Computational Modeling of Neuronal Plasticity : Online Course

Florence I. Kleberg and Jochen Triesch, FIAS 2017 - 2020

Introduction

Welcome to the Online Course "Computational Modeling of Neuronal Plasticity". In this course you will learn how computational neuroscientists use mathematical models and computer simulations to study different plasticity phenomena in the brain. During the course, you will program from scratch your own neuron model, a so-called leaky-integrate-and-fire (LIF) neuron model, and simulate it with a computer. Then you will add various neuronal properties and plasticity mechanisms to the model and study how they operate. This course will deepen your understanding of neural plasticity and prepare you for studying plasticity and learning in larger models such as neural networks.

General advice for the course

- It is advised to be familiar with the basics of the Python programming language before commencing the course. A good place to start is for instance: `http://www.tutorialspoint.com/python/`.
- The book "Theoretical Neuroscience" by Dayan and Abbott is an advised read [1]. It will help you to get acquainted with different concepts in simulating neurons and plasticity mechanisms. This course is in part based on the book.
- With mathematical equations, there is often more than one way to express the same thing. Equations shown in this course may look different from those in other resources, even if they convey the same meaning.
- The main idea is to model a neuron and its plasticity mechanisms from scratch, without the use of specialised neuronal modelling software such as Brian, NEURON or NEST. Of course, the use of python packages such as numpy, scipy, ipython and so on is allowed.
- Good programming practices: It is strongly advised to use good programming habits, such as object-oriented programming, a separate parameter file, and commenting in the code. An excellent guide for scientific programming can be found here:
  `http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000589`
- Good programming practices save a lot of time later on, including other people reading your code.
- Some students also use version control, to keep track of what changes have been made when. A popular tool is Git (`https://git-scm.com/`) which is often used with hosting services such as GitHub (`https://github.com/`) to maintain online repositories.
- When plotting or saving figures, it is important to record information about how the figure was generated, as working on a neuron model involves adjusting parameters over time, whose values are easily forgotten. For example, save a text version of the parameter file under the same name as the figure.

- Every now and then it may be good to update the structure of the code, or do a code cleanup if necessary. Do not be afraid to allocate time for improving code, it always ends up paying off in time.

1 The Leaky Integrate-and-Fire Neuron

We will begin our course with the basics of the neuron model at the centre of this course, the Leaky Integrate-and-Fire (LIF) Neuron. We will try different inputs and spike input patterns, and see how they shape the response of the neuron. The LIF neuron model only gives a very simplified description of a neuron's behaviour. In reality neurons are very elaborate structures with complex dynamics of membrane voltage. The approximation with a single value for the membrane voltage is called a "point-neuron" and is very frequently used in neural network models. More about this neuron model can be found in the book Theoretical Neuroscience [1], section 5.4: "Integrate-and-Fire Models".

1.1 A neuron with step current input

(Video 1.1) We start with the simplest version of the LIF neuron, which receives a constant current input. Constant current inputs were used in classical electrophysiology experiments to detect basic properties of the neuronal membrane. This neuron is characterized by the following membrane voltage equation:

$$\tau_{\text{mem}} \frac{dV}{dt} = E_{\text{leak}} - V + R_{\text{m}} I_{\text{ext}}. \tag{1}$$

Equation (1) describes the membrane voltage V, of which the units can be either in mV or Volt, although mV is used more frequently. τ_{mem} is the membrane time constant, 20 ms. E_{leak} is the reversal potential for the leak, -60 mV. R_{m} is the membrane resistance, 10 MΩ. Finally, I_{ext} is the external current applied to the neuron, 2.0 nA.

The neuron resets back to the reset value V_{reset} = -70 mV whenever its membrane voltage hits the threshold V_{thresh} = -50 mV. Set the program to record an "output spike" every time V hits the threshold. Using the above equation, make a simple python program to simulate a neuron for 200 ms, in response to 2.0 nA of constant applied current. Use Euler integration to solve the equations. It is helpful to make a separate python file in which the Euler integration is performed. At first, use an Euler integration time step of 0.1 ms. This integration time step will be looked at in more detail in a following section. Plot the membrane voltage over time. Show the spike times as vertical lines in the same plot. What is the spiking behaviour of the neuron? What happens if the external current is increased to 4.0 nA?

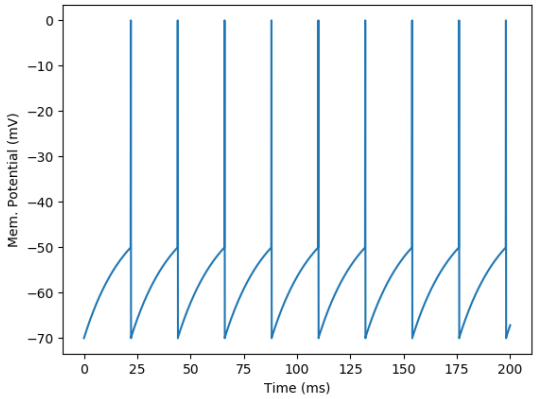

Figure 1: The membrane potential of the LIF neuron with a 2.0 nA current input should look like this. If the external current input is increased, then the neuron should spike more frequently.

1.2 Excitatory and inhibitory inputs

(Video 1.2) We now remove the constant input current and add inputs that more closely resemble the synaptic inputs neurons would receive from other neurons. The equation for the neuron is now:

$$\tau_{\text{mem}} \frac{dV}{dt} = E_{\text{leak}} - V + g_{\text{e}}(E_{\text{e}} - V), \tag{2}$$

where we have added a conductance g_e for an excitatory input synapse. This conductance obeys the following equation, which contains a simple exponential decay:

$$\frac{dg_{\text{e}}}{dt} = -\frac{g_{\text{e}}}{\tau_{\text{e}}} + w_{\text{e}} \sum_{s=1}^{N} \delta(t - t_s). \tag{3}$$

The g_{e} term represents the excitatory input conductance that comes from another neuron. Note that $\delta(s)$ is the Dirac delta function. E_{e} is the reversal potential for excitatory (depolarizing) inputs, here let us set it to 0 mV. τ_{e} is the postsynaptic potential (PSP) time constant, let us use $\tau_{\text{e}} = 3$ ms.

For each spike t_s that arrives, the conductance is increased by an amount w_{e}, which is the strength, or weight, of the excitatory synapse. The equation above is written for a single excitatory synapse. In the case of multiple synapses onto a postsynaptic neuron, g_{e} is increased by each arriving spike by the respective synaptic weight of that synapse. The next step is to adapt the model used in the previous section to include the excitatory input of another neuron, in the form of a single synapse as described by equation (3). Assume periodic (regularly spaced) spikes with firing rate 6 Hz. Set w_{e} to 0.5. Make a plot of the membrane potential again, and look at the spiking behaviour. What happens to the spiking of the neuron if w_{e} is increased? A larger excitatory synaptic weight should lead to larger depolarization and more frequent postsynaptic spikes.

Let us also add an inhibitory synapse to the same neuron:

$$\tau_{\text{mem}} \frac{dV}{dt} = E_{\text{leak}} - V + g_{\text{e}}(E_{\text{e}} - V) + g_{\text{i}}(E_{\text{i}} - V), \tag{4}$$

$$\frac{dg_{\mathrm{i}}}{dt} = -\frac{g_{\mathrm{i}}}{\tau_{\mathrm{i}}} + w_{\mathrm{i}} \sum_{p=1}^{P} \delta(t - t_p). \tag{5}$$

Where g_{i} is the inhibitory conductance, τ_{i} is the PSP time constant of inhibition, 5 ms, and E_{i} the reversal potential for inhibition, -80 mV, w_{i} is the strength of the inhibitory synapse, and t_p the time of an inhibitory input spike. Now add the inhibitory synapse to the neuron by combining equations (3-5), and assume periodic spikes with frequency 6 Hz for excitatory, and 3 Hz for inhibitory input synapses. Show the membrane potential of the neuron for 2 seconds. In order to obtain any spikes in the neuron, the excitatory synapse should be sufficiently strong. Therefore, set $w_{\mathrm{e}} = 3.0$, and $w_{\mathrm{i}} = 3.0$ to match. In following units we will deal with more realistic settings of multiple synaptic inputs.

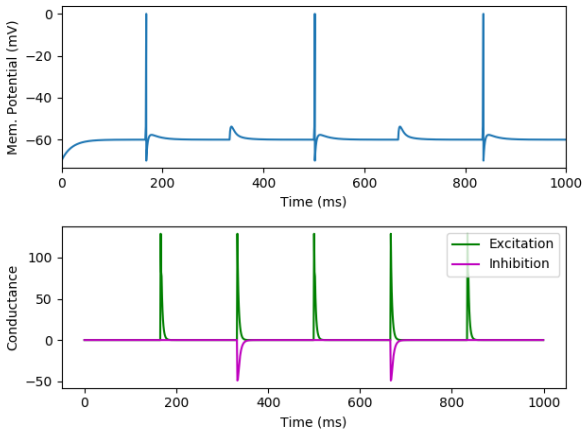

Figure 2: After adding the periodic excitatory and inhibitory synaptic inputs, the membrane potential follows the curve shown in the top figure. The excitatory and inhibitory conductances g_{e} and g_{i} are shown in the bottom figure. As can be seen from the figures, when inhibition arrives, it prevents an output spike caused by excitation.

1.3 Integration time step

(Video 1.3) We have made some simulations of the neuron using various inputs. We can make a small sidestep to ensure our simulation is precise enough. Let us test the influence of the integration time step by looking at the excitatory conductance g_e. The analytical solution of the differential equation of g_{e}, assuming $g_{\mathrm{e}} = 1$ at t=0, is

$$g_{\mathrm{e}}(t) = e^{\frac{-t}{\tau_{\mathrm{e}}}}. \tag{6}$$

Plot the analytical solution in a graph. Now integrate the equation by using the Euler method, assuming

$$g_{\mathrm{e}} \rightarrow g_{\mathrm{e}} + 1, \tag{7}$$

at t=0. This corresponds to the response to a single input spike at t=0, and a synaptic weight of 1. In the same graph, let us plot g_e obtained from the Euler integration, in a different colour. While keeping a fixed simulation time step of 1 ms, you can try an integration step of 0.001 ms, 0.01 ms, 0.1 ms, and 1 ms. Finally, select the optimal time step based on accuracy and computation time. Do not forget to zoom in closely to the curve in your figure to observe smaller inaccuracies.

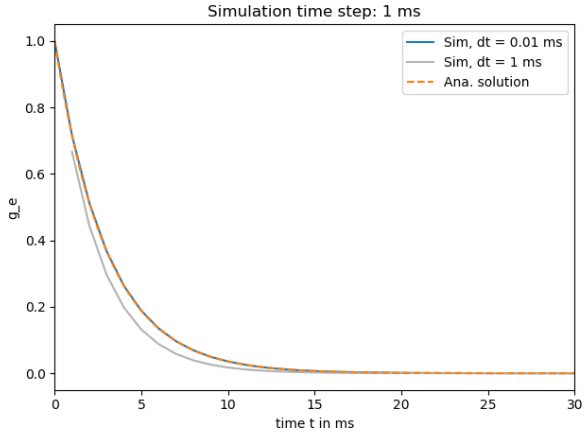

Figure 3: When testing the accuracy of the simulation with different integration time steps, the accuracy is high when the exact solution and the simulation yield a highly similar or identical result, as visualised here by the integration time step of 0.01 ms (blue). On the other hand, an integration time step of 1 ms is highly inaccurate (grey).

1.4 Poisson spikes

(Video 1.4) We now consider that the spikes the neuron receives resemble the irregular patters that are recorded in experiments. For example, a frequently used model for spike trains is the Poisson process. This means that spikes occur independently of each other. If the average firing rate remains stable over time, it is called a homogeneous Poisson process, which is what we will apply here. Since a property of a Poisson spike train is that the inter-spike intervals are distributed exponentially, a sequence of input spike times can easily be obtained by randomly sampling inter-spike intervals from an exponential probability density function.

Now expand the neuron model with N_e =10 excitatory and N_i =10 inhibitory inputs, with Poisson spike trains, each with a firing rate (intensity) of 10 Hz. In this model of multiple excitatory and inhibitory inputs, the equations now become:

$$\tau_\mathrm{mem}\frac{dV}{dt} = E_\mathrm{leak} - V + \sum_{m=1}^{N_\mathrm{e}} g_{\mathrm{e},m}(E_\mathrm{e} - V) + \sum_{n=1}^{N_\mathrm{i}} g_{\mathrm{i},n}(E_\mathrm{i} - V), \tag{8}$$

$$\frac{dg_{\mathrm{e},m}}{dt} = -\frac{g_{\mathrm{e},m}}{\tau_\mathrm{e}} + w_{\mathrm{e},m}\sum_{s=1}^{N_m}\delta(t - t_s), \tag{9}$$

$$\frac{dg_{\mathrm{i},n}}{dt} = -\frac{g_{\mathrm{i},n}}{\tau_\mathrm{i}} + w_{\mathrm{i},n}\sum_{p=1}^{P_n}\delta(t - t_p). \tag{10}$$

When the neuron receives excitation and inhibition, and excitation and inhibition are balanced, the output of the neuron should be irregular [2]. You can measure the irregularity of the output spikes by plotting the inter-spike intervals (ISIs). If the distribution of ISIs follows an exponential shape, and if the coefficient of variation (CV, the standard deviation/mean) of the ISIs is 1, the output is irregular. A perfect Poisson process has exponentially distributed ISIs, and a CV of the ISIs of 1. If the CV of the ISIs deviates from 1, this can be a sign of more regular firing or bursting.

Start with setting all $w_{\mathrm{e},m} = w_{\mathrm{i},n} = 0.5$. Is the spiking output of the neuron irregular? It will be necessary to adjust w_{e} or w_{i} to generate a more/less regular output from the neuron. Since the τ_{i} is larger than τ_{e}, you will need to increase w_{e} a little to compensate for inhibition and obtain excitation/inhibition balance in the input to the neuron. Show plots of the neuronal membrane V, spike times and show the distributions of ISIs and CVs of the ISIs. Insert the mean of the histograms in the plot titles. The number of ISIs in each trial is related to the output firing rate. The output firing rate, in turn, is mainly shaped here by the strength of the input weights and firing rates. Generally speaking, input spike timing also plays a role, but we will not address this yet, as here all input spike trains are Poisson with stationary firing rates. Make sure to run a sufficiently long simulation or a number of separate shorter runs ('trials'), so that you have enough ISI datapoints to clearly see the distribution of ISIs. For the CV of the ISIs, try to obtain at least 50 independent trials of 10 seconds each, so that you have 50 CV datapoints for the distribution, where each CV is based on at least 20 ISIs.

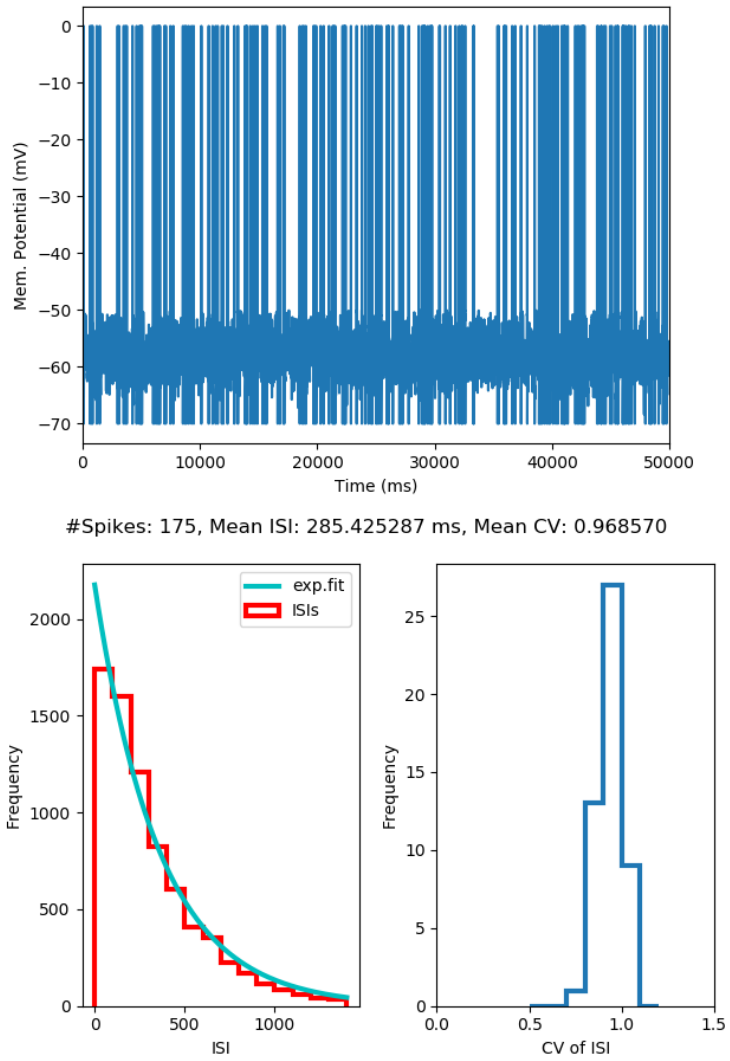

Figure 4: The LIF neuron with Poisson distributed synaptic inputs. Top: The LIF membrane potential shows irregular-looking fluctuations due to the Poisson input spikes. Bottom left: The distribution of inter-spike intervals (ISIs) of the LIF neuron after repeating 50 trials with Poisson inputs resembles a decaying exponential, indicating the irregularity of the output spikes. The exponential fit is found using `curve_fit` from Scipy. Bottom right: The CVs of the ISIs from separate trials, shown here in a histogram, are distributed close to 1.0, which would be expected for a Poisson process. One can obtain a CV close to 1.0 by balancing excitation with inhibition onto the neuron.

2 Adaptation in spiking behaviour

2.1 Spike-Rate Adaptation

We are going to add a new current to the model neuron that is responsible for a phenomenon called Spike Rate Adaptation (SRA), also known as spike frequency adaptation. This potassium-mediated current causes a neuron to decrease its output spikes over time, during constant current input. We can think of this as a fast homeostatic plasticity mechanism that prevents a neuron from becoming overly active. The SRA current is also hyperpolarizing, like inhibition. However, its dynamics are slower. The SRA is added to the LIF neuron equation in the following way:

$$\tau_{\text{mem}} \frac{dV}{dt} = E_{\text{leak}} - V + g_{\text{e}}(E_{\text{e}} - V) + g_{\text{i}}(E_{\text{i}} - V) + g_{\text{SRA}}(E_{\text{k}} - V), \tag{11}$$

and the conductance of SRA

$$\frac{dg_{\text{SRA}}}{dt} = -\frac{g_{\text{SRA}}}{\tau_{\text{SRA}}} + \sum_{k=1}^{K} \delta(t_{\text{post},k} - t)\Delta g_{\text{SRA}}. \tag{12}$$

which indicates that after each postsynaptic spike $t_{\text{post},k}$, with a total of K postsynaptic spikes, g_{SRA} is increased by a fixed amount Δg_{SRA}. In that way, g_{SRA} builds up when output spiking is frequent, contributing to hyperpolarizing the neuron.

Let us remove the synaptic inputs temporarily, and replace them by a current step input, as in our very first exercise in Sec. 1.1.:

$$\tau_{\text{mem}} \frac{dV}{dt} = E_{\text{leak}} - V + R_{\text{m}} I_{\text{ext}} + g_{\text{SRA}}(E_{\text{k}} - V), \tag{13}$$

In this current step input, the membrane resistance R_{m} is set to 10 MΩ, and the external current I_{ext} to 1.45 nA. Furthermore, set $\Delta g_{\text{SRA}} = 0.06$, $\tau_{\text{SRA}} = 100$ ms, and $E_{\text{k}} = -70$ mV. Also ensure in your program that g_{SRA} is always non-negative. Now plot the membrane potential and the Δg_{SRA} over time. Also show the frequency of output spikes, by taking each inter-spike interval in seconds and taking the inverse. Compare these results to the case where SRA is deactivated in the same model. To do this, simply set Δg_{SRA} to zero. Is the frequency of output spikes decreased compared to when SRA is deactivated?

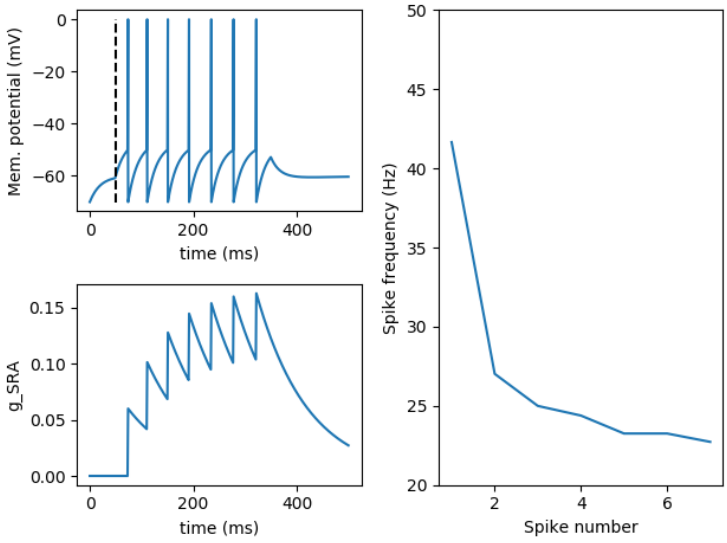

Figure 5: A LIF neuron with spike-rate adaptation (SRA) and current step input applied in the interval 50 to 350 ms. Top left: The membrane potential and spikes are shown. The dotted line indicates the moment the step current has started. The spikes can be seen to be delayed more and more over time. Bottom left: The SRA conductance g_{SRA} builds up over time with each output spike due to potassium-mediated hyperpolarisation, and this conductance decays back to zero when spiking ceases. Right: The frequency of the spikes, inversely proportional to the spike delay, is shown one by one for each output spike. In the presence of SRA, spikes are delayed as SRA conductance builds up, leading to a decrease in output frequency. **JT: I think the participants should reproduce a plot here like Fig 5.6(c) in Dayan and Abbott and compare this to the case without SRA.**

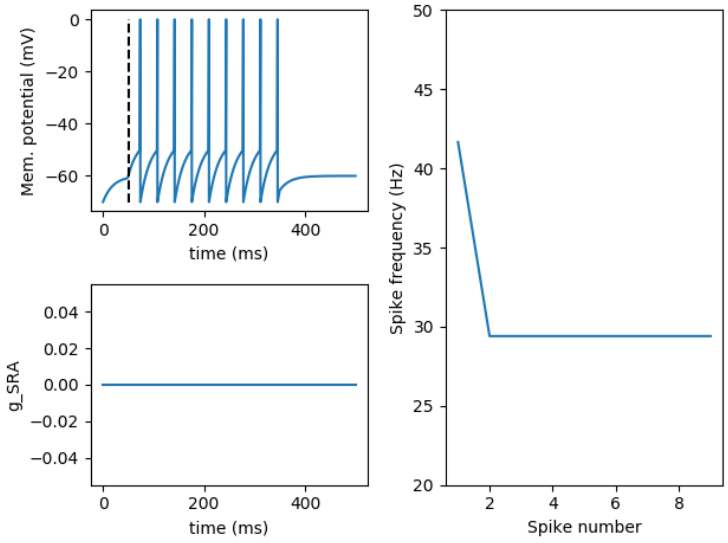

Figure 6: A LIF neuron with current step input applied in the interval 50 to 350 ms but without spike-rate adaptation (SRA). Top left: The membrane potential and spikes are shown. The dotted line indicates the moment the step current has started. The output spikes do not bcome more delayed over time. Bottom left: The SRA conductance g_{SRA} is zero because the SRA has been disabled in this neuron. Right: The frequency of the spikes, inversely proportional to the spike delay, is shown one by one for each output spike. In the absence of SRA, the neuron continues to fire regular ouput spikes in response to the step current input. Output frequency is higher than in the neuron with SRA, and this equilibrium frequency is reached directly after the first spike.

2.2 Refractory Period after spiking

In a real neuron, there usually exists a refractory period after a spike, a result of Na^+ channel inactivation followed by an outward K^+ flux from the neuron, leading to temporary hyperpolarization. The equation for SRA that you have already implemented can also be used to model a refractory period, simply by using different parameters. To do so we decrease the value for τ_{SRA}, increase the value for Δg_{SRA} and rename them τ_{RP} and Δg_{RP} respectively. Also, we keep the reversal potential E_{k} and rename it E_{RP}. For instance, try $\tau_{\mathrm{RP}} = 50$ ms and $\Delta g_{\mathrm{RP}} = 1.2$. Then replace the step current input by the excitatory and inhibitory synaptic inputs with Poisson spiking, as in Sec. 1.4. Now verify the response of the neuron and compare it to the neuron in Sec. 1.4 which did not have a refractory period. With a refractory period, spiking of the neuron should be more regular than without. Plot a histogram of the ISIs of the neuron's output spikes, and a histogram of the CVs of the ISIs, as before. Does the distribution of ISIs look different for this neuron compared to Sec. 1.4? Is the CV lower for a neuron with a refractory period?

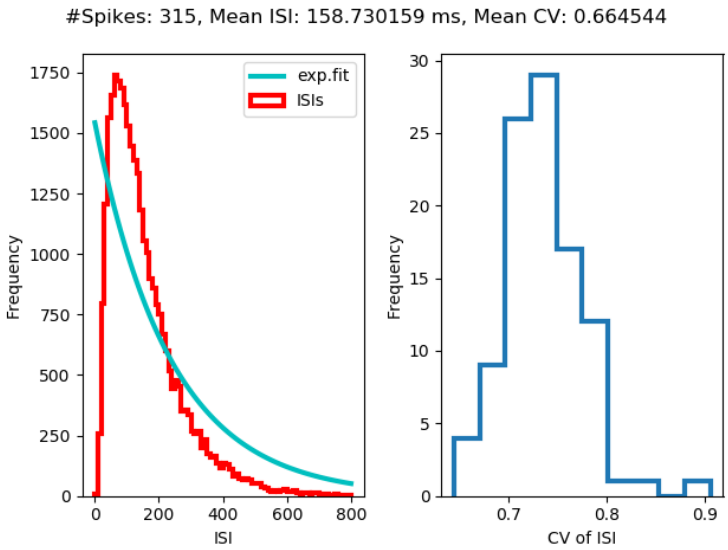

Figure 7: A LIF neuron with a refractory period of 50 ms and Poisson spiking inputs. Left: The distribution of ISIs clearly deviates from a decaying exponential, showing a lack of very small intervals due to the refractory period. Right: the CV of ISIs is decreased on average, compared to the neuron without the refractory period in Sec. 1.4, indicating that the neuron fires more regularly due to the presence of a refractory period.

3 Spike-Timing Dependent Plasticity (STDP)

3.1 STDP

Now we shall implement some long-term plasticity. Spike-Timing dependent Plasticity (STDP) is a type of neuronal activity-dependent synaptic plasticity, in which the strength of a synapse increases or decreases depending on the time difference between a spike from the presynaptic and a spike from the postsynaptic neuron (from now on we refer to these spikes as pre-spikes and post-spikes, respectively). While different subtypes of STDP can be found in the literature [3], the most widely observed and discussed variety is "Hebbian" or asymmetric STDP, which entails long-term potentiation (LTP) after a pre-post spike pair, and long-term depression (LTD) of the synaptic weight after a post-pre spike pair. Excitatory synapses in the hippocampus and cortex are subject to LTP and LTD via Hebbian STDP [4, 5].

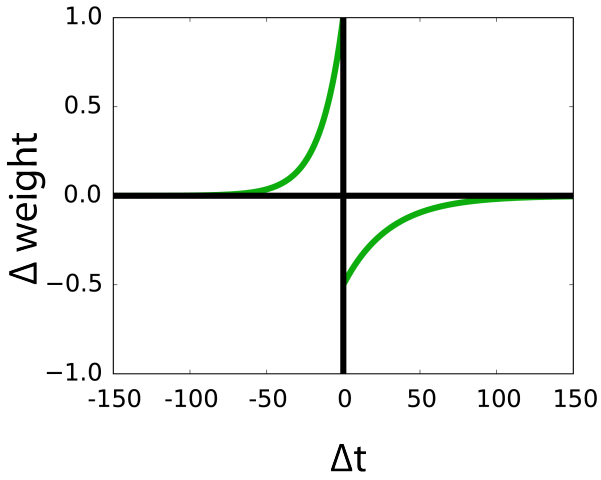

Figure 8: The STDP "window". Left curve, LTP. Right curve, LTD. The value Δt here is the time difference pre-spike - post-spike. When reading literature, keep in mind that some authors switch the sign of Δt.

The dependence on the difference in spike timing can be modeled by an exponentially decaying shape as a function of the time difference between a presynaptic and a postsynaptic spike $\Delta t = t_{\text{prespike}} - t_{\text{postspike}}$ (Fig. 8). The change in the excitatory synaptic strength (weight) w_{e} at the moment of a pre- or postsynaptic spike obeys:

$$\Delta w_{\text{e}} = \begin{cases} A_{\text{LTP}} \exp(\frac{\Delta t}{\tau_{\text{LTP}}}) & \forall \Delta t < 0 \\ A_{\text{LTD}} \exp(-\frac{\Delta t}{\tau_{\text{LTD}}}) & \forall \Delta t > 0 \\ 0 & \forall \Delta t = 0 \end{cases} \tag{14}$$

Fig. 8 uses τ_{LTP} = 17 ms, A_{LTP} = 1.0, τ_{LTD} = 34 ms, and A_{LTD} = - 0.5. These values, leading to a perfect compensation of LTP by LTD in terms of window area, are frequently used, though other values are possible. There are several ways to implement STDP in computational models distinguished by which pre- and post-synaptic spike pairs are allowed to interact to trigger changes in synaptic efficacies. The two most often considered mechanisms are "all-to-all," where every pre-synaptic spike interacts with every post-synaptic spike to trigger a weight change, and "nearest-neighbour," where pre/post spikes only interact with the closest post/pre spikes. There are several possible variants of nearest-neihbor schemes reviewed by Morrison et al. [6]. Here, you should consider the "narrow" scheme depicted in Fig. 7 C of that publication. You can do this using two buffer variables that store the times of the last presynaptic and postsynaptic spikes.

Using the single LIF neuron model from unit 1.4 (without refractory period or SRA), leave out the inhibitory inputs for simplicity, or set their synaptic weight to zero. Reduce the number of excitatory inputs to only 2 and apply the STDP rule, as shown in the figure and the equations above, in these two synapses. Importantly, set A_{LTP} to 0.05 and A_{LTD} to -0.025, such that the change in weight is slow enough for us to see the evolution of the weights. Note also that if the initial weight in combination with the input firing rate and the number of inputs is too weak, there will be no post-spikes. Without post-spikes, STDP is not activated and the weight will remain unchanged, so in this model the synaptic drive must be strong from the start of the simulation to see any effect of STDP. Here, use an initial value

of $w_e = 1$ for each synapse, and apply two Poisson inputs with firing rate 5 Hz. Plot the value of the weight changing over time.

You should now see that the weights are growing in an unlimited fashion. As a result of the growing weight, the postsynaptic neuron will demonstrate an increasing firing rate. This problem is dealt with in later units, for now just put a maximum weight value $w_{max} = 6$, above which the weight cannot increase. Uncontrolled growth of synaptic weights is a property of pure additive STDP models in excitatory synapses: since the pre-post spike combination is related to potentiation, STDP rewards causality in the case of an excitatory synapse. The increased excitatory weight then enhances the probability that the pre-spike causes a post-spike, and hence a positive feedback loop is created. Therefore, the weight will mostly undergo LTP and little LTD, despite the integral of the STDP window being 0. Possible mechanisms that the brain employs to limit this positive feedback have been frequently addressed in the last decade in modeling studies concerned with STDP.

Now change one of the Poisson inputs to a higher rate of 8 Hz. You now have two inputs with STDP, one with a higher and one with a lower firing rate (5 and 8 Hz). What do you see in the weights? If it works well, the weight from the 8 Hz input should "win the race".

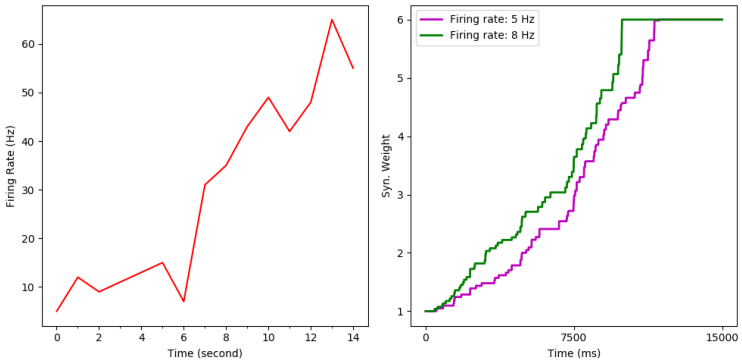

Figure 9: A LIF neuron with Poisson inputs and STDP in two excitatory input synapses with different firing rates (5 and 8 Hz). Left: The neuron is getting increasingly depolarised over time, resulting in more and more output spikes. Right: The cause of the increase in spiking is the growth in excitatory synaptic weights. Importantly, the synaptic weight of the strongly firing input (8 Hz) increases faster than that of the more weakly firing input (5 Hz).

3.2 Correlated Spike Trains

When correlations between spike trains are nonzero, STDP can create additional competition between groups of synaptic inputs. Let us first see how we can implement such correlations in spike times. There are various ways to create correlations between spike trains, while maintaining Poisson-like statistics within the spike trains themselves. For example, here we are going to create Poisson spike trains that have instantaneous and exponential correlations with each other. We discuss one method, for creating such correlations, which has been described by Brette and colleagues [7] (Method II).

Consider a Poisson spike train S_{source} with firing rate r_{source}. One way to create another spike train that has nonzero spike-time correlations with S_{source} is to simply copy spike times from S_{source} to the new spike train, which we call S_{new}. If the spikes are copied with probability p, then the firing rate of any new spike train will be $p \times r_{source}$. More generally,

$$r_{\text{new}} = \sum_{k=1}^{M} p_k r_k, \tag{15}$$

for M different source spike trains. In this example, we just stick to one source spike train, from which we create multiple new spike trains. If $p = 1$, S_{new} is simply identical to S_{source}. However according to equation 14, for any $p < 1$, thinning of a Poisson spike train will result in $r_{\text{new}} < r_{\text{source}}$. What if we want the spike trains to have the same or higher firing rates? In order to compensate for the loss of spikes, the firing rate can be increased to reach r_{target} by adding additional spikes. This is done by superimposing another Poisson spike train S_{noise} onto S_{new}, with firing rate $r_{\text{noise}} = r_{\text{target}} - p \times r_{\text{source}}$. If this process is repeated independently for a number of S_{new} with the same p, S_{source}, and independent S_{noise} for each spike train, one obtains a group of correlated spike trains that each fire with Poisson statistics and firing rate r_{target}. Now, what are the correlations between such spike trains?

The cross-covariance between any two spike trains S_i and S_j is

$$CCVF(s) = \langle S_i(t) S_j(t+s) \rangle - \langle S_i(t) \rangle \langle S_j(t) \rangle \tag{16}$$

The angular brackets denote an average over time. This function is 0 for two independent Poisson spike trains, and is $r\delta(s)$ for two identical Poisson spike trains with firing rate r (this is also called the autocovariance).

The cross-covariance function between two new spike trains generated from the same source spike train by the thinning method described above, is

$$CCVF(s) = p^2 r \delta(s) \tag{17}$$

with $r = r_{\text{source}} = r_{\text{target}}$. The area under the cross-covariance function, $p^2 r$, normalised by the firing rate r, is the correlation c. We see here that $c = p^2$. Therefore, if we choose $p = \sqrt{c}$, the correlations between the spike trains will be c. The method described above generates instantaneous correlations between spike trains. You can also make exponential cross-covariance between spike trains. In order to do so, you can shift the spikes randomly after copying them from S_{source}. Randomly shifting the spikes does not change the properties of single spike trains: they remain Poisson and their firing rate does not change. Choose the random shift from an exponential probability density $P(x)$ for every spike.

$$P(x) = \frac{1}{\tau_{\text{c}}} e^{-\frac{x}{\tau_{\text{c}}}} \tag{18}$$

In this case, the cross-covariance function is a convolution of the probability density function:

$$CCVF(s) = \int P(x) P(x+s) ds = \frac{1}{2\tau_{\text{c}}} e^{-\frac{|s|}{\tau_{\text{c}}}} \tag{19}$$

We will refer to this type of correlation as exponential correlation. The variable τ_{c} controls the width of the correlation. First let us make groups of spike trains that have instantaneous correlations, without random spike time shifting. Create a group of 10 instantaneously correlated spike trains with $c = 0.1$, and a group of 10 instantaneously correlated spike trains with $c = 0.1$ and Show the cross-correlogram (the cross-correlation histogram over different time lags) for each group; the correlation structure should be visible there. Within groups, you should see a large peak at zero lag, which should be absent when taking cross-correlations between groups.

Now apply exponential cross-correlations, by using $\tau_{\text{c}} = 20$ ms. How has the cross-correlogram changed? You should now see an exponential-like decay from both sides of the peak at lag zero.

In the next unit, we will apply these correlated spike trains to the neuron model with STDP.

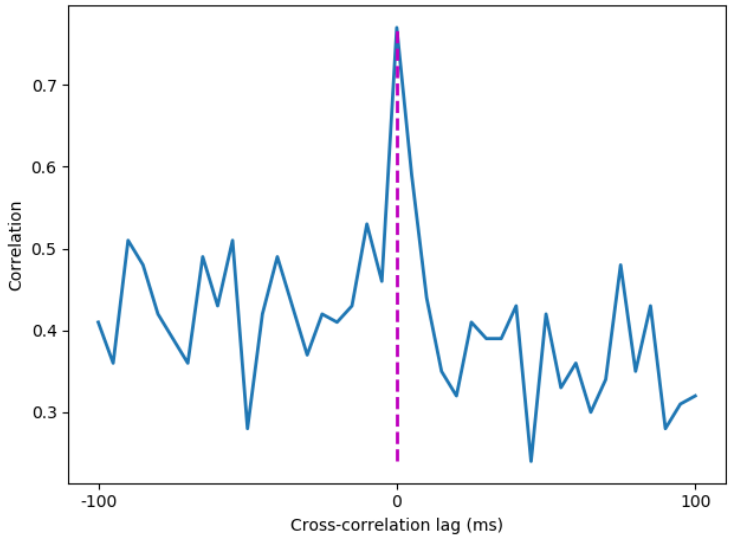

Figure 10: An example of a cross-correlogram between two 10 Hz Poisson spike trains of length 15 seconds. The spike trains are correlated with c = 0.2, and spike correlation jitter τ_c = 20 ms. Spikes are binned in 5 ms bins. The vertical dotted line indicates the zero lag point. Although the spike correlation is jittered and not instantaneous, a peak can still be seen at zero lag.

3.3 Correlations and STDP

We have looked at STDP and at correlations in spike times separately, now let us put the two together. Create two separate groups with each 10 excitatory spike trains that have instantaneous correlation, and use $r_{\text{source}} = r_{\text{target}} = 10$ Hz, and $c = 0.1$. Now input them onto the LIF neuron and put STDP in all the synapses. Set the STDP learning rate to be lower, $A_{\text{LTP}} = 0.02$ and $A_{\text{LTD}} = -0.01$, and start all excitatory weights at $w_{\text{e}} = 0.5$. Also include 10 inhibitory inputs with weight 1 and a firing rate of 10 Hz, without STDP.

What do you see in the weights of the excitatory input groups? Normally, the two groups should compete for the weight increase onto the neuron. Do both groups reach the maximum weight? Now change the correlation between the spikes: group 1, $c_1 = 0.1$ and group 2, $c_2 = 0.2$, but maintain the same firing rate for all inputs. Simulate for long enough so you can see well what happens to the weights of the two groups. If everything is correct you should see that correlations as well as firing rates can influence the competition between groups of weights.

If you add exponential cross-covariance, but do not change $c1$ and $c2$, does it influence the weight evolution of the two competing groups? You may have to adjust the weights and learning rates to produce reasonable output firing rates. Another effect you can look at is how the difference in initial weight in the two groups can influence weight competition by STDP. You can also test if an initial advantage in the weights for one group can overcome a weaker correlation or lower firing rate of that group in the competition for synaptic strength.

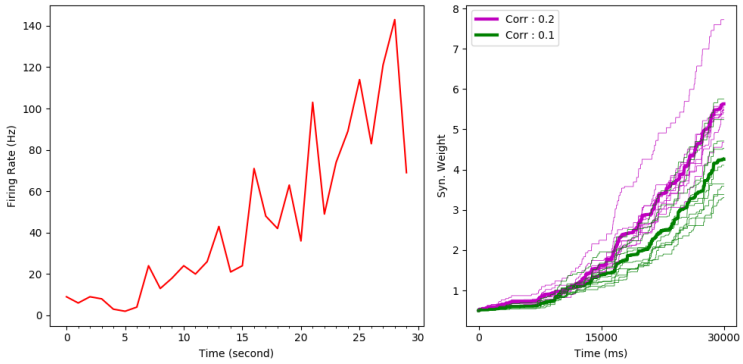

Figure 11: A LIF neuron with two separately correlated excitatory input groups, one with c = 0.2 and one with c = 0.1. Left: The firing rate of the neuron increases over time. Time bins of 1 second are used to record the spikes. Right: Synaptic weights from two groups of excitatory inputs are shown evolving over time through STDP. The thin lines show individual weights, the thick line the average from the corresponding group. Although the firing rates of the inputs groups are identical, the weights from the group with higher correlation increase faster on average.

4 Homeostasis

4.1 Synaptic Normalisation

Besides STDP, there exist forms of synaptic plasticity that aim at restoring or maintaining a certain activity level in a homeostatic manner. For instance, through homeostatic plasticity, an overly active neuron may weaken its excitatory inputs or an overly silent neuron may strengthen its excitatory inputs. The effect of homeostasis on the activity of a neuron or network of neurons depends strongly on its timescale. Some forms of homeostatic plasticity rescale synapses over hour- or day-long timescales in a multiplicative manner, as shown in rodents after growth or shrinkage of synaptic sizes during experience-dependent synaptic plasticity [8, 9]. We refer to this slow homeostatic plasticity as "synaptic scaling". On the other hand, there is also fast synaptic homeostasis [10], which is able to keep runaway excitation under control [11]. The fast synaptic homeostatic plasticity is most likely heterosynaptic, meaning that the change in one synaptic weight depends on the collective dynamics or states of synapses that are connected to the same postsynaptic target. We call the fast homeostatic plasticity "synaptic normalisation". Until more data becomes available, we assume that synaptic normalisation acts multiplicatively [12], like synaptic scaling. The major advantage of multiplicative synaptic normalisation is that the proportional difference between the smaller and larger weights is conserved. Multiplicative synaptic normalisation therefore does not erase pre-existing memories if it is implemented in a network of neurons. An example of multiplicative synaptic normalisation is the following:

$$w_{j,t+1} = w_{j,t}\left(\frac{W_{\text{tot}}}{\sum_{k=1}^{N} w_{k,t}}\right), \tag{20}$$

Where j and k refer to each synapse onto the neuron, and N is the total number of synapses onto the neuron. The parameter W_{tot} is the value of the total allowed weight onto the neuron, which can be assumed to be flexible over time but we here keep fixed for simplicity. Let us say we wish to have a normalised weight at timestep $t+1$ in the simulation. The weight $w_{j,t}$ then is the weight just before a normalisation event. After applying equation (20), we obtain the weight just after normalisation, $w_{j,t+1}$,

for the timestep $t+1$. We assume in this case that normalisation events occur periodically, for instance once every second in simulated time.
One can easily see from the equation above that the weight is scaled up or down, depending on whether the sum of the weights is below or above the target value W_{tot}. Each individual weight w_j of course contributes to that sum. However, there is one problem with this equation: it permits instantaneous change of any size to a single weight, which may be very large if the weight is far removed from its target value after scaling. One could argue that such a large instantaneous change is not very realistic in a biological neuron, where a given synapse cannot undergo arbitrarily large changes in a small timeframe. We therefore make a small modification, adding a parameter η_{SN} that represents the proportion of change from the weight prior to scaling. In other words, for η_{SN} the weight will still move towards its value dictated by complete scaling, but will actually only change by for example 20 percent at every scaling event. By modifying the equation above, we obtain

$$w_{j,t+1} = w_{j,t}\left(1 + \eta_{\text{SN}}\left(\frac{W_{\text{tot}}}{\sum_{k=1}^{N} w_{k,t}} - 1\right)\right). \tag{21}$$

The synaptic normalisation rate η_{SN} can be set from any value between 0 and 1, where 1 results in reinstating W_{tot} precisely at each normalisation, equation 19. For our model neuron, let us use $\eta_{\text{SN}} = 0.2$. For practicality, you can implement the normalisation in the neuron simulation to happen every few seconds, depending on how long you simulate your neuron. Synaptic normalisation is also synapse-type specific, occurring separately for all excitatory and all inhibitory weights. Provide the neuron with excitatory inputs from two groups of each 50 spike trains with spike correlations c=0.1 for group 1 and $c = 0.2$ for group 2, as in the previous section, with STDP, and endow all excitatory synapses with normalisation. Start the weights at 0.1, and use $W_{\text{tot}} = 3$. Also include 30 inhibitory Poisson inputs, with $w_{\text{i}} = 1.0$ and firing rate 10 Hz, but without STDP and normalisation, for simplicity. Normalise the excitatory synapses every second, using equation (21). Run a simulation for 30 seconds. What is happening to the firing rate of the neuron? Is there still competition between the weights of the two input groups? How do the weight distributions of the two groups look at the end of the simulation?

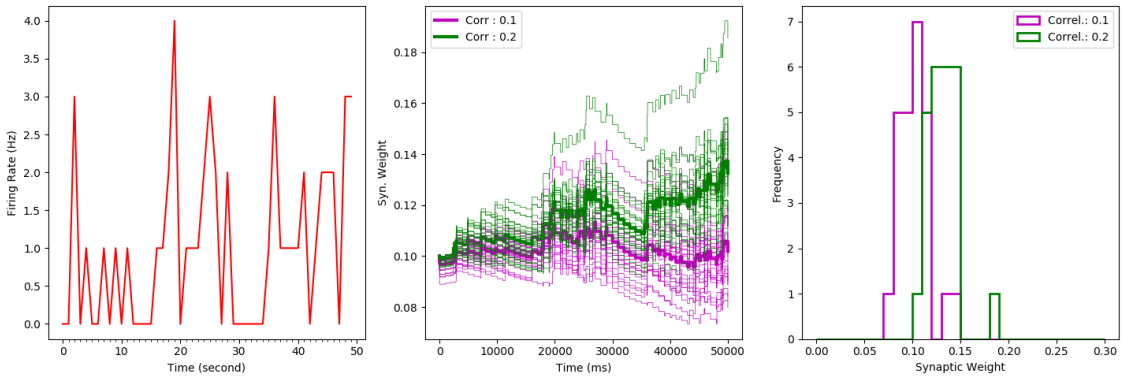

Figure 12: A LIF neuron with two excitatory input groups with STDP and synaptic normalisation in its synapses. The normalisation is applied once every second here. Top left: The LIF neuron is more depolarised over time. Top right: Evolution of the synaptic weights from both input groups. The thin lines show the individual weights, the thick lines the mean weight of the corresponding group. Bottom: The histograms of the weights from each group at the end of the simulation are shown.

4.2 Intrinsic plasticity

Experiments have shown that neurons have multiple strategies for maintaining healthy firing rates. Some types of plasticity can regulate the neuron's output spiking without acting on the input synapses. We call these "Intrinsic Plasticity" (IP) mechanisms [13]. Instead of changing the structure and protein content of the synapses, IP acts upon the ion channels in the neuron membrane, making the neuron less responsive to a single input if the strength of the inputs increases. The IP can therefore implement a homeostatic effect, nudging the spiking threshold of the neuron down or up depending on its recent history of activity. The time scale of IP is typically on the order of hours or even days. In a model neuron, we do not necessarily employ such a long timescale directly, but as with synaptic normalisation, it should be a slower dynamic compared to STDP. In the LIF neuron, IP can be directly implemented with a moving threshold V_{thresh}, and a target firing rate for the neuron, R_{target}. First, create an online spike counter that records the neuron's output spikes, and determine the online firing rate in Hz every second as R_{count}. Set the neuron to change its V_{thresh} by an amount $\eta_{\text{IP}} = 0.1$ mV every second. Reset R_{count} right after every change to V_{thresh}. When the spike counter produces a value that deviates from the target firing rate,

$$V_{\text{thresh},t+1} = V_{\text{thresh},t} + \eta_{\text{IP}}(R_{\text{count},t} - R_{\text{target}}) \tag{22}$$

For instance, V_{thresh} is shifted upwards when the firing rate is too high, and as a consequence the neuron responds less to the same inputs. This is then an opposite effect to for instance, growing synaptic weights from excitatory inputs. Leave out STDP and normalisation in this exercise. Test the IP in a neuron receiving 10 excitatory spike trains with frequency 3 Hz, and a target postsynaptic firing rate of 3 Hz. Set the initial w_{e} to 0.35. Add STDP with $A_{\text{LTP}} = 0.001$ and $A_{\text{LTD}} = =$ -0.0005. Add also 10 inhibitory Poisson inputs of 10 Hz with $w_{\text{i}} = 1.0$. To see the effect best, remove the synaptic normalisation included before. Are the excitatory weights increasing? Does the neuron maintain its firing rate over time? Does the change in firing rate affect the weight evolution?

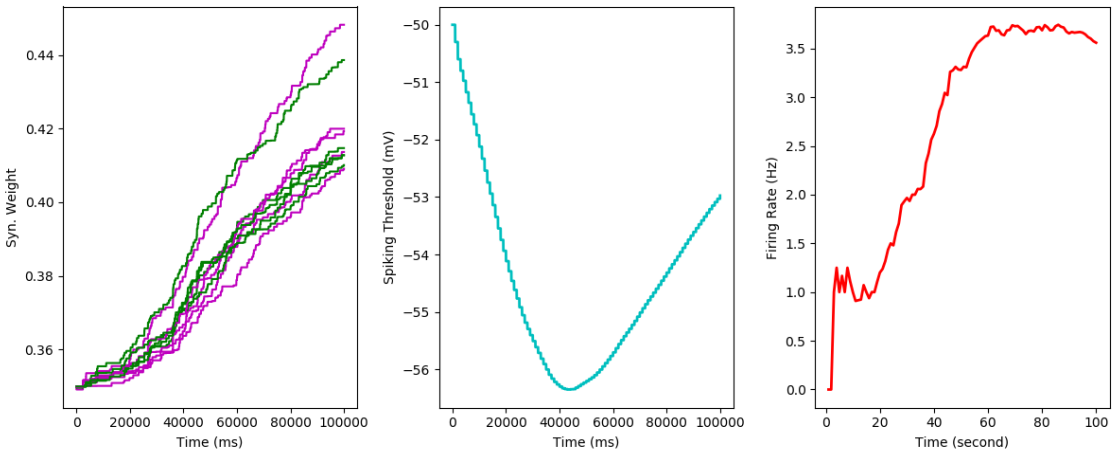

Figure 13: A LIF neuron with an adapable spiking threshold (intrinsic plasticity) and Poisson spiking inputs with STDP. Left: The synaptic weights from the two excitatory input groups increase over time due to STDP. Middle: The spiking threshold first decreases to meet the target firing rate, then increases to compensate for the strong synaptic inputs at the end of the simulation. Right: The output firing rate becomes stable over time.

5 Short-term plasticity (STP)

5.1 Short-term facilitation (STF)

Short-term plasticity (STP) is a transient change in synaptic efficacy caused by the recent activity in the synapse. STP consists of short-term depression (STD) and short-term facilitation (STF). The STD is the result of temporary neurotransmitter depletion, and STF of temporary increase in neurotransmitter release probability due to calcium influx after a spike is generated in the presynaptic neuron. As the name implies, STP does not lead to any lasting memory trace but is nevertheless important for certain dynamics. The best known model of STP is from [14]. It is a simplification of the true processes of STP, but captures the phenomenon qualitatively and is widely used in neuronal and network models. The STF is represented by the variable u:

$$\frac{du}{dt} = -\frac{u}{\tau_f} + U(1-u^-)\delta(t-t_{\text{sp}}) \tag{23}$$

where u is analogous to the fraction of readily available neurotransmitter. The time of the incoming presynaptic spike at the synapse is referred to as t_{sp}. The parameters U and τ_f are constants. At every spike, u is increased by the amount U by the calcium influx in the presynaptic terminal, after which a fraction of u is consumed to produce the postsynaptic current. u^- is the value u just before the spike, and u^+ the value just after the spike. Between spikes u then decays back to 0.

The synaptic efficacy at any time is the product of a fixed maximum weight w_{fixed}, and the facilitation variable u:

$$w_{\text{e}} = w_{\text{fixed}} u^+ \tag{24}$$

Note that the weight in this model does not depend on any post-synaptic behaviour such as whether the postsynaptic neuron spikes or not. Now implement STF in a single excitatory input in the neuron model. Leave out the STDP, normalisation and intrinsic plasticity, so that you can see what the effect of STP will be. Use a Poisson spike train with firing rate 10 Hz. Set $w_{\text{fixed}} = 1.0$, $U = 0.2$, and for τ_f try 50 ms and 750 ms. Plot the change in u, w_{e} and the postsynaptic firing rate over time. What happens to the weight w_{e} and the output firing rate of the neuron?

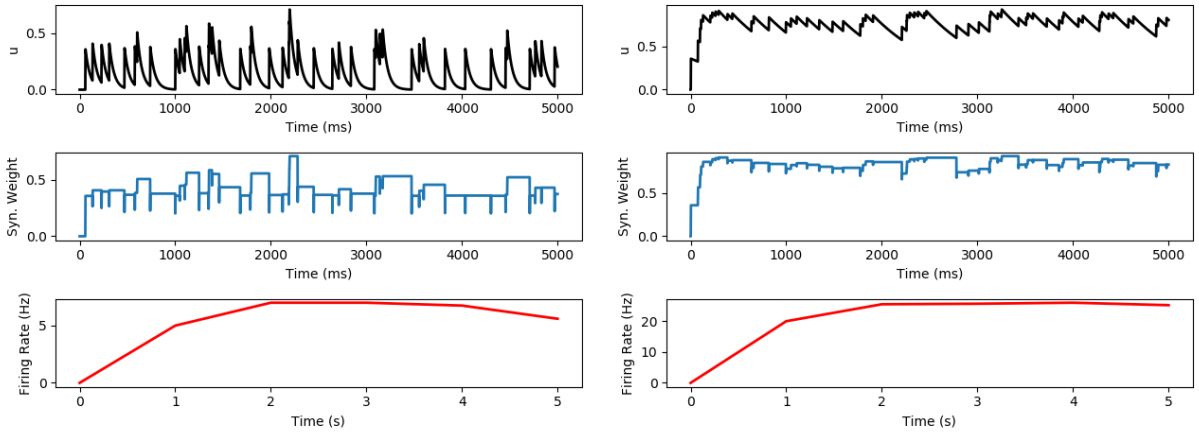

Figure 14: A LIF neuron with excitatory inputs and short-term facilitation in the input synapses, with $\tau_f = 50$ ms (left three figures) and $\tau_f = 750$ ms (right three figures), causing respectively a quickly decaying effect or longer lastig effect of facilitation. Here $U = 0.2$. Top: The fraction of facilitated neurotransmitter in the presynaptic terminal. Middle: The change in the effective synaptic weight. Bottom: The change in output firing rate over time.

5.2 Short-term depression (STD)

Now let us move to STD, which results from temporary depletion of neurotransmitter. The STD is captured by the variable x:

$$\frac{dx}{dt} = \frac{1-x}{\tau_d} - (u^+x^-)\delta(t-t_{\text{sp}}) \tag{25}$$

where x is analogous to the available stock of neurotransmitter. When spiking ceases, x relaxes back to the value of 1, with time constant τ_d. Here, x^- is the value of x just before the spike. If we consider again w_{fixed} to be the maximum efficacy of the synapse, then in a synapse containign both STF and STD, at any moment the synaptic efficacy is the product of u and x:

$$w_{\text{e}} = w_{\text{fixed}}u^+x^- \tag{26}$$

Notice that the product of u and x is bounded by the interval [0,1]. Switch all other types of plasticity off and implement the STF and STD together, with a single excitatory input synapse, connected to a regular (periodic) spike train. Periodic spikes will allow one to see the difference in effect more easily. When both STF and STD are in place, the neuron responds in an input frequency-dependent manner. Which frequency the neuron prefers depends on the ratio between τ_f and τ_d. First set $U = 0.45$, $\tau_f = 50$ ms and $\tau_d = 750$ ms. Use $w_{\text{fixed}} = 2.5$. You now have a synapse that contains both STF and STD, but is mostly dominated by STD. Set the firing rate of the input spike train to 2 Hz to the neuron, and track the variables x, u, w_{e} and the number of output spikes relative to the number of input spikes (ratio of transmitted spikes). Set this ratio in the figure title. Now increase the frequency of the inputs to 20 Hz. How does the neuron respond to low and high frequencies in its inputs? Now switch the values, set $\tau_f = 750$ ms and $\tau_d = 50$ ms, and change U to 0.15. This is now a STF-dominated synapse. Try again the inputs with firing rates 2 Hz and 20 Hz. Does the neuron still prefer lower input firing rates? The key insight here is that the type of STP in the synapse can act as a frequency filter, making the neuron more or less responsive to different input firing rates.

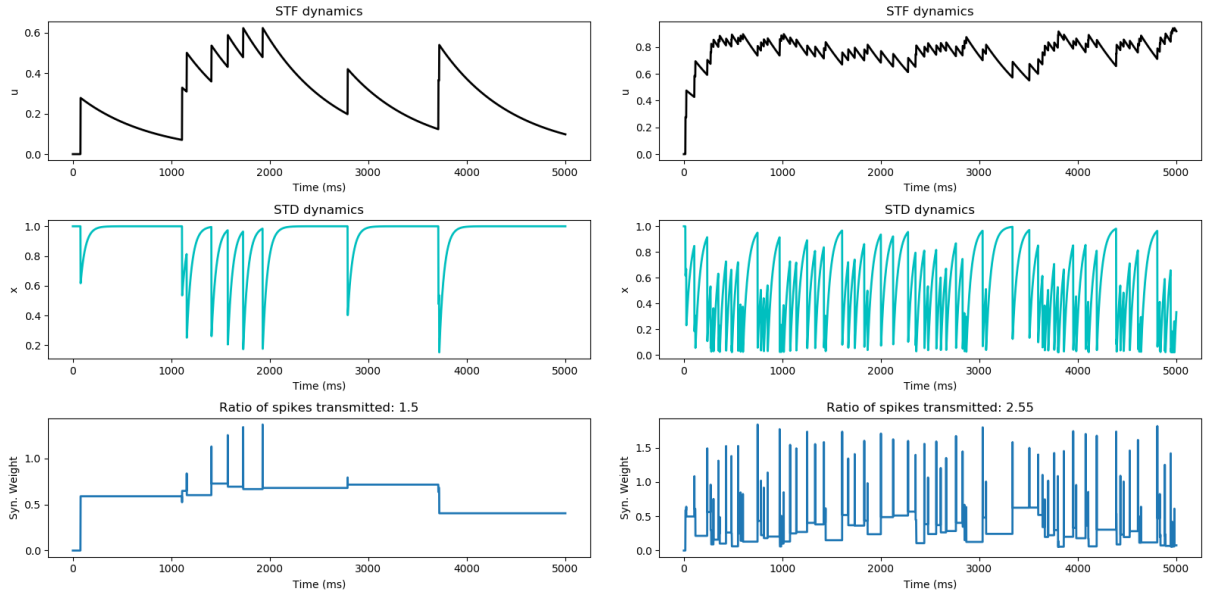

Figure 15: A LIF neuron with excitatory inputs and short-term facilitation in the input synapses. The time constants are $\tau_f = 750$ ms and $\tau_d = 50$ ms, and $U = 0.15$, resulting in facilitation-dominated synapses. The left three figures are for input firing rate 2 Hz, the right three figures are for input firing rate 20 Hz. Here, the ratio of transmitted spikes is higher for the high frequency input. Top: The fraction of facilitated neurotransmitter in the presynaptic terminal. Middle: The fraction of depleted neurotransmitter in the presynaptic terminal. Bottom: The change in the effective synaptic weight.

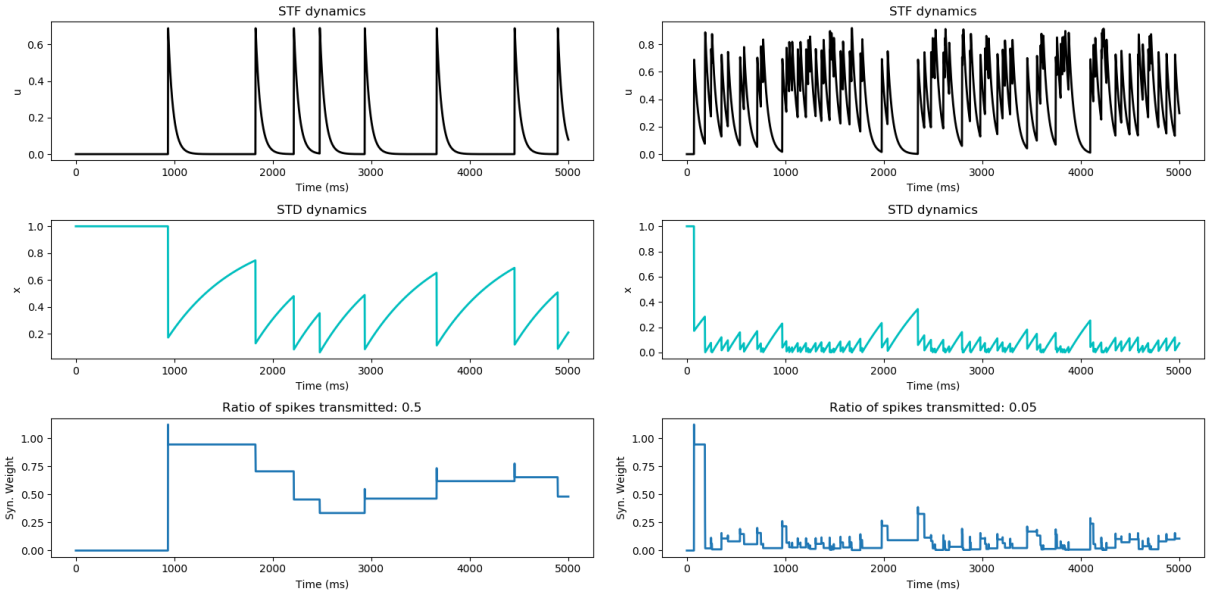

Figure 16: A LIF neuron with excitatory inputs and short-term facilitation in the input synapses. The time constants are $\tau_f = 50$ ms and $\tau_d = 750$ ms, and $U = 0.45$, resulting in depression-dominated synapses. The left three figures are for input firing rate 2 Hz, the right three figures are for input firing rate 20 Hz. Here, the ratio of transmitted spikes is higher for the low frequency input. Top: The fraction of facilitated neurotransmitter in the presynaptic terminal. Middle: The fraction of depleted neurotransmitter in the presynaptic terminal. Bottom: The change in the effective synaptic weight.

6 Epilogue

Congratulations, you have completed the course! We hope you have learned a lot about neural plasticity mechanisms and how we can describe them mathematically and simulate them in a computer. Of course, what you have learned in this course is only a start. The much greater challenge is to understand how these, and other, plasticity mechanisms interact at the network level, allowing our brains to learn how to recognize faces or speak a language, or program neural network simulations, while always staying healthy and efficient. We hope you are as curious about these questions as we are.

Best wishes from Florence Kleberg and Jochen Triesch

References

1. P. Dayan and L. F. Abbott, "Theoretical Neuroscience," *Cambridge, MA; MIT Press*, vol. 806, 2001.

2. M. N. Shadlen and W. T. Newsome, "The variable discharge of cortical neurons: implications for connectivity, computation, and information coding.," *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 18, no. 10, pp. 3870–3896, 1998.

3. L. F. Abbott and S. B. Nelson, "Synaptic Plasticity: Taming the beast," *Nature Neuroscience*, vol. 3, pp. 1178–1183, 2000.

4. G. Q. Bi and M. M. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type.," *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 18, pp. 10464–72, dec 1998.

5. H. Markram, J. Lübke, M. Frotscher, and B. Sakmann, "Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs," *Science*, vol. 275, pp. 213–215, 1997.

6. A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

7. R. Brette, "Generation of correlated spike trains.," *Neural computation*, vol. 21, pp. 188–215, jan 2008.

8. G. G. Turrigiano, K. R. Leslie, N. S. Desai, L. C. Rutherford, and S. B. Nelson, "Activity-dependent scaling of quantal amplitude in neocortical neurons.," *Nature*, vol. 391, no. 6670, pp. 892–896, 1998.

9. T. Keck, G. B. Keller, R. I. Jacobsen, U. T. Eysel, T. Bonhoeffer, and M. Huebener, "Synaptic scaling and homeostatic plasticity in the mouse visual cortex in vivo," *Neuron*, vol. 80, no. 2, pp. 327–334, 2013.

10. J. N. Bourne and K. M. Harris, "Coordination of size and number of excitatory and inhibitory synapses results in a balanced structural plasticity along mature hippocampal CA1 dendrites during LTP," *Hippocampus*, vol. 21, no. 4, pp. 354–373, 2011.

11. F. Zenke, G. Hennequin, and W. Gerstner, "Synaptic Plasticity in Neural Networks Needs Homeostasis with a Fast Rate Detector," *PLoS Computational Biology*, vol. 9, no. 11, 2013.

12. J. Triesch, A. D. Vo, and A. S. Hafner, "Competition for synaptic building blocks shapes synaptic plasticity.," *eLife*, vol. 7, p. e37836, 2018.

13. N. S. Desai, L. C. Rutherford, and G. G. Turrigiano, "Plasticity in the intrinsic excitability of cortical pyramidal neurons.," *Nature neuroscience*, vol. 2, no. 6, pp. 515–520, 1999.

14. M. Tsodyks and H. Markram, "Neural Networks with Dynamic Synapses," *Neural computation*, vol. 10, pp. 821–835, 1998.