Wang et al. (2017) have shown that this type of memristor enables a direct emulation of both short- and long-term plasticity of biological synapses. These memristors may prove to be ideal for both the calcium as well as the resources mechanism of STSP. In case of the resources, a memristor could be utilized with a baseline resistance that represents the value 1.0, which is the initialized value of the resources. If the neuron fires, the memristor can be pulsed, that changes the resistance, in order to represent a 0.0 value (i.e. depleted resource). If then the neuron stops being active, and as a consequence of the diffusive dynamics, the resistance would then over time return to the baseline, representing the value 1.0 again. This would mean that we could integrate positive and negative changes with just a single memristor. Theoretically, we could then implement the diffusive memristors to represent the connection weights of the STSP mechanism.

If we would truly want to create a cognitive computer that is just as efficient as our brain, we would need to be able to simulate a flexible memory system, like WM. The hardware that then runs this simulation should ideally work as energy-efficient as our brain does. Memristors might be such devices, and this research shows that they can be used in the underlying architecture of a model that

simulates WM. In this research a subset of the model parameters was implemented on memristors, specifically the calcium of the neurons. If memristors are that much more efficient than transistors, we would preferably want the entire network, from input to output, to be represented on memristors. The simulated memristor used in this research is not sufficient for implementing all these different, and mostly dynamic, parameters. New types of memristors, like the diffusive memristor, pave the way for entire energy-efficient models that can be directly realised on the underlying hardware.

References

Adhikari, S. P., Yang, C., Kim, H., & Chua, L. O. (2012). Memristor bridge synapse-based neural network and its learning. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(9), 1426–1435.

Bate, R. T. (1988). The quantum-effect device: tomorrow's transistor? *Scientific American*, *258*(3), 96–101.

Bekolay, T., Bergstra, J., Hunsberger, E., DeWolf, T., Stewart, T., Rasmussen, D., ... Eliasmith, C. (2014). Nengo: a Python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics*, *7*(48), 1–13. doi: 10.3389/fninf.2013.00048

Bekolay, T., Kolbeck, C., & Eliasmith, C. (2013). Simultaneous unsupervised and supervised learning of cognitive functions in biologically plausible spiking neural networks. In *Proceedings of the annual meeting of the cognitive science society* (Vol. 35).

Bienenstock, E., Cooper, L., & Munro, P. (1982, February). Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex. *The Journal of neuroscience*, *2*, 32-48. doi: 10.1142/9789812795885_0006

Bostrom, N., & Sandberg, A. (2008). Whole brain emulation: a roadmap. *Lanc Univ Accessed January*, *21*(2008), 2015.

Brivio, S., Conti, D., Nair, M., Frascaroli, J., Covi, E., Ricciardi, C., ... Spiga, S. (2018, October). Extended memory lifetime in spiking neural networks employing memristive synapses with nonlinear conductance dynamics. *Nanotechnology*, *30*. doi: 10.1088/1361-6528/aae81c

Chua, L. (1971). Memristor-the missing circuit element. *IEEE Transactions on circuit theory*, *18*(5), 507–519.

Diamond, A. (2013). Executive functions. *Annual Review of Psychology*, *64*(1), 135-168. doi: 10.1146/annurev-psych-113011-143750

Ebong, I. E., & Mazumder, P. (2011). Cmos and memristor-based neural network design for position detection. *Proceedings of the IEEE*, *100*(6), 2050–2060.

Eliasmith, C. (2013). *How to build a brain: A neural architecture for biological cognition*. Oxford University Press. doi: 10.1093/acprof:oso/9780199794546.001.0001

Eliasmith, C., Stewart, T., Choo, X., Bekolay, T., Dewolf, T., Tang, C., & Rasmussen, D. (2012, November). A large-scale model of the functioning brain. *Science (New York, N.Y.)*, *338*, 1202-5. doi: 10.1126/science.1225266

Frascaroli, J., Brivio, S., Covi, E., & Spiga, S. (2018, May). Evidence of soft bound behaviour in analogue memristive devices for neuromorphic computing. *Scientific Reports*, *8*. doi: 10.1038/s41598-018-25376-x

Fuster, J. M., & Alexander, G. E. (1971). Neuron activity related to short-term memory. *Science*, *173*(3997), 652–654. doi: 10.1126/science.173.3997.652

Hu, M., Li, H., Wu, Q., & Rose, G. S. (2012). Hardware realization of bsb recall function using memristor crossbar arrays. In *Proceedings of the 49th annual design automation conference* (p. 498–503). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/2228360.2228448

Karpathy, A. (2016). *Cs231n convolutional neural networks for visual recognition.* Retrieved from https://cs231n.github.io/convolutional-networks/

Li, B., Wang, Y., Wang, Y., Chen, Y., & Yang, H. (2014). Training itself: Mixed-signal training acceleration for memristor-based neural network. In *2014 19th asia and south pacific design automation conference (asp-dac)* (pp. 361–366).

Li, C., Wang, Z., Rao, M., Belkin, D., Song, W., Jiang, H., ... Xia, Q. (2019, January 1). Long short-term memory networks in memristor crossbar arrays. *Nature Machine Intelligence*, *1*(1), 49-57. doi: 10.1038/s42256-018-0001-4

Mills, M. P. (2013, August). *The cloud begins with coal: big data, big networks, big infrastructure, and big power.* Retrieved from https://www.electronicsilentspring.com/wp-content/uploads/2016/02/Cloud_Begins_With_Coal.pdf

Mongillo, G., Barak, O., & Tsodyks, M. (2008). Synaptic theory of working memory. *Science*, *319*(5869), 1543–1546. doi: 10.1126/science.1150769

Moon, K., Lim, S., Park, J., Sung, C., Oh, S., Woo, J., ... Hwang, H. (2019). Rram-based synapse devices for neuromorphic systems. *Faraday Discuss.*, *213*, 421-451. doi: 10.1039/C8FD00127H

Moscote, A. (2013). *Diagrama de transistor.* Retrieved from https://en.wikipedia.org/wiki/File:Diagrama_de_Transistor_NPN.svg

Ni, L., Zichuan, L., Yu, H., & Joshi, R. (2017, April). An energy-efficient digital reram-crossbar based cnn with bitwise parallelism. *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, *PP*, 1-1. doi: 10.1109/JXCDC.2017.2697910

Pals, M., Stewart, T. C., Akyürek, E. G., & Borst, J. P. (2020, June). A functional spiking-neuron model of activity-silent working memory in humans based on calcium-mediated short-term synaptic plasticity. *PLOS Computational Biology*, *16*(6), 1-25. doi: 10.1371/journal.pcbi.1007936

Prodromakis, T., & Toumazou, C. (2011, January). A review on memristive devices and applications. *2010 IEEE International Conference on Electronics, Circuits, and Systems, ICECS 2010 - Proceedings*, 934 - 937. doi: 10.1109/ICECS.2010.5724666

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In *Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1: Foundations* (p. 318–362). Cambridge, MA, USA: MIT Press.

Serrano-Gotarredona, T., Masquelier, T., Prodromakis, T., Indiveri, G., & Linares-Barranco, B. (2013). Stdp and stdp variations with memristors for spiking neuromorphic learning systems. *Frontiers in Neuroscience*, *7*, 2. doi: 10.3389/fnins.2013.00002

Stevens, C., & Wang, Y. (1995). Facilitation and depression at single central synapses. *Neuron*, *14*, 795-802.

Strukov, D. B., Snider, G. S., Stewart, D. R., & Williams, R. S. (2008). The missing memristor found. *nature*, *453*(7191), 80–83.

Thomas, A. (2013). Memristor-based neural networks. *Journal of Physics D: Applied Physics*, *46*(9), 093001.

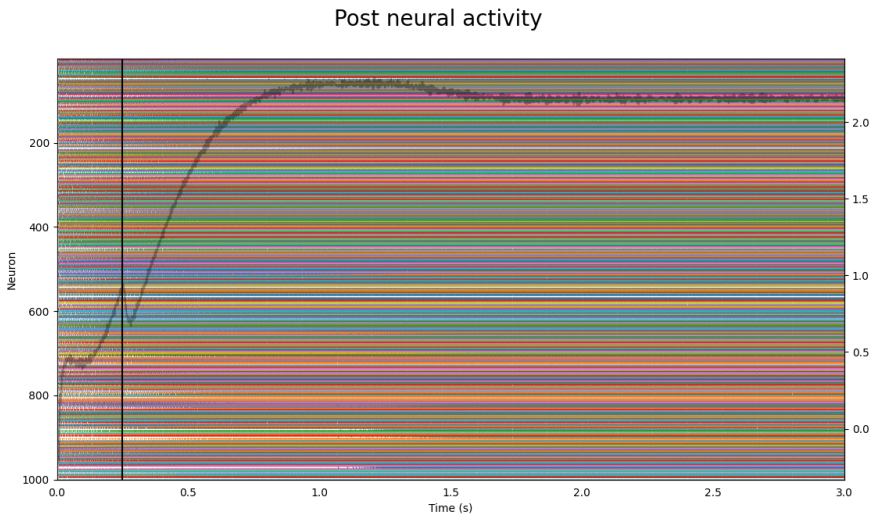
Tiotto, T. F., Goossens, A. S., Borst, J. P., Banerjee, T., & Taatgen, N. A. (2020). Learning to approximate functions using nb-doped srtio_3 memristors. *arXiv preprint arXiv:2011.02794*.

Wang, Z., Joshi, S., Savel'ev, S. E., Jiang, H., Midya, R., Lin, P., ... others (2017). Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing. *Nature materials*, *16*(1), 101–108.

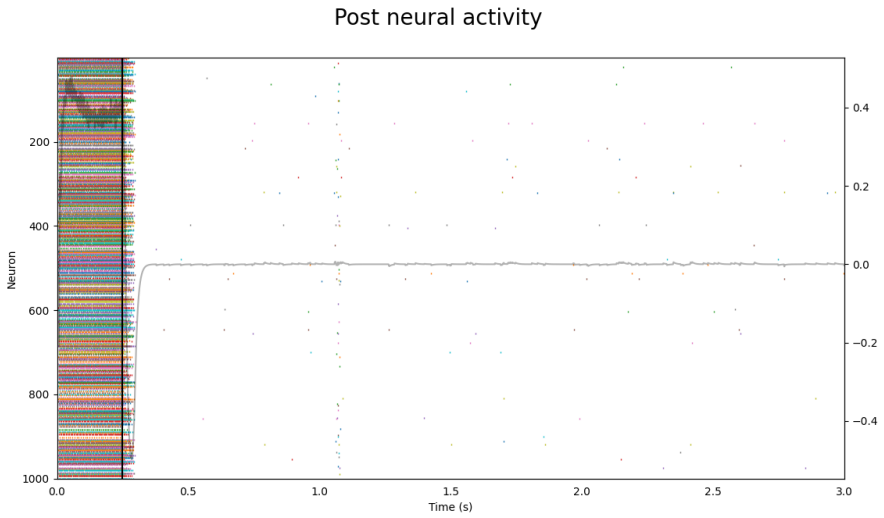
Wolff, M. J., Jochim, J., Akyürek, E. G., & Stokes, M. G. (2017, June 01). Dynamic hidden states underlying working-memory-guided behavior. *Nature Neuroscience*, *20*(6), 864-871. doi: 10.1038/nn.4546

Yao, P., Wu, H., Gao, B., Tang, J., Zhang, Q., Zhang, W., ... Qian, H. (2020). Fully hardware-implemented memristor convolutional neural network. *Nature*, *577*(7792), 641–646.

5 Appendix



(a) BCM with a learning rate of $2e-12$ with AdapLIF



(b) BCM with a learning rate of $1e-12$ with AdapLIF

Figure 20: Neural activity (grey) and spikes per neuron (colored) during simulation.

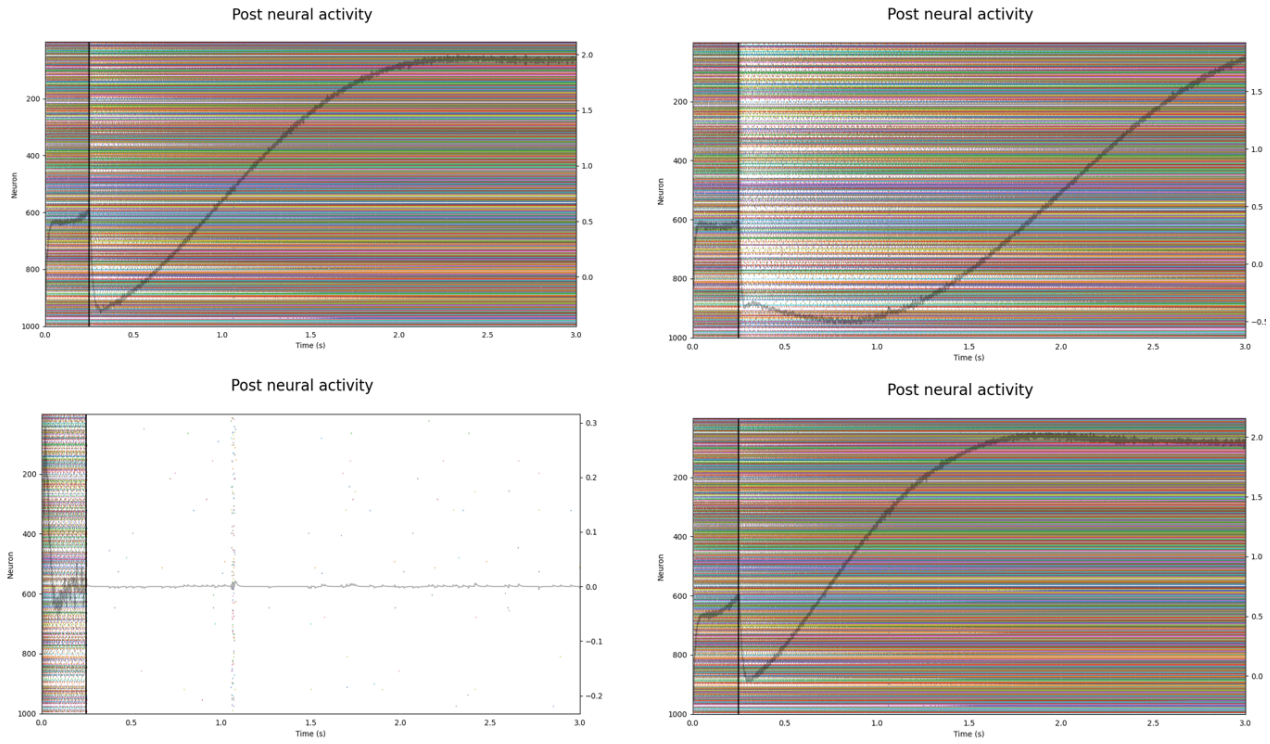


Figure 21: Neural activity (grey) and spikes per neuron (colored) during simulation with different parameters for AdapLIF neurons. All results are using BCM with a learning rate of $1e-12$. Top left figure shows AdapLIF with $tau_n = 0.5$ and $inc_n = 0.01$. Top right: AdapLIF with $tau_n = 0.01$ and $inc_n = 0.01$. Bottom left: AdapLIF with $tau_n = 0.1$ and $inc_n = 0.1$. Bottom right: AdapLIF with $tau_n = 0.1$ and $inc_n = 0.001$.

Table 1. Model parameters for each population. Default parameters in italics.

Parameter	**Sensory**	**Memory**	**Comparison**	**Decision**
Number of neurons	1000	1500	1500	1000
Neuron type	LIF	STSP-LIF	LIF	LIF
Dimensions	24	24	4	1
Intercepts	Uniform(0.01, .1)	Uniform(0.01, .1)	Uniform(.01, 1)	*Uniform(-1, 1)*
τ_{rc} (Membrane RC)	*0.02*	*0.02*	*0.02*	*0.02*
τ_{ref} (refractory period)	*0.002*	*0.002*	*0.002*	*0.002*
U*		0.2		
τ_D*		0.2		
τ_F*		1.5		

Figure 22: Reprinted with permission from Pals et al. (2020)

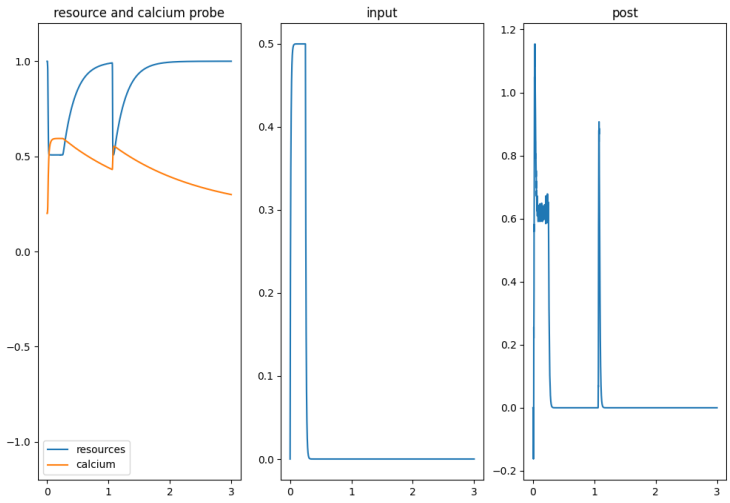


Figure 23: Simulation with STSP. Left: mean resources and calcium during simulation. Middle: input to the ensemble. Right: the output of the post ensemble. On the x-axis time (t) in seconds.

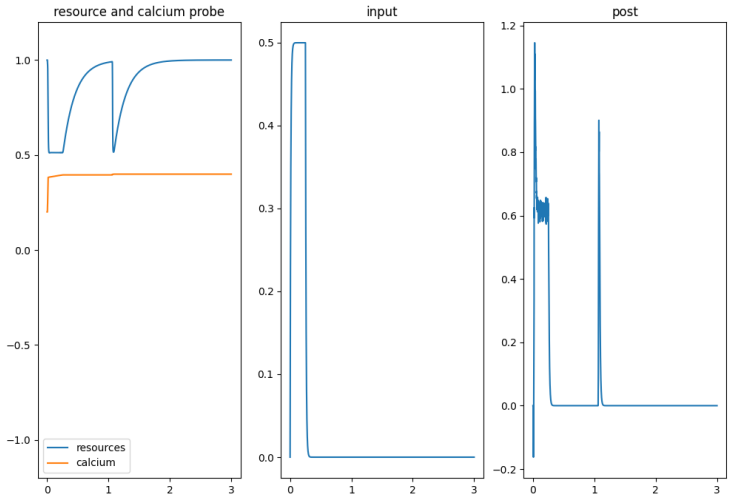


Figure 24: Simulation with mSTSP with a gain of $5 * 10^5$. Left: mean resources and calcium during simulation. Middle: input to the ensemble. Right: output of the post ensemble. On the x-axis time (t) in seconds.

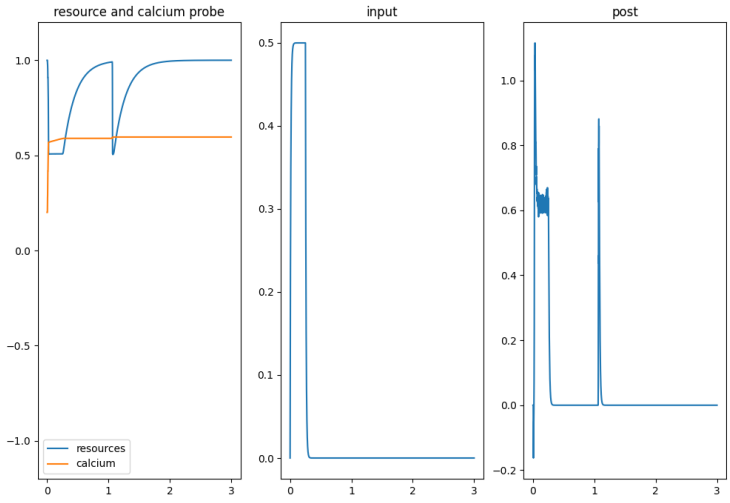


Figure 25: Simulation with mSTSP with a gain of $8.3 * 10^5$. Left: mean resources and calcium during simulation. Middle: input to the ensemble. Right: output of the post ensemble. On the x-axis time (t) in seconds.

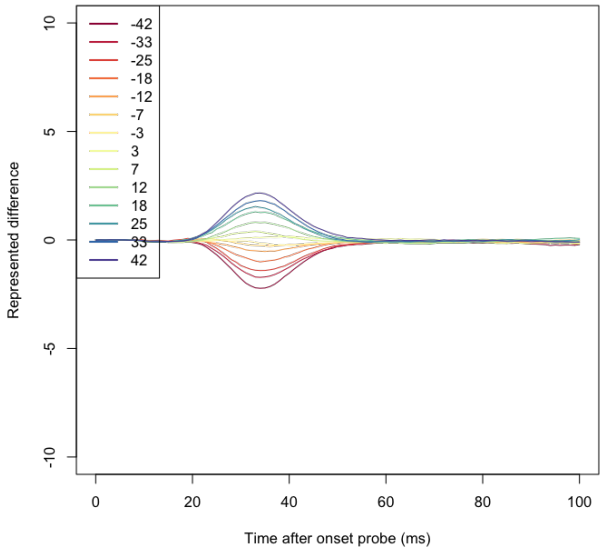


Figure 26: Represented difference in the decision population with mSTSP with a gain of $5 * 10^5$