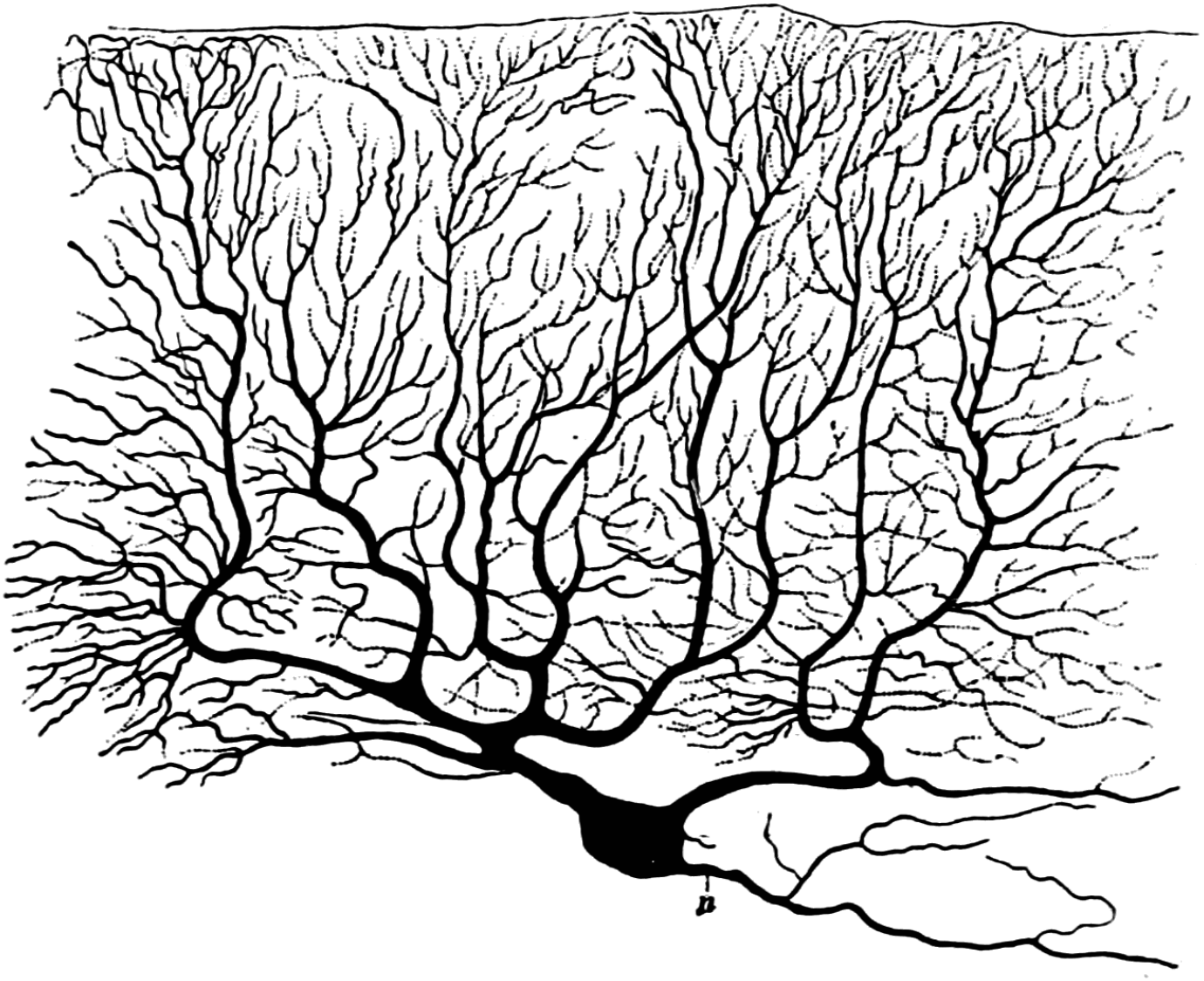


Technische Universiteit Delft

Master Thesis

Encoding of Correlated Temporal Information in a Model Cerebellar Loop with Olivary Oscillators and Long-Term Plasticity

E. M. Fernández Santoro

TU Delft

Master Thesis

Encoding of Correlated Temporal Information in a Model Cerebellar Loop with Olivary Oscillators and Long-Term Plasticity

by

E. M. Fernández Santoro

in partial fulfillment of the requirements for the degree of

Master of Science
in Biomedical Engineering

at the Delft University of Technology,

Student Number: 4293967
Supervisors: dr. ir. M. Negrello, TU Delft & Erasmus MC
dr. ir. A. Schouten, TU Delft
dr. ir. W. Mugge, TU Delft

Declaration of Authorship
With the awareness of my University's regulations concerning plagiarism, including those regulations concerning disciplinary actions that may result from plagiarism, I hereby certify that this work is entirely my own original work except where otherwise indicated. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

TU Delft

To my family and friends

Preface

This Master thesis is carried out in partial fulfillment of the requirements for the degree of Master of Science in Biomedical Engineering at the Delft University of Technology. It is the goal of this student to present a model cerebellar loop with olivary oscillators and long-term plasticity that is encoded with correlated temporal information.

It is assumed throughout this work that the reader has a basic knowledge of human anatomy and is able to understand basic explanations of mathematical models, for this report may be difficult to understand without such knowledge.

This report has been written as two stand-alone parts. Part I: a paper that is the core of this thesis. Part II: an appendix that can be read as a guide in understanding the basic knowledge of spiking neuron models in neuroscience that is needed to fully understand the paper.

Delft, March 10, 2019

I

Encoding of Correlated Temporal Information in a Model Cerebellar Loop with Olivary Oscillators and Long-Term Plasticity

E. M. Fernández Santoro

March 10, 2019

Abstract

The olivocerebellar system plays a central role in motor learning, crucially contributing to the coordination, precision and accurate timing of movements. The system is formed by Purkinje cells (PC), the Deep Cerebellar Nucleus (DCN) and Inferior Olive (IO). Its input activate the PC which produces simple (SS) and complex spikes (CS). The latter induced by the IO. The oscillatory nature of the production of CS seems to play a role in motor control and motor timing. In addition, CS modulate the parallel fiber-Purkinje cell (PF-PC) synaptic plasticity. IO synchrony can help the system learn timing with the PF input as the timing context. Furthermore, the level of synchronization of the coupled IO cells could determine the function of complex spikes, implying that the olivocerebellum is capable of switching between modes of learning by changing the level of synchronization. In this paper we introduced a novel computational model to analyze the role of coupling of the IO and long-term plasticity at the PF-PC synapse in the response of the olivocerebellar loop. It is a resonant system formed by a detailed IO model and Integrate-and-Fire PC and DCN models, with physiologically observed firing frequencies. The IO cells are modeled as coupled oscillators and plasticity is incorporated through a timing dependent specialization of Hebbian learning (Spike-Timing Dependent Plasticity or STDP). Two different simulations are performed both for the coupled and uncoupled scenarios. The STDP is used only in the second type of simulation. Both types of simulations use the same noisy input, which is applied twice during the second type of simulation. The second half of the latter is interpreted as the response of a trained loop. Results show that in the presence of coupling the correlation of the firing rates distribution decreases. This indicates that for the coupled scenario PCs are separating the patterns, while for the uncoupled scenarios the noise is encoded more robustly. Furthermore, a drop in the noise current (inhibiting the PC) leads to an IO spike at about 100 milliseconds later. After training, however, the loop recognizes a drop in the noise and depresses the synapses avoiding an increase in the firing rate of the PC. This effect is more noticeable in the coupled scenario. In conclusion, the model shows that plasticity can lead to learning, and that this process is more efficient for a coupled system.

***Index Terms*—Olivocerebellar System, Purkinje Cell, Olivary Oscillators, Long-Term Plasticity, Complex Spikes, Simple Spikes**

I. Introduction

THE increasing level of understanding of the mechanisms of the human brain has led to numerous technological advances, including compelling pattern recognition programs, learning algorithms and controllers of current widespread use. In particular, these advances have led to impressive progress in the field of robotics which, however, faces at present a fundamental challenge. Indeed, robots are needed with the capacity to learn complex motor behavior the same way humans do, by learning about the body and its environment using a mixture of unsupervised and of goal directed means. The essential step that would make this possible is to develop the equivalent of an artificial cerebellum, as this is the organ playing a central role in motor learning and crucially contributing to the coordination, precision and accurate timing of movements.

The cerebellum is only one component of the olivocerebellar system, which also includes the inferior olive and the cerebellar nucleus. It occupies a central position in the nervous system, having direct communication with the cerebral cortex, midbrain, and spinal cord. Input is carried to the cerebellar cortex through mossy and climbing fibers (CF) arising from precerebellar and inferior olivary nuclei, respectively (Kandel et al., 2000a). These fibers activate Purkinje cells (PC) and produce simple (SS) and complex spikes (CS). With a frequency of up to 100Hz, SS generated by mossy fibers carry sensory information, while the rare CS (1-3Hz), with a powerful influence on the Purkinje cell, is responsible for long-term depression at the PF-PC synapse (Marieb and Hoehn, 2013; Kandel et al., 2000a; Eccles et al., 2013). Purkinje cells are highly organized with each PC receiving one CF. The only output of the olivocerebellar system comes from the deep cerebellar neurons (DCN). In the inferior olivary nucleus (IO), cells oscillate at a low-frequency rhythm (4-10 Hz) (Llinás and Negrello, 2015).

Although the major anatomic features of the cerebellum are well known, the functions the olivocerebellum performs and how it learns to carry out its tasks is fraught with controversy. Researchers do not know what anatomical features allow humans to synchronize their two arms and how different systems are wired together. Despite many discrepancies with physiological results, many neuroscientists have looked at the inferior olive as being responsible of comparing intended with achieved movements (Lang et al., 2017; Marr and Thach, 1991; Albus, 1971; Ito, 1970; Braitenberg, 1983; Tang et al., 2016; Llinás and Negrello, 2015). Cerebellar models such as those proposed by Marr, Albus and Ito assume that the inferior olivary cells act independently while it has been shown experimentally that these cells show high synchronicity (Marr and Thach, 1991; Albus, 1971; Eccles et al., 2013).

Such models have been successfully implemented for limited experiments such as adaptive changes for the vestibulo-ocular reflex and optokinetic eye movement response, or eye blink condition. Some of these models have been successfully employed for robotic applications. Examples include the model by Casellato (Casellato et al., 2014), who used probabilistic complex spikes or the model by Kawato (Kawato and Gomi, 1992).who used a continuous error function. However, it is not clear how an IO could produce the required error computations for complex coordinated tasks. Furthermore, inferior olivary models have concentrated on the oscillation of cells or phase resets and it is not clear how these mechanisms are integrated into the greater picture of motor behavior. Most researchers have concentrated only on the role of the complex spike as a teaching signal while ignoring its oscillatory nature (Llinás and Negrello, 2015). It seems, however, that accurately modeling of the inferior olive is especially important to design brain-like controllers able to learn and to abide by the attributes of a real brain.

There is an ongoing debate about the role of the complex spike that opens an opportunity for further progress (Lang et al., 2017). As more physiological results are found, this ongoing debate is tested and shows fragility. Nevertheless, recent experiments and models that looked at the oscillatory nature of the complex spike suggest that complex spike synchronicity plays a role in the olivocerebellar loop, specifically in motor control and motor timing (Lefler et al., 2013; Llinás and Yarom, 1986; Ankri et al., 2015; Druol et al., 2016; Negrello and Schutter, 2016). Models focusing on the inferior olive and the production of complex spikes are needed to explain why nature found oscillations necessary and what the consequences of these oscillations are in the system. Most control theories have failed to notice or integrate the rhythmicity and synchronicity of climbing fibers arising from the subthreshold oscillations of the inferior olive. The observed relationship between the frequency of olivary oscillations and that of muscular contractions leads to the conjecture that the rhythms related to motor activity are produced by cerebellar activity and that sophisticated motor patterns can be generated by the cerebellar activity that control such rhythms (Braitenberg, 1987). Complex spike synchronicity could play a role in the olivocerebellar loop, specifically in motor control and motor timing. Due to the synchronicity, the inferior olive enables the deep cerebellar nuclei to see differences in magnitude of the signal and thus, to distinguish between simple and complex spikes. Furthermore, strong couplings of the inferior olivary cells induce quick but coarse learning while lower couplings result in precise learning with little contribution to on-line motor control (Sotelo et al., 1974; Schweighofer et al., 1999). These observations imply that the function of complex spikes is determined by the level of synchronization of the coupled olivary cells and, as a consequence, that the cerebellum is capable of switching between modes of learning by changing the level of synchronization. To capture this essential attribute of the cerebellum it is important to introduce models that exploit the oscillatory nature of the production of complex spikes.

In this work, a new cerebellar loop with olivary oscillators and long-term plasticity is proposed to investigate these relationships (Fig. 1). A detailed IO compuational model that is capable of spiking and oscillating is chosen as these are the important features that are investigated. The main novelty is the importance that the conductivity of the IO gap junctions has on the olivocerebellar loop. The coupling of the cells can be changed and the influence on the response of the loop can be analyzed. We model the synapses of this resonant system between the IO PC and DCN according to physiological results. The detailed Inferior Olive model (Schweighofer et al., 1999) is a two compartmental model that has both spiking and oscillatory properties. A two compartmental model is chosen as it is more biophysically accurate. The PC and DCN models are both modified one compartmental adaptive exponential integrate-and-fire models (AdEx) (Brette and Gerstner, 2005) with firing frequencies similar to physiological results (Steuber et al., 2011; Luque et al., 2018; De Schutter and Bower, 1994). The plasticity mechanism used is the Spike-Timing Dependent Plasticity (STDP) that looks at the sum over all pre- and post-synaptic spike times. In this paper learning is interpreted as a change in the response of the system. The analysis focuses on why such a change takes place, without advancing any conjecture as to its meaning.

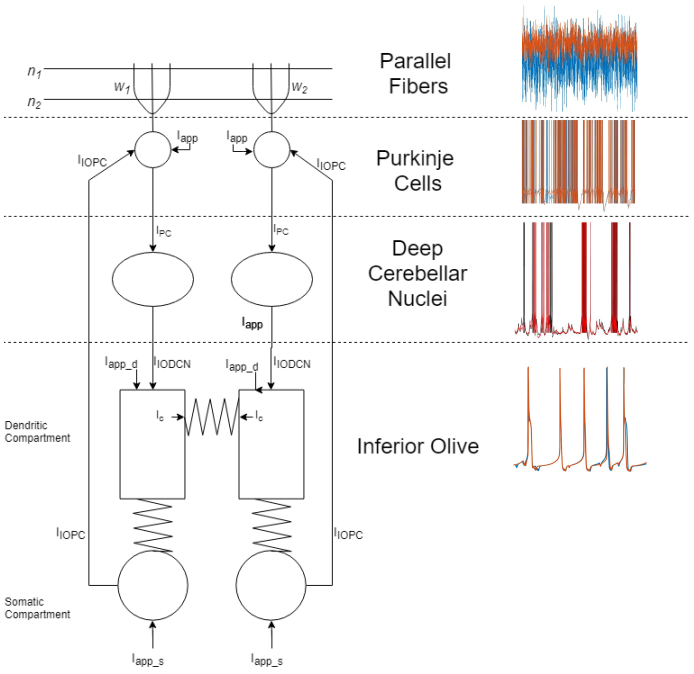


Fig. 1: Model of the cerebellar loop. Input enters the PC through the PF synapse. The PC AdEx (App. B) receives 3 types of inputs: the input from the PF I_{Noise}, an activatory current I_{app} and the input from the IO $I_{IO_{PC}}$ (App. A). The PC projects an inhibitory signal I_{PC} to the DCN (App. C) which in turn, projects a signal to the IO ($I_{IO_{DCN}}$) that resets the oscillations. The 2 compartment IO model receives the $I_{IO_{DCN}}$ and an activatory input current I_{app_d} on the dendritic compartments. This is also the compartment where the IO cells are electrically coupled (I_c). The Somatic compartment has an input current I_{app_s} and projects the $I_{IO_{PC}}$ signal to the PC.

The apriori hypothesis of this present study was that coupling strengthens the effect of plasticity and that the loop with both coupling and plasticity can be more sensitive to the parameters of the input signals. To test this, two different simulations are performed both for the coupled and uncoupled scenarios. The STDP is used only in the second type of simulation. Both types of simulations use the same noisy input, which is applied twice during the second type of simulation. The second half of the latter is interpreted as the response of a trained loop. Results show that in the presence of coupling the correlation of the firing rates distribution decreases. This indicates that for the coupled scenario PCs are separating the patterns, while for the uncoupled scenarios the noise is encoded more robustly. Furthermore, a drop in the noise current (inhibiting the PC) leads to an IO spike at about 100 milliseconds later. In the presence of plasticity, after training the loop recognizes a drop in the noise and depresses the synapses avoiding an increase in the firing rate. This effect is more noticeable in the coupled scenario. For both scenarios the trained loop recognizes specific characteristics of PSC 2 (this is enhanced for the uncoupled scenario). In conclusion, the model shows that plasticity can lead to learning, and that this process is more efficient for a coupled system.

The next section presents the different components of the cerebellar loop. First, the IO model is described. Then, the two Adaptive Exponential Integrate-and-Fire Models of the PC and the DCN are shown. This is followed by a description of the input noise current. The 5 different synapses are explained. Namely, the Noise-PC synapse with and without STDP, the IO-PC synapse, the PC-DCN synapse, and the DCN-IO synapse.

II. Methods

A. Inferior Olive Model

The IO model is a modification of the Schweighofer model (Schweighofer et al., 1999) as it includes a new current $I_{IO_{DCN}}$ that models the influence of the DCN on the IO (further explanation of the original model is found in App. A). The general cell equations for the model are shown below and in Tab. IX. Furthermore, the parameters used for the experiments in this model are shown in Tab. XII.

$$C_m \frac{dV_s}{dt} = (-\sum I_{Somatic} + I_{app_s})$$
$$C_m \frac{dV_d}{dt} = (-\sum I_{Dendritic} + I_{app_d} + I_{IO_{DCN}})$$
$$\frac{dI_{IO_{DCN}}}{dt} = \frac{(I_0 - I_{IO_{DCN}})}{\tau_I}$$

The dynamics of each activation or inactivation variable is given by the differential equation:

$$\frac{dx}{dt} = \frac{(x_\infty - x)}{\tau_x}$$

where x indicates the variables h, k, l, n, q, r and s. Furthermore, the dynamics of the calcium concentration $[Ca^{2+}]$ and the transjunctional voltage dependence of the gap junction conductance f are shown below.

$$\frac{d[Ca^{2+}]}{dt} = -3.0 I_{Ca_h} - 0.075[Ca^{2+}]$$

$$f(V) = 0.6e^{-\frac{V^2}{50^2}} + 0.4$$

The current flowing out into the dendritic (I_{ds}) and into the somatic compartments (I_{sd}), as well as the leakage currents for the soma (I_{ls}) and for the dendrites (I_{ld}) are represented in this model (Schweighofer et al., 1999). As the window of conductance of the calcium low-threshold current (I_{Ca_l}) is

TABLE I: Equations of the Ionic Conductances of the IO Cell

Somatic Currents	Dendritic Currents
$I_{Ca_l} = g_{Ca_l} k^3 l (V_s - V_{Ca})$	$I_{Ca_h} = g_{Ca_h} r^2 (V_d - V_{Ca})$
$I_h = g_h q (V_s - V_h)$	$I_{K_{Ca}} = g_{K_{Ca}} s (V_d - V_k)$
$I_{Na} = g_{Na} m_\infty^3 h (V_s - V_{Na})$	$I_{ld} = g_{ld} (V_d - V_l)$
$I_{K_{dr}} = g_{K_{dr}} n^4 (V_s - V_k)$	$I_{sd} = \left[\frac{g_{int}}{(1-p)}\right] (V_d - V_s)$
$I_{ds} = \left(\frac{g_{int}}{p}\right) (V_s - V_d)$	$I_c = g_c f(V_d - V_{de})(V_d - V_{de})$
$I_{ls} = g_{ls} (V_s - V_l)$	

TABLE II: Parameters of the IO Standard Cell

Conductances (mS/cm^2)		Reversal Potentials (mV)	
g_{Na}	80	V_{Na}	55
$g_{K_{dr}}$	2	V_K	-75
g_{Ca_l}	1	V_{Ca}	120
g_h	0.2	V_h	-43
g_{Ca_h}	4.0	V_l	10
$g_{K_{Ca}}$	35	Membrane Capacitance ($\mu F/cm^2$)	
g_{ls},g_{ld}	0.016		
g_c	0.125	C_m	1
Cell Morphology		IO-DCN Synapse	
g_{int}	0.13	I_0	0 $\mu A/cm^2$
p	0.20	τ_I	9ms

TABLE III: Steady-state Activation/Inactivation and Time Constants of the Ionic Conductances of the IO Cell

Conductance Type	Steady-State Activation/Inactivation	Time Constant (ms)	Forward Rate Function α	Backward Rate Function β
I_{Na}	$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$		$\alpha_m = \frac{0.1(V+41)}{1 - e^{\frac{-(V+41)}{10}}}$	$\beta_m = 9.0e^{\frac{-(V+66)}{20}}$
	$h_\infty = \frac{\alpha_h}{\alpha_h + \beta_h}$	$\tau_h = \frac{170}{\alpha_h + \beta_h}$	$\alpha_h = 5.0e^{\frac{-(V+60)}{15}}$	$\beta_h = \frac{(V+50)}{1 - e^{\frac{-(V+50)}{10}}}$
$I_{K_{dr}}$	$n_\infty = \frac{\alpha_n}{\alpha_n + \beta_n}$	$\tau_n = \frac{5}{\alpha_n + \beta_n}$	$\alpha_n = \frac{(V+41)}{1 - e^{\frac{-(V+41)}{10}}}$	$\beta_n = 12.5e^{\frac{-(V+51)}{80}}$
I_{Ca_l}	$k_\infty = \frac{1}{1 + e^{\frac{-(V+61}{4.2}}}$	$\tau_k = 5$		
	$l_\infty = \frac{1}{1 + e^{\frac{(V+85.5)}{8.5}}}$	$\tau_l = \frac{20e^{\frac{(V+160)}{30}}}{1 + e^{\frac{(V+84)}{7.3}}} + 35$		
I_h	$q_\infty = \frac{1}{1 + e^{\frac{(V+75)}{5.5}}}$	$\tau_q = \frac{1}{e^{(-0.086V - 14.6)} + e^{(0.07V - 1.87)}}$		
I_{Ca_h}	$r_\infty = \frac{\alpha_r}{\alpha_r + \beta_r}$	$\tau_r = \frac{1}{\alpha_r + \beta_r}$	$\alpha_r = \frac{1.6}{1 + e^{\frac{-(V-5)}{14}}}$	$\beta_r = \frac{0.02(V+8.5)}{1 - e^{\frac{(V+8.5)}{5}}}$
$I_{K_{Ca}}$	$s_\infty = \frac{\alpha_s}{\alpha_s + \beta_s}$	$\tau_s = \frac{1}{\alpha_s + \beta_s}$	$\alpha_s = \min\left(2 \cdot 10^{-5}[Ca^{2+}], 0.01\right)$	$\beta_s = 0.015$

around the resting membrane potential, the IO cell is excited in response to hyperpolarizing currents (Llinás and Yarom, 1981). On the other hand, the calcium high-threshold current (I_{Ca_h}) is noninactivating. Thus, a depolarizing dendritic input results in a prolonged plateau potential. As calcium enters the dendrites, the calcium-activated potassium current ($I_{K_{Ca}}$) is activated. This current abruptly terminates the plateau potential after about 30 milliseconds (Llinás and Yarom, 1981). Due to the long time constant (several hundred milliseconds) of the $I_{K_{Ca}}$, there is a long afterhyperpolarization which, in turn, deinactivates the I_{Ca_l} resulting in a postinhibitory rebound spike. The afterhyperpolarization also activates an anomalous rectifying current I_h as it is a current that activates at hyperpolarized potentials. This current contributes to the subthreshold oscillations as it contributes to amplitude and frequency (Schweighofer et al., 1999; Bal and McCormick, 1997). Furthermore, similarly to Hodgkin-Huxley type neuron models, somatic sodium spikes are generated with the sodium current I_{Na}. These are terminated by an outward delayed rectifier potassium current $I_{K_{dr}}$ (Llinás and Yarom, 1981). Finally, the electronic coupling between the cells is represented by a current, I_c.

B. Other Cerebellar Neuron Models

The other cerebellar neurons (Purkinje Cell and Deep Cerebellar Nuclei) are simulated as adaptive exponential integrate-and-fire models (Brette and Gerstner, 2005). The general cell equations for the PC and DCN are described below.

$$\frac{dV}{dt} = \frac{1}{C_m}\left(g_L\left(E_L - V\right) + g_L\Delta_T e^{\left(\frac{V - V_T}{\Delta_T}\right)} + \sum_i I_i - w\right)$$

$$\frac{dw}{dt} = \frac{a\left(V - E_L\right) - w}{\tau_w}$$

Here, Δ_T is the slope factor, V_T is the the threshold potential, w is the adaptation variable, a is the adaptation coupling factor and τ_w is the adaptation time constant. $\sum_i I_i$ is the sum of the currents specific to the PC and the DCN (where *i* = PC, DCN). When a spike happens both the membrane potential and the adaptive variables are reset, however, while the first resets to the resting potential, the second decays until either reaching 0 or the next spike time (Dayan and Abbott, 2005). The slope factor Δ_T is a quantification of the sharpness of the spike. This can be seen as the sharpness of the sodium activation curve if the activation time constant is ignored (Goodman and Brette, 2008).

1) Purkinje Cell AdEx: For the Purkinje cell, the currents are I_{Noise} and I_{app} (further explanation of this model is found in App. B). The former is the input from the parallel fibers and the latter is an activation input current that may be given to the cell.

TABLE IV: Parameters of the AdEx Cell Types

Parameters	PC	DCN
C	75 pF	281 pF
g_L	30 nS	30 nS
E_L	-70.6 mV	-70.6 mV
V_T	-50.4 mV	-50.4 mV
Δ_T	2 mV	2 mV
V_{cut}	$V_T + 5\Delta_T$	$V_T + 5\Delta_T$
τ_w	144 ms	30 ms
a	4 nS	4 nS
b	0.0805 nA	0.0805 nA
V_r	-70.6 mV	-65 mV
τ_I		30 ms
$I_{PC_{max}}$		0 nA
$I_{PC_{Intrinsic}}$		2 nA

$$\sum_i I_{PC} = I_{Noise} + I_{app}$$

2) Deep Cerebellar Nuclei AdEx: In the case of the DCN AdEx, I_{PC} is the inhibitory projection from the PC (further explanation of this model is found in App. C). $I_{Intrinsic}$ defines the intrinsic firing of the cell that is inhibited by the PC when it spikes.

$$\sum_i I_{DCN} = I_{Intrinsic} - I_{PC}$$

When the PC is not spiking, $I_{PC} = 0nA$, however, upon a spike of the PC, I_{PC} is updated to a certain value (See II-D). This means that the total current $I_{Intrinsic} - I_{PC}$ will be smaller and the DCN will spike less (inhibition from PC).

$$\frac{dI_{PC}}{dt} = \frac{(I_{PC_{max}} - I_{PC})}{\tau_I}$$

C. Input Noise (Parallel Fiber Input)

For the parallel fiber input, a current with mean I_0 and Ornstein-Uhlenbeck noise is used. ξ is a Gaussian random variable with mean 0 and standard deviation 1 that scales with units of $seconds^{-0.5}$. The parameters of each noise source are randomized based on the number of sources (N_{Noise}). This means that each noise source will have a different I_0 and σ. If the noise is negative it is looked at as being the influence of inhibitory interneurons on the loop.

$$\frac{dI}{dt} = \frac{I_0 - I}{\tau_{noise}} + \frac{\sigma\xi}{\sqrt{\tau_{Noise}}}$$

TABLE V: Noise Parameters

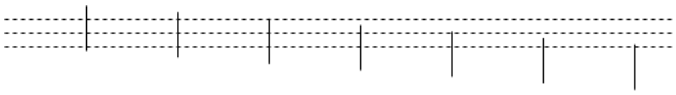
Parameters	Values
τ	50 ms
I_0	1.5± 0.1N_{Noise} nA
σ	0.5± 0.1N_{Noise} nA

D. Synapses Between Neurons

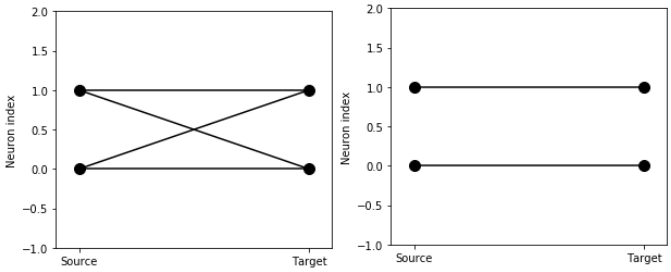
1) Noise-Purkinje Cell Synapse without STDP: The PC-PF synapses are not linear in the longitudinal direction (Fig. 2a). This phenomenon is modelled by synaptic weights that decrease with the distance between the source and the target neurons (Fig. 2b). The weight for the synapses from the i^{th} presynaptic neuron (source) to the j^{th}postsynaptic neuron (target) is computed as follows.

$$w_{syn_{i,j}} = 1 - \frac{|i-j|}{N_P C}$$

As j increases (gets further away), w_{syn} decreases, lowering the effect this presynaptic neuron has on the postsynaptic one. For example, for PC_1 its synaptic weights $w_{syn_{1,1}} = 1$ and $w_{syn_{1,2}} = 0.5$.



(a) Top View of PC (solid lines) and PF (dashed lines). The PC are not ordered in a linear way in the longidtudinal direction, hence, the PF do not make synapses with all the PC in one strip of PC



(b) Distance Based Connectivity, each source neuron is connected to all target neurons

(c) Direct Connectivity, each source is connected to one target neuron

Fig. 2: Synaptic Connectivity

2) Noise-Purkinje Cell Synapse with STDP: When introducing plasticity to the model, the connectivity of the PF-PC synapse is the same but the synaptic weights are changed. For spike-timing dependent plasticity, the change in synaptic weight $w_{syn_{i,j}}$ is the sum over all pre- and post-synaptice spike times, t_{pre} and t_{post}. Two new variables are defined: a_{SS} and a_{CS} which are the traces of the pre- and post-synaptic activity. These are governed by the following differential equations:

$$\tau_{pre}\frac{da_{SS}}{dt} = -a_{SS} \qquad \tau_{post}\frac{da_{CS}}{dt} = -a_{CS}$$

The traces are updated every time their respective neuron spikes. In the case of the PC, for each SS the trace (a_{SS}) is updated with a constant value A_{SS}. For the IO, a_{CS} depends of the amplitude of the noise (with respect to I_0) and thus, is updated by $A_{CS} \cdot f(amplitude)$. Both A_{SS} and A_{CS} depend on the length of the experiment. Nevertheless, the synaptic weight is not updated in the same fashion as the traces. The weight is updated with the trace of the PC (a_{SS}) when there is a CS. The weight is then updated with the trace of the IO (a_{CS}) when the next SS occurs. In this way, the number of SS before the CS has an influence on the change in the synaptic weight.

More precisely, when a CS occurs $a_{CS} = a_{CS} + A_{CS}$ and $w_{syn_{i,j}} = w_{syn_{i,j}} + a_{SS}$. When a SS occurs $a_{SS} = a_{SS} + A_{SS}$ and $w_{syn_{i,j}} = w_{syn_{i,j}} + a_{CS}$. Notice that only the first SS leads to an update, as a_{CS} remains unchanged until the next CS. This update of the weight from the SS can be thought of as a potentiation of the synapse while the CS trace leads to a depression of the synapse.

TABLE VI: STDP Parameters

Parameters	Values
τ_{pre}	20 ms
τ_{post}	20 ms
A_{SS}	0.01/(runtime)
A_{CS}	0.01/(runtime)

3) Inferior Olive - Purkinje Cell Synapse: The modeling of the IO-PC synapse focuses on the pause mechanism following a complex spike. Each PC is connected to one IO as seen in Fig. 2c (for bigger neuronal populations each IO can connect to up to 10 different PC). As the PC AdEX model already has an adaptation variable that increases the interspike interval (ISI), the synaptic modeling only requires the increase of the adaptation variable w following a presynaptic spike (IO spike). As w is increased to a higher value, the PC cannot spike until w decays to a lower value.

4) Purkinje Cell - Deep Cerebellar Nuclei Synapse: The DCN is always spiking but is inhibited by a spike of the PC. When I_{PC} is large, the difference $I_{Intrinsic} - I_{PC}$ becomes smaller and the DCN is inhibited. When there is a presynaptic spike (PC), I_{PC} is increased and until it decays back to $I_{PC_{max}}$ the DCN cannot fire.

5) Deep Cerebellar Nuclei - Inferior Olive Synapse: In analogy to the PC-DCN synapse, $I_{IO_{DCN}}$ increases to a high current value when the DCN spikes. As the decay time of $I_{IO_{DCN}}$ is small (9 milliseconds) this can be seen as a large current pulse entering the IO cell resulting in a reset of the oscillations.

TABLE VII: Summary of the Synaptic Configurations

Synapse (Source-Target)	On Presynaptic Spike	Connection	Synaptic Weight
PF-PC		Distance Based	$1 - \frac{\lvert i-j \rvert}{N_{PC}}$
IO-PC	$w = 0.9$nA	Direct	1
PC-DCN	$I_{PC} = 0.9$nA	Direct	1
DCN-IO	$I_{IO_{DCN}} = 9\mu$A/cm^2	Direct	1

E. Analysis

The main steps of the present analysis are the following:

1) Description of the behavior of individual cells. This allows the study of their mutual influence and the role of the loop.
2) Study of differences between the responses for the coupled and uncoupled scenarios without STDP.
   2.1) Describe the best PC stimulus for an IO spike.
3) Study of differences between the responses for the cound uncoupled scenarios with STDP .
4) Comparison of the different scenarios with and without the presence of plasticity.
   4.1) Explain how the firing rates of each cell change after STDP.
   4.2) Study how the input correlate with the PC output after STDP.

Two different simulations are performed both for the coupled and uncoupled scenarios. The STDP is used only in the second type of simulation. Both types of simulations use the same noisy input, which is applied twice during the second type of simulation. The second half of the latter is interpreted as the response of a trained loop.The PC, DCN and IO firing rates are compared for each simulation. For the IO, the spike amplitudes and periods are also analyzed. The behavior of the model is compared graphically and through statistical tests, including t and F tests. The study of the simulation outcomes involved processing almost 300 plots[1], from which only the more relevant are shown in this paper. These include raster plots, firing rate plots, CS triggered noise averages plots, and CS triggered PC firing rate averages. For the raster plots, a CS is taken as an event. For each event at time t_i, the PC response is registered for the time interval (t_i-600ms,t_i+600ms). All of the SS and the IO spikes found in that snippet of time are plotted as a function of time, placing the CS event at time zero. For the plot of the CS triggered noise averages the snippets of time are computed in the same fashion, but averaging over all the events of both the IO response and the noise. The same is done for the averages of PC firing rates.

[1]These can be found in the repository of TU Delft, otherwise contact the author

III. Results

A. Response of Individual Cells to Random Noise

The postsynaptic current (PSC) of each PC is shown in Fig. 3. Fig. 5 show the responses of the PC, DCN and IO during a 2 second window of the experiment. The mechanisms for the cerebellar loop (as explained in Sec. II) are working as expected as CS create PC pauses which induces a DCN rebound spike which in turn, influences the IO.

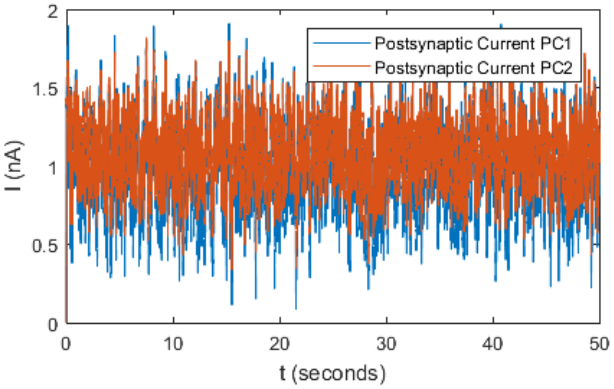


Fig. 3: The noise current entering each PC (postsynaptic current noise or PSC) for a duration of 50 seconds. Each postsynaptic current is a combination of the noise sources (presynaptic) as explained in Sec. II-D. PSC of PC1 has both a higher variance and lower intensity than PSC of PC2 ($I_{0_1} = 1$nA, $\sigma_1 = 0.67$ and $I_{0_2} = 1.33$nA, $\sigma_2 = 0.33$).

Fig. 5 shows the behavior of the three cells:

- An IO spike (CS) is preceded by a drop followed by an increase in the rate of firing of the PC.
- The IO spike is followed by a pause in the PC activity, which gives rise to a burst of the DCN spikes.
- Both DCN cells seem to have similar membrane potential responses. Nevertheless, the small differences in their response are, in reality, significant when looking at the IO response.

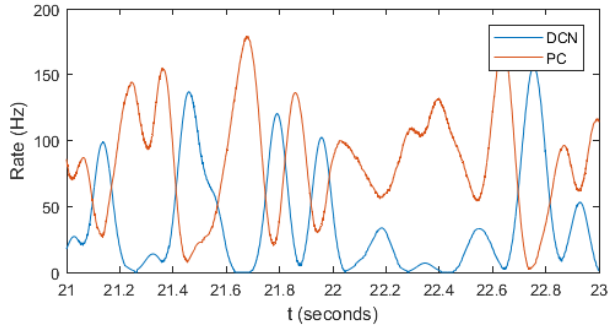


Fig. 4: Population firing rates (both cells) of PC and DCN. The PC inhibits the DCN, this leads to an inverse relationship between the firing rates. As the PC population decreases its activity, the DCN population spikes more.

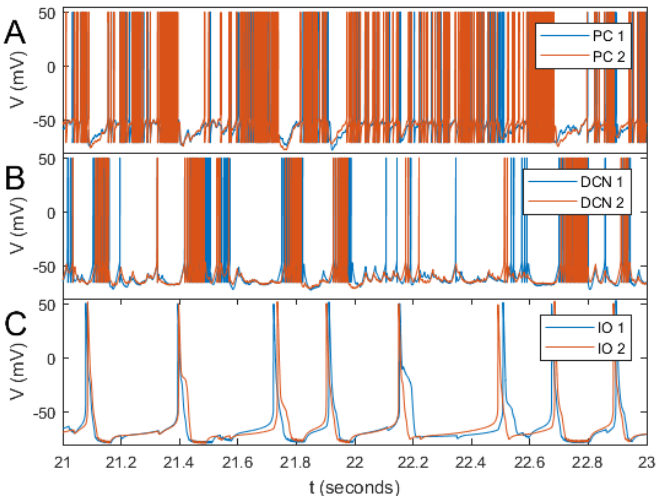


Fig. 5: The response of each cell of the cerebellar loop for a small window of time (2 seconds). A: the PCs response, B: the DCN response and C: the IO response of coupled cells.

We further observe the following:

- In consistency with physiological observations, Fig. 5.A shows that PCs fire extremely frequently (at about 100Hz) (Marieb and Hoehn, 2013; Kandel et al., 2000b).
- With a frequency of about 40Hz, the DCN is fully inhibited by the PC and fires when there is a drop in the PC's activity (cf. Fig. 5.A and 5.B). This firing frequency is also in accordance to physiological results ((Steuber, 2016)). The relationship between DCN firing rate and PC firing rate is shown in Fig. 4.
- The DCN has a bursting response following a pause in the PC due to a CS (see Fig. 5 at 21.1, 21.4, 21.7, 21.9, 22.1, 22.5, 22.7 and 22.9 seconds).
- The IO has a tonic spiking response with a frequency of about 2Hz (Fig. 5).
- The DCN spiking has an effect on the IO:
  - At a time of about 23.1 seconds we observe the rebound spike of DCN 1.
  - As a result of the DCN rebound, the response of IO 1 is modified when compared to IO 2 This is shown in Fig. 5.B, where the membrane potential of IO 1 and IO 2 become different due to the DCN spiking. This also seen at 22.1 seconds.
  - The influence is better seen following the IO spike at 21.7 seconds. The two DCN spike differently and the effect of a longer burst is seen in the response of IO 1 at that time.
- The effect of the IO on the PC is seen at 23.4 seconds. There is a spike from IO 2 but IO 1 does not spike (see Fig. 5). It is seen that only PC 2 is pausing while PC 1 is spiking.
- The PC pause at 23.6 seconds in Fig. 5.A is particularly interesting when thinking of the STDP mechanism. While there is a CS for PC 2 relatively close to the one at this time, PC 1 did not pause and kept on firing. This difference can be seen as the membrane potential of PC 1 decreases much more than the one of PC 2.

B. Response of the Cerebellar Loop Without Plasticity

The main result is that coupling increases both the firing rate and the period variability of the IO. This is because each IO cell is influenced by the other. As the current from the coupling I_c entering each cell increases due to the firing of the other cell, more IO spikes happen. Moreover, these IO spikes happen at arbitrary points in time, thus increasing the variability in the period.

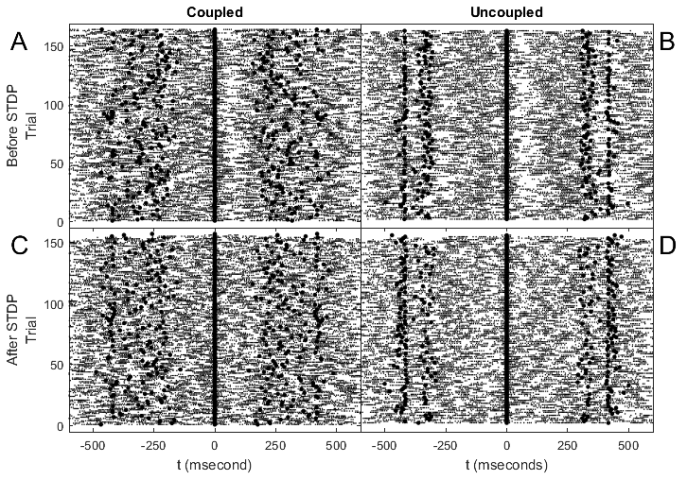


Fig. 6: Raster Plots of IO cell 1 for each scenario without STDP (A and B) and with STDP (C and D). Fig. A and C are the coupled scenarios, and B and D are the uncoupled scenarios. The uncoupled scenario has a lower variability in the period. There is no change after STDP.

More precisely, we observe the following:

1) Period Variability
   - The raster plots (Fig. 6) show that there is a higher variability in the period of CS when the cells are coupled than when uncoupled.
   - The average period of the IO spikes of the coupled cells is 298.06 ms for cell 1 and 294.45 ms for cell 2 (with standard deviations of 81.93 ms and 80.40 ms).
   - In the case of the uncoupled cells, the mean period for cell 1 is 369.53 ms with a standard deviation of 47.80 ms and 354.13 ms with standard deviation of 40.38 ms for cell 2.
   - The average period of IO spikes is significantly different between coupled and uncoupled cells (*p=5.9855e-35* for cells 1 and *p=9.3597e-29* for cells 2).
   - Fig. 7 shows that the variability in the period for the coupled scenario is higher than for the uncoupled.
2) Firing rate
   - When looking at the firing rates of the IO cells, it is seen that coupled cells have a higher firing rate than uncoupled cells (Fig. 8).
   - The average IO firing rate for the coupled scenario is 1.21 Hz.
   - The average IO firing rate for the uncoupled scenario is 0.78 Hz.
3) Amplitude of the IO spikes
   - Both cells have similar amplitude averages of the IO spikes.
   - The average amplitude of the IO spikes for coupled cells 1 and 2 is of 50.17 mV and 50.69 mV with standard deviations 5.60 mV and 0.90 mV, respectively.
   - For the uncoupled cells, IO cell 1 has an amplitude average of 48.56 mV with a standard deviation of 9.39 mV and IO cell 2 has 50.18 mV amplitude average with 0.46 mV standard deviation.

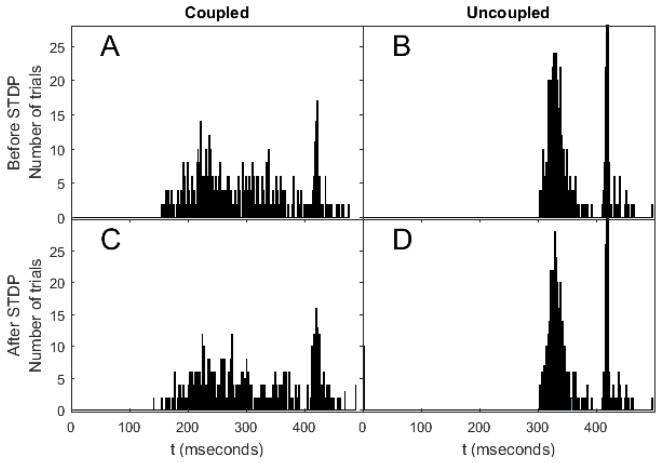


Fig. 7: Histograms of Periods of IO population (both cells) for each scenario before STDP (A and B) and after STDP (C and D). Fig. A and C are the coupled scenarios, and B and D are the uncoupled scenarios. The uncoupled scenario has a lower variability in the period. There is no change after STDP.

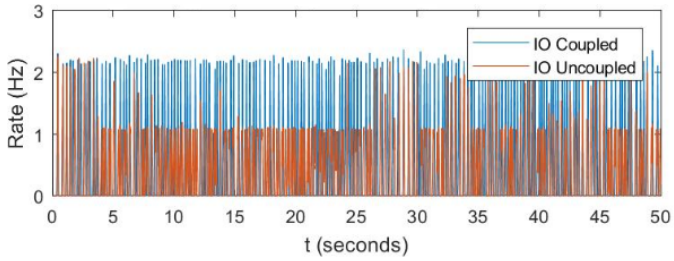


Fig. 8: IO Firing rates between coupled and uncoupled scenarios before STDP. The firing rate for the uncoupled scenario is lower than for the coupled one.

Further observations are found for the DCN and PC:

1) PC response:
   - The mean PC population firing rate is of 91.77 Hz for coupled cells and 96.42 Hz for the uncoupled scenario.
   - According to physiological results, the PC fires regularly whether it is in the coupled or uncoupled scenario (*F=0.4382* for coupled and *F=0.3620* for uncoupled).
   - The results are consistent: uncoupled IO cells have a lower population firing rate and the PC for that scenario have a higher firing rate. This shows that there are less pauses happening as there are less CS.

2) DCN response:
- The mean DCN population firing rate is of 33.43 Hz for coupled and 29.22 Hz for uncoupled scenarios.
- As there is a higher PC firing rate in the uncoupled scenario, it follows that the DCN firing rate for that scenario will be smaller. This relationship is shown in Fig. 4.

C. What is the Preferred PC Stimulus for an IO Spike?

We find in Fig. 9 that for both coupled and uncoupled scenarios a drop in the noise current (inhibiting the PC) leads to an IO spike at about 100 milliseconds later. The noise decreases and then increases leading to an increase in the PC firing rate (see Fig. 10). These figures show that an increase in the PC firing rate about 50 ms before the IO spike is ideal to find a CS. Following the IO spike, the average PC firing rate decreases indicating a PC pause. Furthermore, the figures also show that the variability in the IO spikes for the uncoupled scenario affects the PC firing rates. We see in Fig. 10 that this small variability is translated to a higher overlap of the different IO spikes. In turn, the average of the membrane potential is higher and the PC firing rate average is decreased. This is shown for the firing rate of PC 2 in the uncoupled scenario.

D. What are the Main Differences After Training Between Coupled and Uncoupled?

1) IO Period Variability
- The average period of the IO spikes of the coupled scenario is 312.99 ms for cell 1 and 311.57 ms for cell 2 (with standard deviations of 88.23 ms and 83.13 ms, respectively).
- For the uncoupled scenario, the mean period for cell 1 is 391.13 ms with a standard deviation of 52.38 ms and 372.75 ms with standard deviation of 48.13 ms for cell 2.
- The average period of IO spikes is significantly different between coupled and uncoupled cells (p=*1.1148e-14* for cells 1 and p=*1.2205e-25* for cells 2).
- There is a significant difference in variability of the periods between both scenarios (p=*0.2921* for coupled cells and p=*0.1753* for uncoupled cells).
- There is not a significant difference in the variability between before and after STDP for each scenario (p= *0.1849* for cells 1 and p=*0.1421* for cells 2).

2) IO Firing rate
- The firing rates of the IO cells is higher for the coupled scenario than for the uncoupled cell one (1.82 Hz and 0.83 Hz respectively).
- The firing rates of the IO are higher after plasticity (p = *2.7495e-115* for coupled, p = *0* for uncoupled).

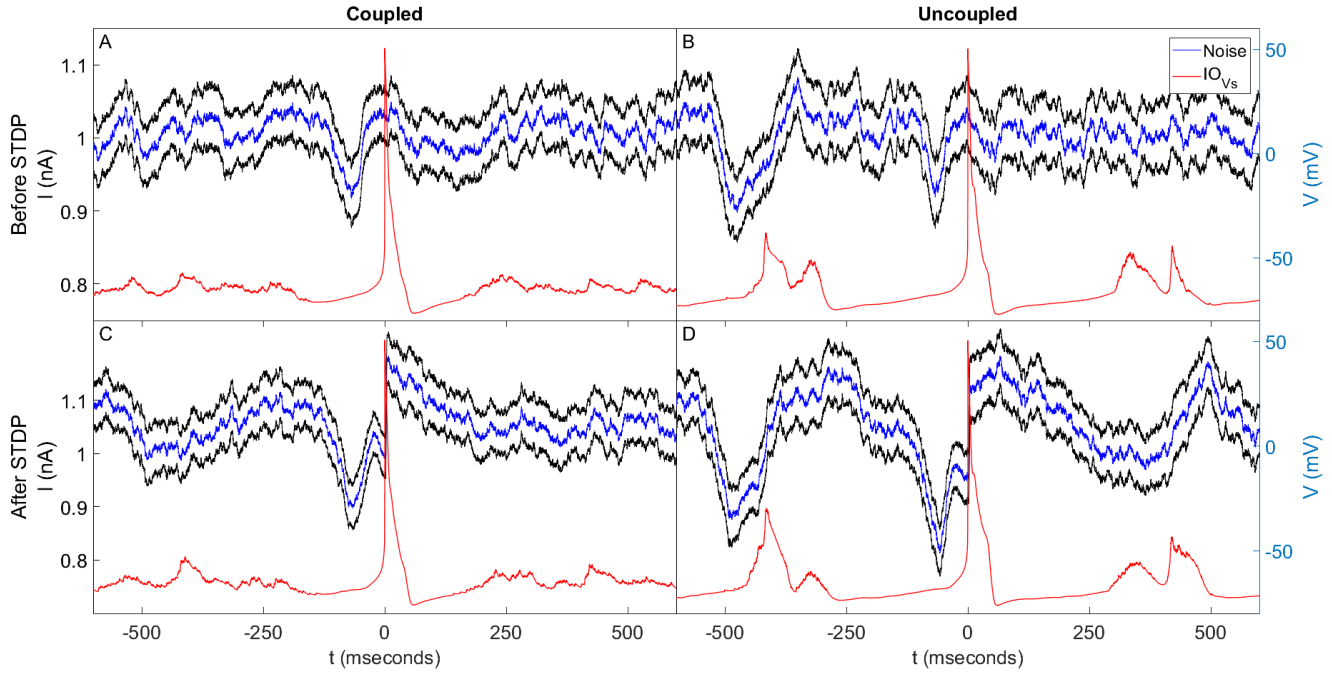


Fig. 9: CS Triggered Noise Averages for coupled (A and C) and uncoupled (B and D) scenarios for both before (A and B) and after (C and D) STDP. The left vertical axis shows the current of the PSC of PC 1 and the right axis the membrane potential of IO 1. The upper and lower black lines represent the confidence interval of the average PSC. A lower variability in CS period is seen for the uncoupled scenario (B and D) as the average shows membrane potential peaks before and after the CS. The coupled scenario (A and C) is seen to have less variability in the PSC preceeding the drop at 100ms. This relates to the higher variability in CS period for this scenario.

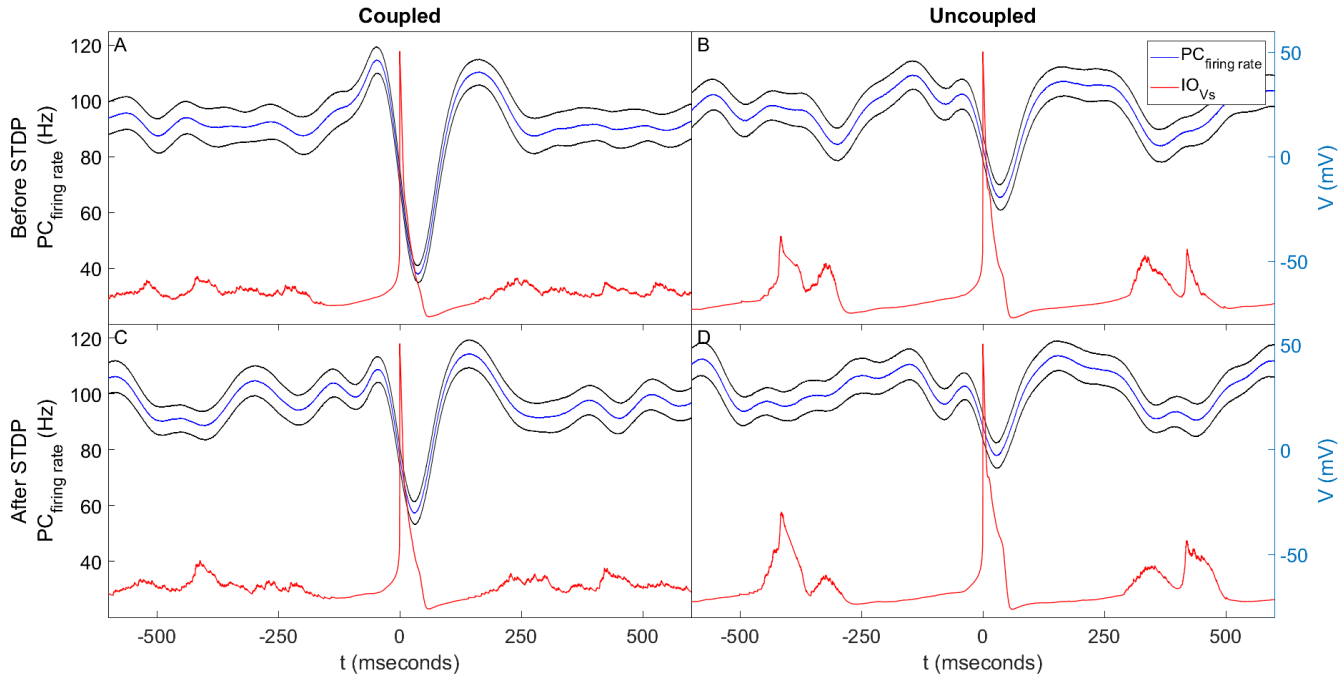


Fig. 10: CS Triggered PC Firing Averages for coupled (A and C) and uncoupled (B and D) scenarios for both before (A and B) and after (C and D) STDP. The left vertical axis shows the population firing rate of the PC with confidence intervals. The right axis the membrane potential of IO 1. The firing rate increases at about 50ms prior to a CS. It then starts to decreases and keeps decreasing more pronouncedly after the CS occurs. This decrease in firing rate following the CS is lower for the uncoupled scenario (C and D). Moreover, after STDP (B and D) the firing rate decreases less than before STDP. It is also seen that, for the coupled scenario (A and C), following a CS the firing rate increases until about 100ms after the CS. Then, the rate decreases until reaching a plateau. This is not seen in the uncoupled scenario (B and D). The firing rate reaches the plateau at about 100ms and does not decrease before reaching it. This is in relation to the lower variability in CS period in the uncoupled scenario.

3) Amplitude of the IO spikes
   - We also find larger amplitude of the spikes in IO 1 than in IO 2.
   - The average amplitude of the IO spikes for coupled cells 1 and 2 is of 50.83 mV and 50.66 mV with standard deviations 4.20 mV and 0.98 mV, respectively.
   - For the uncoupled cells, IO cell 1 has an amplitude average of 48.46 mV with a standard deviation of 10.56 mV and IO cell 2 has 50.27 mV amplitude average with 0.52 mV standard deviation.
   - There are no significant differences in spike amplitudes before and after STDP (for cell 1 $p = 0.2257$ for coupled and $p = 0.9349$ for uncoupled, and for cell 2 $p = 0.7794$ for the coupled scenario and $p = 0.1362$ for the uncoupled one).
4) PC response:
   - The mean PC population firing rate after STDP is of 96.67 Hz for coupled cells and 101.33 Hz for the uncoupled IO cells scenario.
   - For both coupled and uncoupled scenarios, the PC fires more after STDP.
5) DCN response:
   - The mean DCN population firing rate is of 36.21 Hz for coupled cells and 32.19 Hz for uncoupled cells.
   - As expected from the higher PC firing rates, the DCN firing rates are decreased with Plasticity.

E. What are the Main Differences in the Cerebellar Loop Response After Training?

When plasticity is added to the cerebellar loop, the response changes significantly. This can be seen in Fig. 9. The postsynaptic current of the PC is modified. The synapse is depressed when the noise decreases at about 100 ms prior to a CS. Following the drop in PSC, the current does not increase as much as it does when there is no STDP. It stays closer to 1nA and then increases instantaneously when the IO spike happens. In Fig. 10 it is shown that the variability in the PC firing rate is decreased. Moreover, the PC firing rate increases less prior to the IO spike as than when there is no plasticity. This indicates that the synapse has been modified and recognizes that an IO spike is arriving.

F. How Do the Firing Rates of Each Cell Change After STDP?

Fig. 11.A shows that there is a high level of cross-correlation of the IO firing rates before and after training for the uncoupled scenario. This indicates that training leads to little change in the uncoupled IO firing rates. On the other hand, in the coupled scenario the IO have significantly different firing rates. For the PC, the correlation is also lower for the coupled than the uncoupled scenarios (see Fig. 11.B). Nevertheless, it is still a relatively high correlation (peak at 0.58). Moreover, the peak amplitude of the uncoupled is larger than the coupled correlation, indicating that for the coupled scenario the PC response is modified. Namely, in the coupled scenario the firing rate distribution of the PC is different after training. The difference in the peaks of the correlation between the two scenarios is significantly larger for the DCN plot (Fig. 11.C). In the coupled scenario learning lead to significant changes in the DCN firing rates.

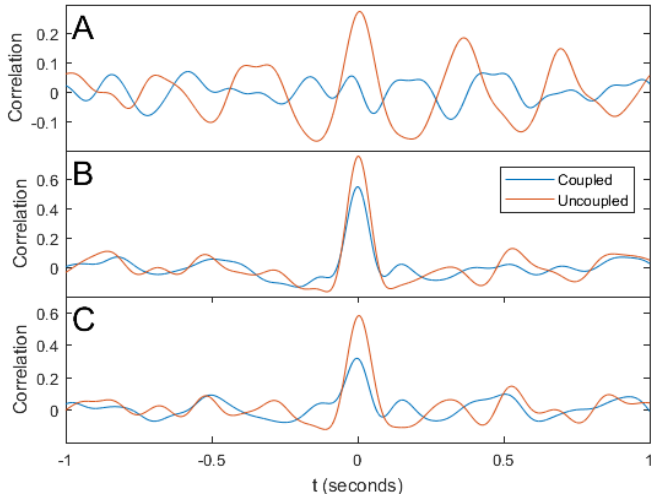


Fig. 11: Cross-Correlation plots between the population firing rates before and after STDP for each scenario (coupled and uncoupled). A: IO cell cross correlation. B: PC cross correlation. C: DCN cross correlation. In the coupled scenario (blue), the three cells have lower correlation.

G. How Does the Input Correlate with the PC Output After Training?

In the coupled scenario (Fig. 12.A) the cross correlation of the membrane potential of PC 1 with the PSC of PC 1 indicates that, before training, the PSC affects what will happen to the membrane potential. On the other hand, after training, what happened earlier has less effect on the membrane potential. A similar phenomenon is found for the cross correlation of the membrane potential of PC 2 and PSC of PC 2 (see Fig. 12.D). However, the peak appears with a delay of 100ms. This indicates that after STDP seems to predict the current. An analogous delay is found for the cross correlations of PC 1 PSC 2, and for PC 2 and PSC 1. This indicates that the main actors are both PSC 2 and PC 2. In contrast, for the uncoupled scenario, we do not see after training an effect of the PSC prior on the PC response (see Fig. 13). Nevertheless, we see that learning leads to a decrease in the sidelobes, indicating a lower response of the PC to the PSC. Overall, Fig. 12 and 13 show that plasticity plays a larger role in the coupled scenario.

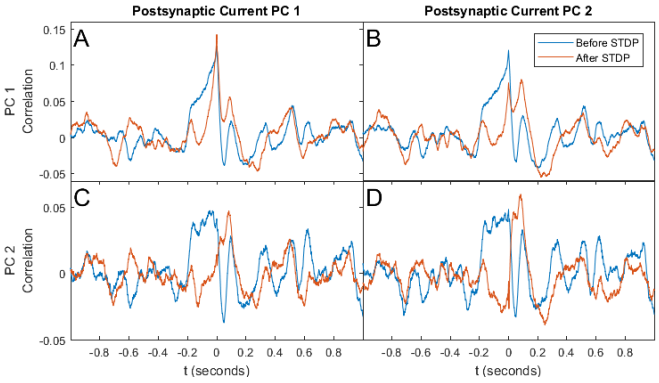


Fig. 12: PC PSC Cross-Correlation coupled scenario. A: PC 1-PSC 1 cross correlation. Before training there is an increasing correlation starting 200 ms prior. After training, a sharpened response and higher peak is found. B: PC 1-PSC 2 cross correlation. After STDP there is less correlation and the response prior has less effect. C: PC 2-PSC 1 cross correlation. After STDP the prior correlation is lower and the peak is now at a 100ms delay. D: PC 2-PSC 2 cross correlation. The peak correlation increases after STDP.

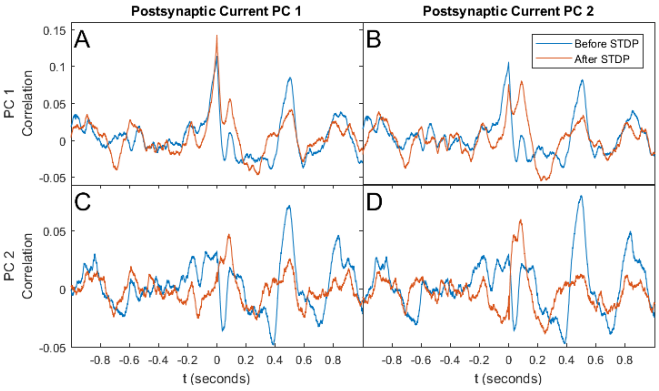


Fig. 13: PC PSC Cross-Correlation uncoupled scenario. A: PC 1-PSC 1 cross correlation. After STDP the peak is higher.. B: PC 1-PSC 2 cross correlation. After STDP there is less correlation. C: PC 2-PSC 1 cross correlation. After STDP the prior correlation is lower and the peak is now at a 100ms delay. The sidelobes found before STDP at 0.5 seconds are lowered after STDP. D: PC 2-PSC 2 cross correlation. The center peak correlation increases after STDP and the sidelobes are decresed.

Looking closer at the PSC 2 (see Fig. 14), we find that, for this simulation, after training the correlation has oscillatory components. This means that (especially for the uncoupled scenario) PC 2 learns specific characteristics of the noise by acquiring this oscillatory response.

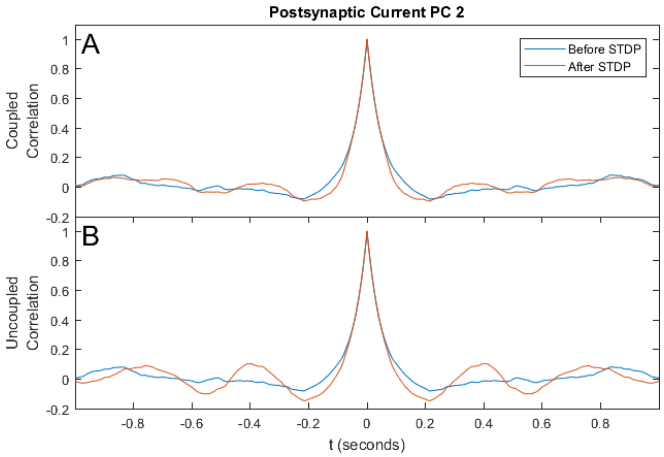


Fig. 14: Auto Correlation Of the PSC 2 for both coupled (A) and uncoupled (B) scenarios. After STDP (red) the auto correlation shows oscillatory components which are enhanced in the uncoupled scenario (B).

IV. Discussion

This cerebellar loop is a simplified model of the olivo-cerebellar system. It is missing various components such as inhibitory interneurons or basket and stellate cells. Nevertheless, the study of this model enables a vast range of questions with regards to the control of the musculoskeletal system as well as sensorimotor feedback. The importance of IO coupling is shown for a small amount of cells. Increasing the cell population opens the possibility for both a further research in the effect that coupling has, as well as a study in the properties of subthreshold oscillations of the IO. The significance of this work and its importance in the understanding of the function of the olivocerebellum becomes clear when looking at its position in motorcontrol. Boahen proposes that IO neurons use their subthreshold oscillations to mirror joint dynamics to implement an inverse controller (Boahen and Alvarez-Icaza, 2012). On the other hand, Higgins proposes that, in motor control, the discharge of antagonist motor neurons is regulated in concert with that of agonist muscles (Higgins, 1986) and the olivocerebellum is a modulator. The idea that IO neurons can express the dynamics required to mirror biomechanical joints and that motor control is regulated by discharges of antagonist and agonist muscle neurons could be investigated using the current cerebellar loop. The signals from the antagonist and agonist muscles are correlated temporal information that enter the loop through the PF-PC synapse. Moreover, injecting a current into the IO allows us to affect overall motor output performance and the joint's natural dynamics with amplitude proportional to the amount of current can be found (Boahen and Alvarez-Icaza, 2012).

V. Conclusion

The present model exhibits an interesting interplay between coupling and plasticity. In the presence of coupling, there is a significant decrease in the correlation of the distribution of firing rates before and after training. Furthermore, the firing distribution of the PCs is similar in the uncoupled scenario. Which, according to theory, corresponds to a more robust encoding of the noise. In contrast, in the coupled scenario, the PCs are separating the patterns (Negrello and Schutter, 2016; Sotelo et al., 1974; Schweighofer et al., 1999).
Plasticity has also a significant effect in the characteristics of the signal response of the network. Indeed, after training the IO, PC and DCN firing rates are higher. In addition, the trained system recognizes the drop in noise preceding a CS and depresses the synapses avoiding an increase in the firing rate of the PC (see Fig. 11).
The observed difference in the cross-correlation between the membrane potential and PSC of PC (Fig. 12 and 13) shows that the network responds or recognizes the parameters of each noise source. For both scenarios (coupled and uncoupled) the trained loop recognizes specific characteristics of PSC 2, namely the noise with higher intensity and smaller variance. This preference is enhanced for IO gap junctions with lower conductances (lower coupling).

Acknowledgement

The author would like to thank W. Mugge and A. Schouten for providing the possibility to work on this project, as well as their contribution in stimulating suggestions and encouragement. A special thanks of gratitude is expressed to Mario Negrello, as for his countless talks and motivated speeches about this subject matter made the work much easier.

Appendix A
Modeling Schweighofer's Inferior Olive Neuron Model with the *Brian 2* Simulator

This appendix presents how to implement Schweighofer's two compartmental model of the Inferior Olive. This model consists of the soma and dendrites compartments and a gap junction in which cells are connected. Sec. A-A introduces the model by explaining the role of each ionic current. Sec. A-B shows how to implement the general cell model in *Brian2*. Sec. A-B3 defines the parameters of the standard cell and how to implement them. The results are shown in Sec. A-C and the gap junction between two cells is shown in subsection A-C1.

A. Introducing Schweighofer's Model

In the original model introduced by Schweighofer (Schweighofer et al., 1999) it is discussed how IO cells have a rhythmic activity with a frequency of 4-8 Hz, that takes the form of either rhythmic generation of sodium spikes or subthreshold sinusoid-like oscillations (Bal and McCormick, 1997; Bernardo and Foster, 1986; Llinás and Yarom, 1986). Spontaneous oscillations occur for about 10% of these IO cells (Schweighofer et al., 1999). Moreover, IO cells generate dendritic spikes as a result to an injection of depolarizing currents, whereas somatic spikes are generated due to release from hyperpolarization (Llinás and Yarom, 1981). Furthermore, IO cells are electronically coupled by a gap junction at the dendrites (Sotelo et al., 1974). This coupling, added to the underlying rhythmicity, suggests that the inferior olive

is a network of damped oscillators. When these are coupled, sustained oscillations can be achieved (Schweighofer et al., 1999). Inferior olivary cells receive three types of inputs. There is one excitatory dendritic input and two inhibitory inputs: one affecting the excitability of the cell and the other one modulating the coupling strength between the cells (Lang et al., 1996).

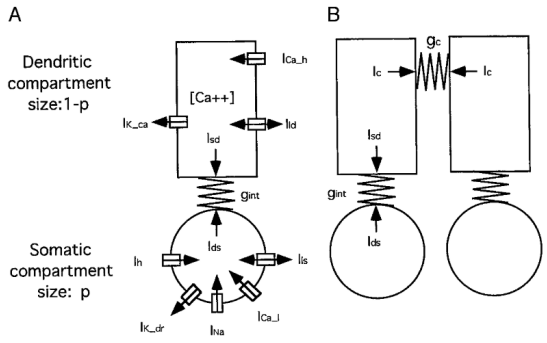


Fig. 15: Original model of inferior olive (IO) neurons (Schweighofer et al., 1999).
A: 2-compartment biophysical model of an IO cell. B: 2 cells electronically coupled by a single hypothetical gap junction.

Somatic Currents		Dendritic Currents	
I_{ds}	Current flowing out into the dendritic compartment	I_{sd}	Current flowing into the somatic compartment
I_{ls}	Somatic leakage current	I_{ld}	Dendritic leakage current
I_{Ca_l}	Low-threshold calcium inward current	I_{Ca_h}	High-threshold calcium inward current
I_h	Anomalous inward rectifier current	$I_{K_{Ca}}$	Calcium-activated potassium current
I_{Na}	Hodgkin-Huxley type inward sodium current	I_c	Electronic coupling current between the cells
$I_{K_{dr}}$	Hodgkin-Huxley type outward delayed rectifier potassium current		

TABLE VIII: Summary of Ionic Conductances in the soma and dendrites

Based on previous studies, Schweighofer developed a model that includes the membrane currents of the IO neurons (as show in Fig. 15). Table VIII summarizes these currents and shows which ones are in the soma and and which are in the dendrites of the IO cell. Current is exchanged between the two compartments, this is represented by I_{ds} and I_{sd} as the current flowing out into the dendritic compartment and the current flowing into the somatic compartment respectively. Both compartments also have leakage currents: I_{ls} for the soma and I_{ld} for the dendrites (Schweighofer et al., 1999). Two types of calcium currents are present in the IO: the low- and high- threshold currents (I_{Ca_l} and I_{Ca_h} respectively). While the former is in the soma, the latter is in the dendrites (Llinás and Yarom, 1981). The IO cell is excited in response to hyperpolarizing currents as a consequence of the I_{Ca_l}'s window of conductance being around the resting membrane potential. On the other hand, I_{Ca_h} is noninactivating thus, a depolarizing dendritic input results in a prolonged plateau potential. As calcium enters the dendrites, the calcium-activated potassium current, $I_{K_{Ca}}$ is activated. This current abruptly terminates the plateau potential after about 30 milliseconds (Llinás and Yarom, 1981). Due to the $I_{K_{Ca}}$'s long time constant of several hundred milliseconds, there is a long afterhyperpolarization which, in turn, deinactivates the I_{Ca_l}, resulting in a postinhibitory rebound spike. The afterhyperpolarization also activates an anomalous rectifying current I_h as it is a current that activates at hyperpolarized potentials. This current contributes to the subthreshold oscillations as it contributes to amplitude and frequency (Schweighofer et al., 1999; Bal and McCormick, 1997). Furthermore, similarly to Hodgkin-Huxley type neuron models, somatic sodium spikes are generated with the sodium current I_{Na}. These are terminated by an outward delayed rectifier potassium current $I_{K_{dr}}$ (Llinás and Yarom, 1981). Finally, the electronic coupling between the cells is represented by a current, I_c. All the currents are used in the next section but the coupling between cells is shown in Sec. A-C1.

In the next sections the Inferior Olive neuron model developed by Schweighofer (Schweighofer et al., 1999) is presented. The approach for modeling in *Brian2* is shown subsequently.

B. Implementing the General Cell Model

This model's electrotonic properties are based on two morphological parameters: the ratio of the somatic to total surface areas p and the electrotonic coupling conductance of the compartments g_{int}. For each compartment of Schweighofer's model, the membrane potential is defined by the following equation (Eq. (8)).

$$C_m \frac{dV}{dt} = -\sum_i I_i + I_{app} \tag{1}$$

where C_m is the membrane capacitance, I_{app} is the applied current (common to both compartments) and $\sum_i I_i$ represent the sum of the ionic currents, the current flowing in between the comparments and the current flowing through the gap junctions. In *Brian* the general equations are shown bellow. The applied current for the soma and the dendrites are I_{app_s} and I_{app_d}. This is useful when looking at the response to dendritic or somatic pulses. I_c is the coupling between the cells and is defined in the synapse created in Sec. A-C1.

```
eqs_IO_V = '''
dVs/dt = (-(I_ds + I_ls + I_Na + I_Ca_l + I_K_dr +
    I_h) + Iapp_s)/Cm : volt
dVd/dt = (-(I_sd + I_ld + I_Ca_h + I_K_Ca + I_c) +
    Iapp_d)/Cm : volt
I_c : metre**-2*amp
```

Somatic Currents	Dendritic Currents
$\sum_i I_i = I_{Ca_l} + I_h + I_{Na} + I_{K_{dr}} + I_{ds} + I_{ls}$	$\sum_i I_i = I_{Ca_h} + I_{K_{Ca}} + I_{sd} + I_{ld} + I_c$
$I_{Ca_l} = g_{Ca_l} k^3 l (V_s - V_{Ca})$	$I_{Ca_h} = g_{Ca_h} r^2 (V_d - V_{Ca})$
$I_h = g_h q (V_s - V_h)$	$I_{K_{Ca}} = g_{K_{Ca}} s (V_d - V_k)$
$I_{Na} = g_{Na} m_\infty^3 h (V_s - V_{Na})$	$I_{ld} = g_{ld} (V_d - V_l)$
$I_{K_{dr}} = g_{K_{dr}} n^4 (V_s - V_k)$	$I_{sd} = \left[\frac{g_{int}}{(1-p)}\right] (V_d - V_s)$
$I_{ds} = \left(\frac{g_{int}}{p}\right) (V_s - V_d)$	$I_c = g_c f (V_d - V_{de})(V_d - V_{de})$
$I_{ls} = g_{ls} (V_s - V_l)$	

TABLE IX: Equations of the Ionic Conductances

```
Iapp_s : metre**-2*amp
Iapp_d : metre**-2*amp
'''
```

Listing 1: General Cell Model

1) Somatic and Dendritic Currents: The equations to the different ionic currents discussed in Sec. A-A are shown in Tab. IX. Their implementation in *Brian* is shown next. The unit of these currents is μamp/cm^2. However, since *Brian* is unit consistent, these are defined as amp/meter^2 (the multiplication of the conductances (mS/cm^2) and voltage difference ($mvolt$) result in μamp/cm^2, hence, the unit is consistent).

```
eqs_IO_Isom = '''
I_ls    = g_ls*(Vs-V_l)              : metre**-2*amp
I_ds    = (g_int/p)*(Vs-Vd)          : metre**-2*amp
I_Na    = g_Na*m_inf**3*h*(Vs-V_Na)  : metre**-2*amp
I_Ca_l  = g_Ca_l*k*k*k*l*(Vs-V_Ca)   : metre**-2*amp
I_K_dr  = g_Kdr*n*n*n*n*(Vs-V_K)     : metre**-2*amp
I_h     = g_h*q*(Vs-V_h)             : metre**-2*amp
'''
eqs_IO_Iden = '''
I_sd  = (g_int/(1-p))*(Vd-Vs)        : metre**-2*amp
I_ld  = g_ld*(Vd-V_l)                : metre**-2*amp
I_Ca_h  = g_Ca_h*r*r*(Vd-V_Ca)       : metre**-2*amp
I_K_Ca  = g_K_Ca*s*(Vd-V_K)          : metre**-2*amp
'''
```

Listing 2: Somatic and Dendritic Currents

2) Activation/Inactivation Equations: The dynamics of each activation or inactivation variable is given by the differential equation:

$$\frac{dx}{dt} = \frac{(x_\infty - x)}{\tau_x} \tag{2}$$

where x indicates the variables h, k, l, n, q, r and s. Tab. X shows the equations of the steady-state activation/inactivation variables, as well as their time constant (except for the instantaneous m_∞. Their implementation in Brian is shown in the next page.

```
eqs_IO_activation = '''
dh/dt = (h_inf - h)/tau_h : 1
dk/dt = (k_inf - k)/tau_k : 1
dl/dt = (l_inf - l)/tau_l : 1
dn/dt = (n_inf - n)/tau_n : 1
dq/dt = (q_inf - q)/tau_q : 1
dr/dt = (r_inf - r)/tau_r : 1
ds/dt = (s_inf - s)/tau_s : 1
'''
eqs_IO_inf = '''
m_inf = alpha_m /(alpha_m+beta_m)        : 1
h_inf = alpha_h/(alpha_h+beta_h)         : 1
k_inf = 1/(1+e**(-(Vs/mvolt+61)/4.2))    : 1
l_inf = 1/(1+e**((Vs/mvolt+85.5)/8.5))   : 1
n_inf = alpha_n/(alpha_n+beta_n)         : 1
q_inf = 1/(1+e**((Vs/mvolt+75)/(5.5)))   : 1
r_inf = alpha_r/(alpha_r + beta_r)       : 1
s_inf = alpha_s/(alpha_s+beta_s)         : 1
'''
eqs_IO_tau = '''
tau_h = 170*msecond/(alpha_h+beta_h) : second
tau_k = 5*msecond : second
tau_l = 1*msecond*(35+(20*e**((Vs/mvolt+160)/30/(1+e
    **((Vs/mvolt+84)/7.3))))) : second
tau_n = 5*msecond/(alpha_n+beta_n) : second
tau_q = 1*msecond/(e**((-0.086*Vs/mvolt-14.6))+e
    **((0.07*Vs/mvolt-1.87))) : second
tau_r = 5*msecond/(alpha_r + beta_r) : second
tau_s = 1*msecond/(alpha_s + beta_s) : second
'''
```

Listing 3: Activation/Inactivation Equations

The time constants τ are in milliseconds, this means that the equations are defined as being in second but the constants are given in milliseconds. Moreover, as the potentials are in millivolts, those potentials have to be divided by *mvolt* as the constants in the equations are all in millivolt. Furthermore, the Forward and Backward rates α and β respectively, for each conductance type are defined in Tab. XI. In the next page, these are implemented in *Brian*, analogously to the previouse equations.

Conductance Type	Steady-State Activation/Inactivation	Time Constant (ms)
I_{Na}	$m_\infty = \frac{\alpha_m}{\alpha_m + \beta_m}$ $h_\infty = \frac{\alpha_h}{\alpha_h + \beta_h}$	$\tau_h = \frac{170}{\alpha_h + \beta_h}$
$I_{K_{dr}}$	$n_\infty = \frac{\alpha_n}{\alpha_n + \beta_n}$	$\tau_n = \frac{5}{\alpha_n + \beta_n}$
I_{Ca_l}	$k_\infty = \frac{1}{1 + e^{\frac{-(V+61}{4.2}}}$ $l_\infty = \frac{1}{1 + e^{\frac{(V+85.5)}{8.5}}}$	$\tau_k = 5$ $\tau_l = \frac{20e^{\frac{(V+160)}{30}}}{1 + e^{\frac{(V+84)}{7.3}}} + 35$
I_h	$q_\infty = \frac{1}{1 + e^{\frac{(V+75)}{5.5}}}$	$\tau_q = \frac{1}{e^{(-0.086V-14.6)} + e^{(0.07V-1.87)}}$
I_{Ca_h}	$r_\infty = \frac{\alpha_r}{\alpha_r + \beta_r}$	$\tau_r = \frac{1}{\alpha_r + \beta_r}$
$I_{K_{Ca}}$	$s_\infty = \frac{\alpha_s}{\alpha_s + \beta_s}$	$\tau_s = \frac{1}{\alpha_s + \beta_s}$

TABLE X: Steady-state Activation/Inactivation and Time Constants of the Ionic Conductances

Conductance Type	Forward Rate Function α	Backward Rate Function β
I_{Na}	$\alpha_m = \frac{0.1(V+41)}{1 - e^{\frac{-(V+41)}{10}}}$ $\alpha_h = 5.0e^{\frac{-(V+60)}{15}}$	$\beta_m = 9.0e^{\frac{-(V+66)}{20}}$ $\beta_h = \frac{(V+50)}{1 - e^{\frac{-(V+50)}{10}}}$
$I_{K_{dr}}$	$\alpha_n = \frac{(V+41)}{1 - e^{\frac{-(V+41)}{10}}}$	$\beta_n = 12.5e^{\frac{-(V+51)}{80}}$
I_{Ca_h}	$\alpha_r = \frac{1.6}{1 + e^{\frac{-(V-5)}{14}}}$	$\beta_r = \frac{0.02(V+8.5)}{1 - e^{\frac{(V+8.5)}{5}}}$
$I_{K_{Ca}}$	$\alpha_s = \min(2 \cdot 10^{-5}[Ca^{2+}], 0.01)$	$\beta_s = 0.015$

TABLE XI: Forward and Backward rates α and β of the Ionic Conductances

```
eqs_IO_alpha = '''
alpha_m = (0.1*(Vs/mvolt + 41))/(1-e**(-(Vs/mvolt
    +41)/10)) : 1
alpha_h = 5.0*e**(-(Vs/mvolt+60)/15) : 1
alpha_n = (Vs/mvolt + 41)/(1-e**(-(Vs/mvolt+41)/10))
     : 1
alpha_r = 1.7/(1+e**(-(Vd/mvolt - 5)/13.9)) : 1
alpha_s = ((0.00002*Ca/mM)*int((0.00002*Ca/mM)<0.01)
    + 0.01*int((0.00002*Ca/mM)>=0.01)) : 1
'''
eqs_IO_beta = '''
beta_m = 9.0*e**(-(Vs/mvolt+60)/20)
          : 1
beta_h = (Vs/mvolt+50)/(1-e**(-(Vs/mvolt+50)/10))
          : 1
beta_n = 12.5*e**(-(Vs/mvolt+51)/80)
          : 1
beta_r = 0.02*(Vd/mvolt + 8.5)/(e**((Vd/mvolt + 8.5)
```

```
    /5)-1) : 1
beta_s = 0.015
           : 1
'''
```

Listing 4: Forward and Backward rates α and β

Finally, the calcium concentration is calculated following Eq. (3) and implemented in the code. The unit for molar concentration in *Brian* is mmolar (1mM = 1mol/m^3) and not molar (10^3 mol/m^3).

$$\frac{d[Ca^{2+}]}{dt} = -3.0 I_{Ca_h} - 0.075[Ca^{2+}] \tag{3}$$

```
eqs_IO_Ca = '''
dCa/dt = (-3*I_Ca_h*((uamp / cm**2)**-1)*mM - 0.075*
    Ca)/ms : mM
'''
```

Listing 5: Calcium Concentration

In the definition of the equations, the neuron-dependent parameters are defined as vector. Then, all the equations are added to one equation: (*eqs_IO*). The neuron group can be created using the equation *NeuronGroup* as shown below. The time step for the clock is chosen to be 0.025 milliseconds and the Euler method as the integration method. The number of cells modelled can be defined in *N_Cells_IO* and a threshold can be given, for instance: spiking occurs after a certain calcium concentration is reached ($Ca > 2mM$).

```
eqs_vector = '''
V_Na : volt
V_K  : volt
V_Ca : volt
V_l  : volt
V_h  : volt
Cm : farad*meter**-2
g_Na    : siemens/meter**2
g_Kdr   : siemens/meter**2
g_Ca_l  : siemens/meter**2
g_h     : siemens/meter**2
g_Ca_h  : siemens/meter**2
g_K_Ca  : siemens/meter**2
g_ls  : siemens/meter**2
g_ld  : siemens/meter**2
g_int  : siemens/meter**2
p : 1
'''
eqs_IO = eqs_IO_beta
eqs_IO += eqs_IO_alpha
eqs_IO += eqs_IO_tau
eqs_IO += eqs_IO_inf
eqs_IO += eqs_IO_activation
eqs_IO += eqs_IO_Iden
eqs_IO += eqs_IO_Isom
eqs_IO += eqs_IO_Ca
eqs_IO += eqs_IO_V
eqs_IO += eqs_vector

IO_SingleNeuron = NeuronGroup(N_Cells_IO, model =
    eqs_IO, threshold='Ca>2*mM', method = 'euler',
    name = 'SchweighoferOlive',dt=0.025*msecond)
```

Listing 6: Creating the Neuron Group

3) Modeling a "Standard" Cell: The parameters used by Schweighofer to model the "standard cell" are shown in Tab. XII. These are implemented in the code following the neuron group created previously. The parameters are implemented for a neuron group of size *N_Cells_IO* and thus, a *forloop* is created.

Conductances (mS/cm^2)		Reversal Potentials (mV)	
g_{Na}	70	V_{Na}	55
$g_{K_{dr}}$	18	V_K	-75
g_{Ca_l}	1.0	V_{Ca}	120
g_h	1.5	V_h	-43
g_{Ca_h}	4.0	V_l	-10
$g_{K_{Ca}}$	35	Membrane Capacitance ($\mu F/cm^2$)	
g_{ls}	0.015		
g_{ld}	0.015	C_m	1
Cell Morphology			
g_{int}	0.13	p	0.20

TABLE XII: Parameters of the Standard Cell

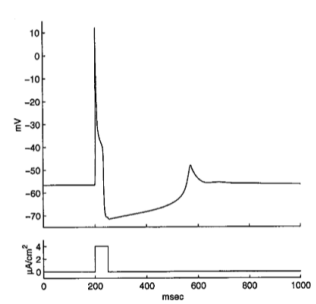
```
IO_SingleNeuron = NeuronGroup(N_Cells_IO, model =
    eqs_IO, threshold='Ca>2*mM', method = 'euler',
    name = 'SchweighoferOlive',dt=dtt)

for ii in range(0, N_Cells_IO, 1):
    IO_SingleNeuron.V_Na[ii] = 55*mvolt
    IO_SingleNeuron.V_K[ii] = -75*mvolt
    IO_SingleNeuron.V_Ca[ii] = 120*mvolt
    IO_SingleNeuron.V_l[ii] = -10*mvolt
    IO_SingleNeuron.V_h[ii] = -43*mvolt
    IO_SingleNeuron.Cm[ii] = 1*uF*cm**-2
    IO_SingleNeuron.g_Na[ii] = 70*mS/cm**2
    IO_SingleNeuron.g_Kdr[ii] = 18*mS/cm**2
    IO_SingleNeuron.g_Ca_l[ii] = 1.0*mS/cm**2
    IO_SingleNeuron.g_h[ii] = 1.5*mS/cm**2
    IO_SingleNeuron.g_Ca_h[ii] = 4.0*mS/cm**2
    IO_SingleNeuron.g_K_Ca[ii] = 35*mS/cm**2
    IO_SingleNeuron.g_ls[ii] = 0.015*mS/cm**2
    IO_SingleNeuron.g_ld[ii] = 0.015*mS/cm**2
    IO_SingleNeuron.g_int[ii] = 0.13*mS/cm**2
    IO_SingleNeuron.p[ii] = 0.20
```

Listing 7: Parameters of the Standard Cell

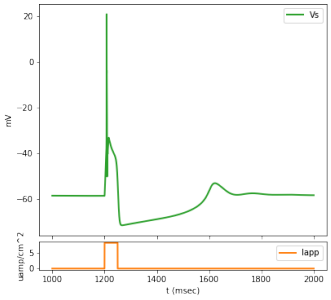
C. Results

The standard cell is defined and the neuron group is created. To reproduce Figure 2 of the paper (seen in Fig. 16a, 16c and 16e), a dendritic current pulse of 50 milliseconds duration and an intensity of 8 μamp/cm^2 needs to be given to the cell. This is shown in the next page. There is a transient of 1200 milliseconds so that the cell reaches its equilibrium before the current pulse is given. Nevertheless, the plots start at 1000 milliseconds to show the response of the somatic membrane potential V_s to the current pulse. Fig. 16b, 16d and 16f show the response of the cell as modelled in *Brian*.

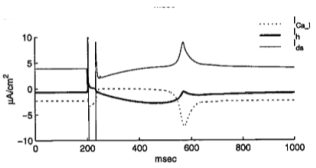
```
transient = 1200*msecond
step_duration = 50*msecond
```



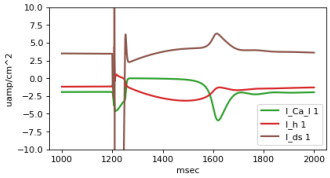
(a) Somatic membrane potential (Fig 2A from paper)



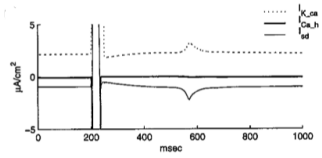
(b) Somatic membrane potential as modelled in *Brian*



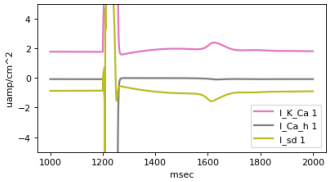
(c) Somatic currents (Fig 2B from paper)



(d) Somatic currents as modelled in *Brian*



(e) Dendritic currents (Fig 2C from paper)



(f) Dendritic currents as modelled in *Brian*

Fig. 16: Comparing Fig.2 of Schweighofer's paper with the results obtained from the *Brian* reproduction

```
inter_stimulus_interval = transient-step_duration
    +2000*msecond
N_Cells_IO = 1
I_app_initial = 0*uA/cm**2
Pulse_intensity = I_app_initial + (8*uA/cm**2)
IO_SingleNeuron.Iapp_s = [I_app_initial]
IO_SingleNeuron.Iapp_d = [I_app_initial]
run(transient)
IO_SingleNeuron.Iapp_d = [Pulse_intensity]
run(step_duration)
IO_SingleNeuron.Iapp_d = [I_app_initial]
run(inter_stimulus_interval)
```

Listing 8: Recreating Figure 3 of Paper

1) Coupling Two Cells: The equations for the electronic coupling of two cells is shown in Eq. (4). In *Brian* the number of cells *N_Cells_IO* is now updated to 2. The synaptic equations are given and defined in the *Synapses*. Fig. 17 and 18 show the response of the somatic membrane potentials of two cells. The first cell does not receive any current input and the second cell receives two current pulses of magnitude 1 μamp/cm^2 and duration 50 milliseconds. The first pusle is depolarizing and 1000 milliseconds later a second hyperpolarizing pulse is given to the second cell. For the cell to be more excitable the following parameters are changed: V_l = *10mV*, g_{Kdr} = *9mS/cm*^{2}, g_{Ca_l} = *1.2*mS/cm*^{2}, and g_{ls} = g_{ld} = *0.016*mS/cm*^{2}.

$$I_c = g_c f(V_d - V_{de})(V_d - V_{de})$$
$$f(V) = 0.6e^{-\frac{V^2}{50^2}} + 0.4 \tag{4}$$

```
# Synapse
eqs_IO_syn = '''
    I_c_pre = (g_c)*(0.6*e**(-((Vd_pre/mvolt-Vd_post
    /mvolt)/50)**2) + 0.4)*(Vd_pre-Vd_post) : metre
    **-2*amp (summed)
    g_c : siemens/meter**2
'''
IO_synapse = Synapses(IO_SingleNeuron,
    IO_SingleNeuron, eqs_IO_syn, name = 'IO_Synapse'
    )
IO_synapse_g_c = 0.18*mS/cm**2
IO_synapse.connect(i=[0,1],j=[1,0])
```

Listing 9: Coupling Two Cells

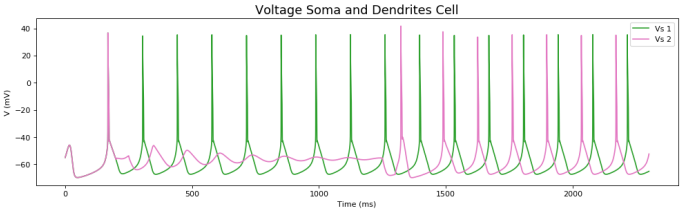


Fig. 17: Response of somatic membrane potentials of two coupled cells to applied current to cell 2

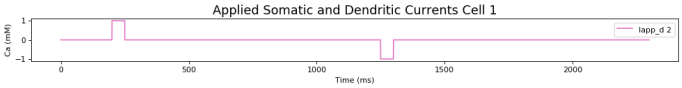


Fig. 18: Dendritic and somatic current pulse for cell 2

APPENDIX B
MODELING SYNAPSES OF AN ADEX PURKINJE CELL MODEL FOR STDP AND COMPLEX SPIKE PAUSE

A single compartment Purkinje Cell is considered using an adaptive exponential integrate-and-fire model (Sec. B-A). The PC and PF inputs are linked by an STDP synapse (Sec. B-B1), and a pause mechanism is generated after each complex spike by modeling the synapse between the PC and an IO cell (Schweighofer et al., 1999), (sec. B-B2).

A. Generall Cell

The membrane potential of the AdEx model is defined by Eq. (Eq. 5). It has two current inputs I_{Noise} and I_{app} which represent the current input from the noise (parallel fiber input) and the external pulse current.

$$\frac{dV}{dt} = \frac{1}{C_m}\left(g_L\left(E_L - V\right) + g_L\Delta_T e^{\left(\frac{V-V_T}{\Delta_T}\right)} + I_{Noise} + I_{app} - w\right)$$
$$\frac{dw}{dt} = \frac{a\left(V - E_L\right) - w}{\tau_w} \tag{5}$$

Here, Δ_T is the slope factor, V_T is the the threshold potential, w is the adaptation variable, a is the adaptation coupling factor and τ_w is the adaptation time constant.

```
PC_Equations = """
    dv/dt = (gL*(EL - v) + gL*DeltaT*exp((v - VT)/
    DeltaT) + I_Noise + I_app -w)/C : volt
    dw/dt = (a*(v - EL) - w)/(tauw) : amp

    I_app : amp
    I_Noise : amp

    C : farad
    gL : siemens
    EL : volt
    VT : volt
    DeltaT : volt
    Vcut : volt
    tauw : second
    a : siemens
    b : ampere
    Vr : volt
"""
```

Listing 10: General Cell Model

Tab. XIII shows the cell parameters. These parameters were taken from various sources (De Schutter and Bower, 1994; Brette and Gerstner, 2005; Badura et al., 2013; Herzfeld et al., 2015).

In the definition of the equations, the neuron-dependent parameters are defined as vector and the neuron is generated using the equation *NeuronGroup* as shown below. The time step for the clock is chosen to be 0.025 milliseconds and the euler method as the integration method. The number of cells modelled can be defined in *N_Cells_PC* and a threshold can be given, for instance: spiking occurs after a certain potential V_{cut} is reached. When a spike happens both the membrane potential and the adaptive variables are reset. Nevertheless, while the first resets to the resting potential, the second decays until either reaching 0 or the next spike time (Dayan and Abbott, 2005). This is also included in the neuron group as the reset variable.

Parameters	Values
C	75 pF
g_L	30 nS
E_L	-70.6 mV
V_T	-50.4 mV
Δ_T	2 mV
V_{cut}	$V_T + 5\Delta_T$
τ_w	144 ms
a	4 nS
b	0.0805 nA
V_r	-70.6 mV

TABLE XIII: PC Parameters

```
PC_SingleNeuron = NeuronGroup(N_Cells_PC, model=
    PC_Equations, threshold='v>Vcut', reset="v=Vr; w
    +=b", method='euler', name = 'PC',dt=0.025*ms)

for jj in range(0,N_Cells_PC,1):
    PC_SingleNeuron.C[jj] = 75*pF
    PC_SingleNeuron.gL[jj] = 30 * nS
    PC_SingleNeuron.EL[jj] = -70.6 * mV
    PC_SingleNeuron.VT[jj] = -50.4 * mV
    PC_SingleNeuron.DeltaT[jj] = 2 * mV
    PC_SingleNeuron.Vcut[jj] = PC_SingleNeuron.VT[jj
    ] + 5*PC_SingleNeuron.DeltaT[jj]
    PC_SingleNeuron.tauw[jj] = 144*ms
    PC_SingleNeuron.a[jj] = 4*nS
    PC_SingleNeuron.b[jj] = 0.0805*nA
    PC_SingleNeuron.Vr[jj] = -70.6*mV

PC_Statemon = StateMonitor(PC_SingleNeuron, ['v', 'w
    ', 'I_Noise', 'I_app','tauw','I_IO_PC'], record=
    True,dt=0.025*ms)
PC_Spikemon = SpikeMonitor(PC_SingleNeuron)
```

Listing 11: Neuron Group and Parameters

B. Results

Simple spikes have a frequency that can reach 100 spikes per second Kandel et al. (2000a). Thus, it is useful to plot the f-I curve to find the input current that is needed to find such a spiking frequency.

```
N_Cells_PC = 100
PC_SingleNeuron.I_app = '5*nA*i/N_Cells_PC'
run(1000*ms)
```

Listing 12: f-I curve

Fig. 19 shows that 100Hz firing frequency is reached for a current of $2nA$. Fig.20, 21 and 22 show the response of the Purkinje cell membrane potential V as well as the the adaptation variable w.

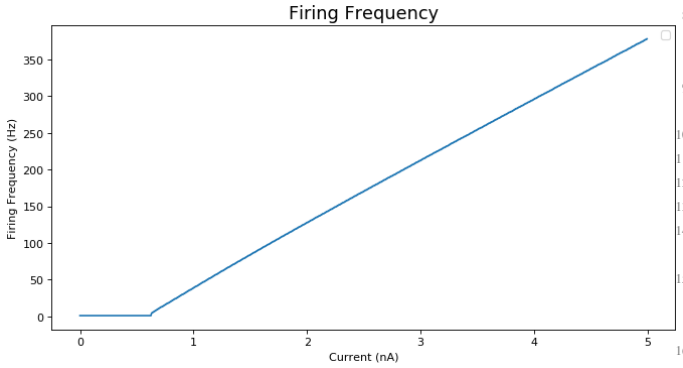


Fig. 19: Firing frequency vs. Current

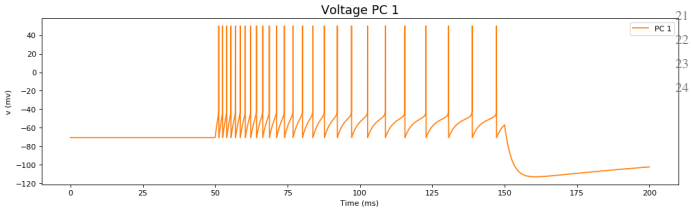


Fig. 20: Purkinje Cell Response

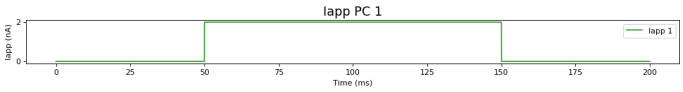


Fig. 21: Applied Current on Purkinje Cell

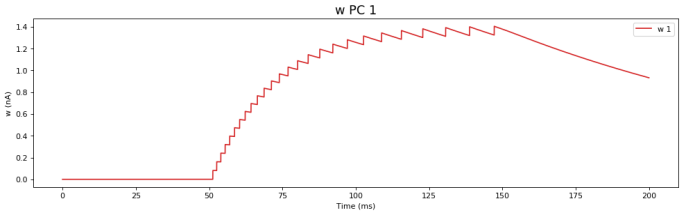


Fig. 22: w from Purkinje Cell

1) STDP: Purkinje Cell - Parallel Fiber Synapse: For the parallel fiber input, a neuron group is created with a current with mean I_0 and Ornstein-Uhlenbeck noise as shown in Eq. (6). Based on Fig. 19, I_0 is chosen to be $2nA$ with $\pm 0.5nA$ amplitude. This means that the σ of the OU component equals $0.25nA$ (shown in Fig. 23). The synapse is defined with the PF source being the presynaptic neuron and the PC being the postsynaptic neuron. The response of the Purkinje Cell is shown in Fig. 24.

$$\frac{dI}{dt} = \frac{(I_0 - I)}{\tau_{Noise}} + \frac{\sigma\xi}{\sqrt{\tau_{Noise}}} \tag{6}$$

```
N_noise = 1
tau_noise = 100*ms
eqs_noise = '''
dI/dt = (I0 - I)/tau_noise + sigma*xi*tau_noise
    **-0.5 : amp
I0 : amp
sigma : amp
'''
```

```
Noise = NeuronGroup(N_noise, eqs_noise, threshold =
    'True', method='euler',name = 'Noise',dt=0.025*
    ms)
Noise_statemon = StateMonitor(Noise, 'I', record=
    True,dt=0.025*ms)
eqs_syn = '''
    weight : 1 # gap junction conductance
    I_Noise_post = weight* (I_pre) : amp (summed)
'''
S = Synapses(Noise, PC_SingleNeuron, eqs_syn,name =
    'PC_Noise_Synapse',dt=0.025*ms)
S.connect(j='k for k in range(i-N_Cells_PC, i+
    N_Cells_PC) if i!=k-N_Cells_PC', skip_if_invalid
    =True)
S.weight = '1-(abs(i-j)/N_Cells_PC)'
Noise.I0 = 2*nA
Noise.I = 2*nA
Noise.sigma = 0*nA
run(100*ms)
Noise.sigma = 0.25*nA
run(500*ms)
Noise.sigma = 0*nA
run(100*ms)
```

Listing 13: Parallel Fiber input

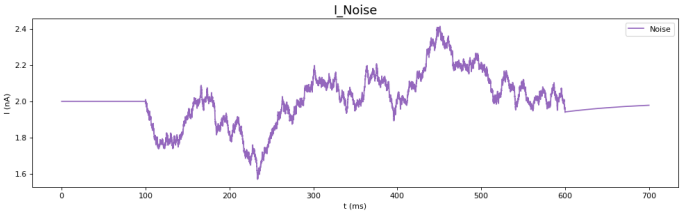


Fig. 23: A current with noise; I_0=$2nA$ and σ=$0.25nA$

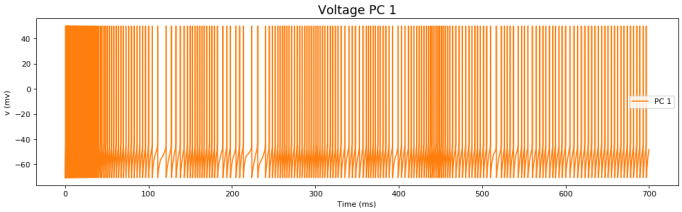


Fig. 24: Response of Purkinje cell membrane potential to parallel fiber noise input

2) Purkinje Cell - Inferior Olive Synapse: In the Synapses function, at each pre-synaptic event the adaptation variable w is updated to $+0.008nA$ to simulate a pause following a complex spike. This means that when there is a spike in the IO, w increases by $0.008nA$. This can be seem in Fig. 28. The response of the membrane potentials of both the PC and IO is shown in Fig. 25.

```
Synapse_IO_PC = Synapses(IO_SingleNeuron,
    PC_SingleNeuron, on_pre ='w +=(0.008*nA)', name
    = 'IO_PC_Synapse',method = 'euler',dt=0.025*ms)
Synapse_IO_PC.connect(i=1,j=0)
```

Listing 14: PC-IO Synapse

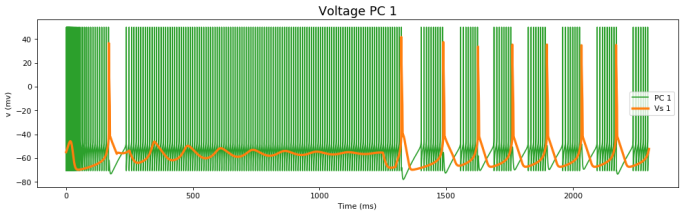


Fig. 25: Purkinje Cell Response

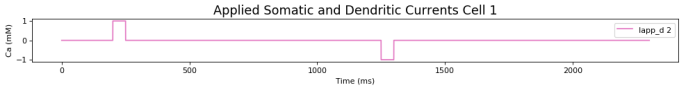


Fig. 26: Dendritic and Somatic current pulses for IO cell 2

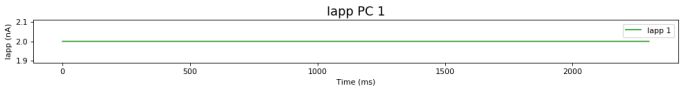


Fig. 27: Applied Current on Purkinje Cell

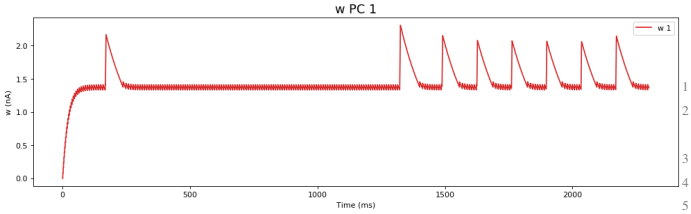


Fig. 28: w from Purkinje Cell

APPENDIX C
AN ADAPTIVE EXPONENTIAL INTEGRATE-AND-FIRE MODEL OF THE DEEP CEREBELLAR NUCLEI

Similar to the PC, the DCN is modelled as a single compartment adaptive exponential integrate-and-fire model (Sec. C-A). The PC inhibits the DCN when it spikes (Sec. C-B), and the DCN-IO synapse resets the oscillations of the IO after a DCN spike (Steuber, 2016) (Sec. C-C).

A. Generall Cell

The equation defining the change in membrane potential of the AdEx model very similar to Eq. 5 (Ch. B). Instead of I_{Noise} and I_{app}, its two current inputs are $I_{Intrinsic}$ and I_{PC}. This allows to model the intrinsic firing of the cell and to inhibit the DCN when the PC fires. The new differential equation is needed to describe I_{PC} is Eq. (7). When the PC is not spiking, $I_{PC} = 0nA$, however, as shown in Sec. C-B, upon a spike of the PC, I_{PC} is updated to a certain value. This means that the total current $I_{Intrinsic} - I_{PC}$ will be smaller and thus, the DCN will spike less (inhibition from PC). The corresponding listing is given in Listing 15.

$$\frac{dI_{PC}}{dt} = \frac{I_{PC_{max}} - I_{PC}}{\tau_I} \tag{7}$$

```
DCN_Equations = """
dv/dt = (gL*(EL - v) + gL*DeltaT*exp((v - VT)/DeltaT
    ) + I_intrinsic - I_PC - w)/C : volt
dw/dt = (a*(v - EL) - w)/tauw :
I_intrinsic : amp

dI_PC/dt = (0*nA - I_PC)/tauI : amp

C : farad
gL : siemens
EL : volt
taum : second
VT : volt
DeltaT : volt
Vcut : volt
tauw : second
a : siemens
b : ampere
Vr : volt
tauI : second
I_PC_max : amp
"""
```

Listing 15: General Cell Model

Tab. XIV shows the cell parameters. Most parameters are similar to the model of the PC, except for the membrane capacitante C_m, the adaptation time constant τ_w and the reset voltage V_r (Steuber et al., 2011, 2007; Steuber, 2016).

```
DCN_SingleNeuron = NeuronGroup(N_Cells_DCN, model=
    DCN_Equations, threshold='v>Vcut', reset="v=Vr;
    w+=b", method='euler', name = 'DCN',dt=0.025*ms)

for dd in range(0,N_Cells_DCN,1):
    DCN_SingleNeuron.C[dd] = 281*pF
    DCN_SingleNeuron.gL[dd] = 30 * nS
    DCN_SingleNeuron.EL[dd] = -70.6 * mV
    DCN_SingleNeuron.VT[dd] = -50.4 * mV
    DCN_SingleNeuron.DeltaT[dd] = 2 * mV
    DCN_SingleNeuron.Vcut[dd] = DCN_SingleNeuron.VT[
    dd] + 5*DCN_SingleNeuron.DeltaT[dd]
    DCN_SingleNeuron.tauw[dd] = 50*ms
```

Parameters	Values
C	281 pF
g_L	30 nS
E_L	-70.6 mV
V_T	-50.4 mV
Δ_T	2 mV
V_{cut}	$V_T + 5\Delta_T$
τ_w	144 ms
a	4 nS
b	0.0805 nA
V_r	-65 mV
τ_I	30 ms
$I_{PC_{max}}$	0 nA

TABLE XIV: DCN Parameters

```
    DCN_SingleNeuron.a[dd] = 4*nS
    DCN_SingleNeuron.b[dd] = 0.0805*nA
    DCN_SingleNeuron.Vr[dd] = -65*mV
    DCN_SingleNeuron.tauI[dd] = 30*ms
    DCN_SingleNeuron.I_PC_max[dd] = 0*nA

DCN_Statemon = StateMonitor(DCN_SingleNeuron, ['v',
    'I_PC','w'], record=True,dt=0.025*ms)
DCN_Spikemon = SpikeMonitor(DCN_SingleNeuron)
```

Listing 16: Parameters

Deep cerebellar nuclei cells have an intrinsic firing of about 90Hz (Person and Raman, 2012; Steuber, 2016). Looking at Fig. 29, $I_{Intrinsic}$ is chosen to be 1.34nA. The response of the DCN is shown in Fig. 30.

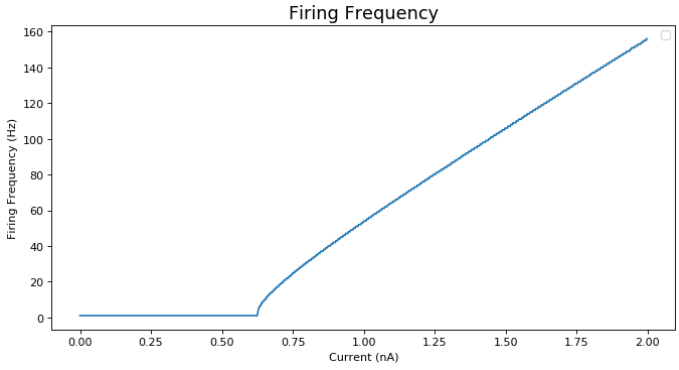


Fig. 29: f-I curve to find an $I_{Intrinsic}$ with 90Hz

B. Purkinje Cell - Deep Cerebellar Nuclei Synapse

The Purkinje cell inhibits the DCN when it is spiking, thus, the DCN can only fire when there is a pause in the PC spiking. The synapse is generated so that after a presynaptic spike

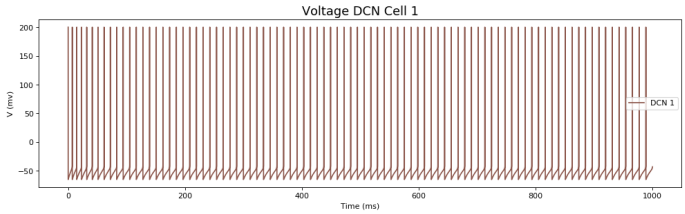


Fig. 30: Deep Cereballar Nuclei Intrinsic Spiking

occurs, the $I_{IO_{PC}}$ is inceased by 0.9*nA. The response of the DCN to a certain PC is shown in Fig. 31 and the behavior of $I_{IO_{PC}}$ in Fig. 33.

```
DCN_PC_Synapse = Synapses(PC_SingleNeuron,
    DCN_SingleNeuron, on_pre='I_PC_post = 0.9*nA',
    delay=2*ms, name = 'PC_DCN_Synapse',dt=0.025*ms)
DCN_PC_Synapse.connect(i=[0],j=[0])

PC_SingleNeuron.v = -70.6 * mV
DCN_SingleNeuron.v = -70.6 * mV
DCN_SingleNeuron.I_intrinsic = 1.34*nA
run(300*ms)

PC_SingleNeuron.I_app = 2*nA
run(400*ms)

PC_SingleNeuron.I_app = 0*nA
run(300*ms)
```

Listing 17: PC-DCN Synapse

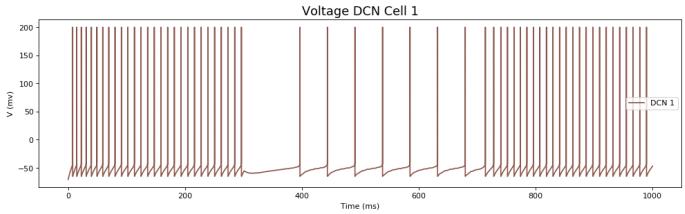


Fig. 31: Response of DCN to a certain PC

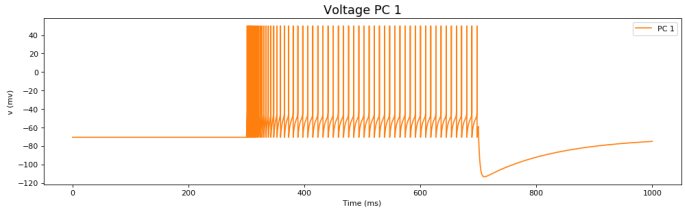


Fig. 32: Purkinje Cell AdEx

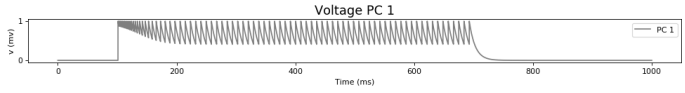


Fig. 33: Behavior of $I_{IO_{PC}}$

C. Deep Cerebellar Nuclei - Inferior Olive Synapse

When the DCN spikes, it resets the IO oscillation (Steuber, 2016). The generall cell equation from the IO model (Ch. A)

is modified to include the current $I_{IO_{DCN}}$, from the DCN to the IO. This is a current with mean 0μA/cm^2 and a decay time constant of 9 milliseconds. After spike of the DCN, $I_{IO_{DCN}}$ increases by 9μA/cm^2.

$$C_m \frac{dV}{dt} = -\sum_i I_i + I_{app} + I_{IO_{DCN}} \tag{8}$$

```
eqs_IO_V = """
dVs/dt = (-(I_ds + I_ls + I_Na + I_Ca_l + I_K_dr +
    I_h) + Iapp_s)/Cm : volt
dVd/dt = (-(I_sd + I_ld + I_Ca_h + I_K_Ca + I_c) +
    Iapp_d + I_IO_DCN)/Cm : volt
dI_IO_DCN/dt = (0*uA*cm**-2 - I_IO_DCN)/(9*ms) : amp
    *meter**-2
I_c : metre**-2*amp
Iapp_s : metre**-2*amp
Iapp_d : metre**-2*amp
"""
```

Listing 18: Modify IO

```
IO_DCN_Synapse = Synapses(DCN_SingleNeuron,
    IO_SingleNeuron, on_pre = 'I_IO_DCN_post += 9*uA
    *cm**-2', delay=3*ms, name = 'IO_DCN_Synapse',
    method = 'euler',dt=0.025*ms)
IO_DCN_Synapse.connect(i=[0,1],j=[0,1])
```

Listing 19: DCN-IO Synapse

Fig. 34 shows the response of two IO cells as shown in Sec. A-C1. The effect of adding the synapse between the DCN and the IO is shown in Fig. 35. It is clear that the oscillatons of the IO are reset due to the DCN spike.

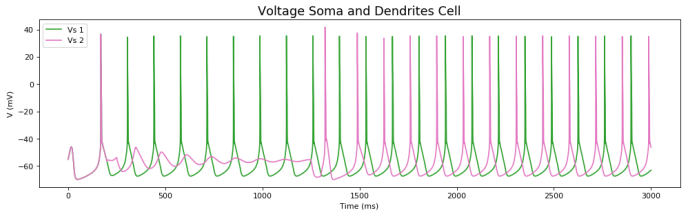


Fig. 34: Response of somatic membrane potentials of two coupled cells without DCN synapse

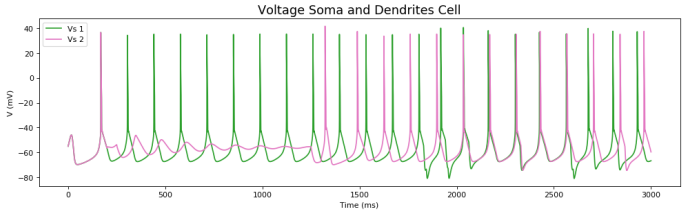


Fig. 35: Response of somatic membrane potentials of two coupled cells with DCN synapse

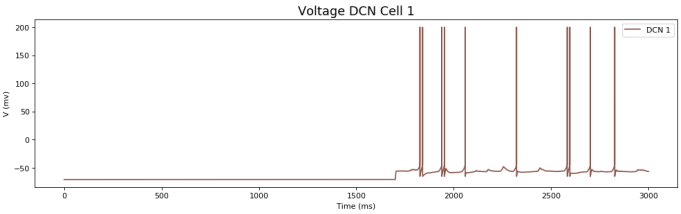


Fig. 36: Response of DCN

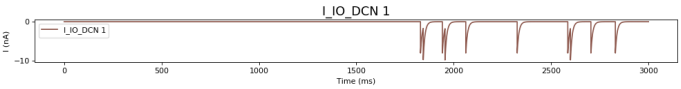


Fig. 37: I_IO_DCN

REFERENCES

Albus, J. S. (1971). A theory of cerebellar function. *Mathematical Biosciences*, 10(1):25 – 61.

Ankri, L., Husson, Z., Pietrajtis, K., Proville, R., Lna, C., Yarom, Y., Dieudonn, S., and Uusisaari, M. Y. (2015). A novel inhibitory nucleo-cortical circuit controls cerebellar golgi cell activity. 4:e06262.

Badura, A., Schonewille, M., Voges, K., Galliano, E., Renier, N., Gao, Z., Witter, L., Hoebeek, F., Chdotal, A., and De Zeeuw, C. (2013). Climbing fiber input shapes reciprocity of purkinje cell firing. 78.

Bal, T. and McCormick, D. (1997). Synchronized oscillations in the inferior olive are controlled by the hyperpolarization-activation cation current I_h. *The Journal of physiology*, 77.

Bernardo, L. and Foster, R. (1986). Oscillatory behavior in inferior olive neurons: mechanism, modulation, cell aggregates. *Brain Res. Bull.*, 17.

Boahen, K. and Alvarez-Icaza, R. (2012). Inferior oilve mirrors joint dynamics to implement an inverse controller. *Biological Cybernetics*.

Braitenberg, V. (1983). The cerebellum revisited. *Journal of Theoretical Neurobiology*, 2:237–241.

Braitenberg, V. (1987). The cerebellum and the physics of movement: Some speculations. pages 193–208.

Brette, R. and Gerstner, W. (2005). Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity. *J Neurophysiol*, 94.

Casellato, C., Antonietti, A., Garrido, J. A., Carrillo, R. R., Luque, N. R., Ros, E., Pedrocchi, A., and D'Angelo, E. (2014). Adaptive robotic control driven by a versatile spiking cerebellar network. *PLOS ONE*, 9:1–17.

Dayan, P. and Abbott, L. (2005). *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press.

De Schutter, E. and Bower, J. (1994). An active membrane model of the cerebellar purkinje cell i. simulation of current clamps in slice. 71:375–400.

Druol, D. L., Koibuchi, N., Manto, M., Molinari, M., Schmahmann, J. D., and Shen, Y. (2016). *Essentials of Cerebellum and Cerebellar Disorders - A Primer For Graduate Students*. Springer.

Eccles, J., Ito, M., and Szentagothai, J. (2013). *The Cerebellum as a Neuronal Machine*. Springer Berlin Heidelberg.

Goodman, D. and Brette, R. (2008). Brian: a simulator for spiking neural networks in Python. *Frontiers in neuroinformatics*, 2(5).

Herzfeld, D., Kojima, Y., Soetedjo, R., and Shadmehr, R. (2015). Encoding of action by the purkinje cells of the cerebellum. 526:439–442.

Higgins, D. C. (1986). The cerebellum and initiation of movement: The stretch reflex.

Ito, M. (1970). Neurophysiological aspects of the cerebellar motor control system. *International Journal of Neurology*, 7:62–176.

Kandel, E., Schwartz, J., and Jessell, T. (2000a). *Principles of Neural Science, Fourth Edition*. McGraw-Hill Companies,Incorporated.

Kandel, E. C., Schwartz, J. H., and Jessell, T. (2000b). *Principles of neural science*. Elsevier, 4th edition.

Kawato, M. and Gomi, H. (1992). A computational model of four regions of the cerebellum based on feedback-error learning. *Biological Cybernetics*, 68(2):95–103.

Lang, E., Sugihara, I., and Llins, R. (1996). GABAergic modulation of complex spike activity by the cerebellar nucleoolivary pathway in the rat. *J. Neurophysiol.*, 76.

Lang, E. J., Apps, R., Bengtsson, F., and Cerminara, N. L. (2017). The roles of the olivocerebellar pathway in motor learning and motor control. a consensus paper. *The Cerebellum*, 16(1):230–252.

Lefler, Y., Torben-Nielsen, B., and Yarom, Y. (2013). Oscillatory activity, phase differences, and phase resetting in the inferior olivary nucleus. *Frontiers in Systems Neuroscience*, 7:22.

Llinás, R. and Yarom, Y. (1981). Electrophysiology of mammalian inferior olivary neurons in vitro. different types of voltage-dependent ionic conductances. 315:549–67.

Llinás, R. and Yarom, Y. (1986). Oscillatory properties of guinea pig inferior olivary neurons and their pharmacology modulation: an in-vitro study. *The Journal of physiology*, 376.

Llinás, R. R. and Negrello, M. N. (2015). Cerebellum. *Scholarpedia*, 10(1):4606. revision #148143.

Luque, N. R., Naveros, F., Carrillo, R. R., Ros, E., and Arleo, A. (2018). Spike burst-pause dynamics of purkinje cells regulate sensorimotor adaptation. *bioRxiv*.

Marieb, E. N. and Hoehn, K. (2013). *Human Anatomy & Physiologoy*. 9^{th} edition.

Marr, D. and Thach, W. T. (1991). *A Theory of Cerebellar Cortex*, pages 11–50. Birkhäuser Boston, Boston, MA.

Negrello, M. and Schutter, E. D. (2016). *Models of the Cortico-cerebellar System*. Springer Science.

Person, A. and Raman, I. (2012). Purkinje neuron synchrony elicits time-locked spiking in the cerebellar nuclei. *Nature*, 481.

Schweighofer, N., Doya, K., and Kawato, M. (1999). Electrophysiological properties of inferior olive neurons: A compartmental model. *The American Physiological Society*.

Sotelo, C., Llins, R., and Baker, R. (1974). Serotonin modulation of inferior olivary oscillations and synchronicity: a multiple-electrode study in the rat cerebellum. *EuR J. Neurosci.*, 7.

Steuber, V. (2016). Chapter 5 - modeling the generation of cerebellar nuclear spike output. In Heck, D. H., editor, *The Neuronal Codes of the Cerebellum*, pages 117 – 133. Academic Press, San Diego.

Steuber, V., Mittmann, W., Hoebeek, F., Angus Silver, R., De Zeeuw, C., Hausser, M., and De Schutter, E. (2007). Cerebellar ltd and pattern recognition by purkinje cells. 54:121–36.

Steuber, V., Schultheiss, N., Silver, R. A., Schutter, E. D., and Jaeger, D. (2011). Determinants of synaptic integration and heterogeneity in rebound firing explored with data-driven models of deep cerebellar nucleus cells. *J. Comput. Neurosc.*, 30.

Tang, T., Suh, C. Y., Blenkinsop, T. A., and Lang, E. J. (2016). Synchrony is key: Complex spike inhibition of the deep cerebellar nuclei. *The Cerebellum*, 15(1):10–13.

II

An Introduction Spiking Neuron Models on the *Brian2* Simulator

E. M. Fernández Santoro

February 28, 2019

Abstract

Morphologically identical neurons may react differently to the same synaptic input. This is due to the relationship between the electrophisiology, bifurcations and computational properties of neurons. Understanding that cells can experience distinct bifurcations, and thus, have different neurocomputational properties has allowed computational neuroscientists to develop a variety of neuronal models. The majority of such models have been developed using different simulators and need specific languages to reproduce them. The Brian2 simulator has been developed as an attempt to bridge the gap between models and mainstream simulation languages. This work introduces Brian2 codes for four benchmark spiking neuron models: the Integrate-and-Fire, Leaky Integrate-and-Fire, Hodgkin-Huxley and Adaptive Exponential Integrate-and-Fire models.

Contents

List of Abbreviations

List of Functions

1 Introduction

Neurons have historically been divided into excitatory and inhibitory types, and their electrophysiological properties were neglected (Kandel, Schwartz, and Jessell, 2000). In some cases these properties were taken to be identical to those of the Hodgkin-Huxley's squid axon (Izhikevich, 2007). Nevertheless, neurons with identical morphological features and with the same synaptic input may respond differently. This is due to their electrophysiological properties (Thompson, 1993). In fact, even if two neurons have the same morphology as well as the same electrophysiological properties, they may still respond differently to the same synaptic input due to the cells different bifurcation dynamics (Shepherd, 2004). Hence, the electrophysiology of individual neurons as well their dynamical properties are of paramout importance for the understanding of neurons and neuroscience as a whole (Dayan and Abbott, 2005). Through the development of biological neuron modes, current neuroscience research is aimed at the understanding of a cell's intrinsic neurocomputational properties (Purves et al., 2008).

The properties of neurons can be described mathematically and such biological neuron models are used to explain the underlying mechanisms in the nervous system. Two categories of neuron models are distinguished: electrical input-output membrane voltage models, and pharmacological input neuron models (Izhikevich, 2007). Whereas the latter category is out of the scope of this work; the former type of models are able to predict the output membrane potential as a function of an input electrical stimulation. The various models in this category differ in the level of detail as well as in the relationship between input current and output voltage (Purves et al., 2008). As a result of the difficulty in separating the intrinsic properties of a single neuron from a measurement, as well as the high number of experimental settings, there is a very high number of spiking neuron models.

This work presents the basic knowledge needed to simulate four benchmark spiking neuron models in the Brian simulator on python. Ch. 2 presents an introduction to the Brian simulator. Subsequently, Ch. 3, 4, 5, and 6 present the four models and how they are simulated on Brian. Finally, Ch. 7 explains how synapses are modelled in this simulator.

2 The Brian Simulator

Romain Brette and Dan Goodman developed the Brian simulator for spiking neural networks as a response to the lack of hegemony between different softwares used in neural network simulation (Goodman and Brette, 2008). As each of these softwares require learning different scripting languages. Brian presents the following advantages:

1. As it is written on Python, it offers a tighter integration with the various tools and libraries of Python, resulting in a lot of flexibility[1].
2. Differential equations can be defined at the highest level using standard mathematical notation.
3. For linear differential equations, exact updates are used while for non linear differential equations, Euler and exponential Euler methods are used.
4. Contrary to other simulators, Brian is unit consistent. This reduces errors when modeling.
5. Further features such as network connectivity can be easily controlled and offers a lot of flexibility (all-to all random connectivity, specific connectivity, delays, synaptic weight functions, among others).

2.1 An Example to Get Started with Brian

The following section is a step-by-step guide of how to get started Brian and model a simple neuron[2]. In this example the neuron is modelled by the following differential equation:

$$\frac{dV_m(t)}{dt} = \frac{I(t)}{C_m} \tag{1}$$

Where, V_m is the membrane potential, C_m is the membrane capacitance and I is the current input.

2.1.1 Importing the Brian Toolbox

First, every Brian script in Python begins by importing the Brian toolbox. Other toolboxes can also be imported, such as *matplotlib* for plotting tools and *numpy* to help for mathematical computations. After importing the libraries, it is useful to invoke the *start_scope* function as it starts a new scope for the magic functions.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
```

Listing 1: Brian Toolbox

[1]While these are valid for both versions, this work concentrates on the second version of Brian.
[2]More information can be found in https://brian2.readthedocs.io/en/latest/index.html#

2.1.2 Defining Parameters and Dealing with Units

Second, the parameters of the neurons are defined with the correct units. Brian accepts all basic SI units accompanied with all standard prefixes.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
```

Listing 2: Units

2.1.3 Defining an Equation

Equations are defined using standard mathematical notations; units also have to be defined as shown below. In this example, the voltage of the membrane is being computed, thus the unit is in volts. The input for this example is the current I and thus, is defined as shown below (with unit amp/m^2).

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
eqs = '''
    dV/dt = I/Cm : volt
    I : amp*meter**-2
'''
```

Listing 3: Defining an Equation

2.1.4 Creating a Neuron

Neuron models are generated using the *NeuronGroup* function, which requires the specification of the number of neurons N, the neuron equation *eqs* as well as the integration method.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
eqs = '''
    dV/dt = I/Cm : volt
    I : amp*meter**-2
'''
N = 1
Neuron = NeuronGroup(N, eqs, method = 'euler')
```

Listing 4: Creating a Neuron

2.1.5 Recording

At this point, the objective of the simulation must be specified through the *StateMonitor* function. For this example, both the membrane potential V and the input current I are measured.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
eqs = '''
    dV/dt = I/Cm : volt
    I : amp*meter**-2
'''
N = 1
Neuron = NeuronGroup(N, eqs, method = 'euler')
Neuron_statemon = StateMonitor(Neuron, variables=['V','I'], record =
    True)
```

Listing 5: Recording

2.1.6 Simulating a Neuron

To start the simulation, the runtime —100 milliseconds in this example— and the input variables —applied current in this example— are defined. Then, the results can be plotted. Fig. 1 shows the response of the neuron. As expected from Eq.(1), the membrane potential increases linearly with a constant current.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
eqs = '''
    dV/dt = I/Cm : volt
    I : amp*meter**-2
'''
N = 1
Neuron = NeuronGroup(N, eqs, method = 'euler')
Neuron_statemon = StateMonitor(Neuron, variables=['V','I'], record =
    True)
runtime = 100*ms
Neuron.I = 1*uamp*cm**-2
run(runtime)
import matplotlib.gridspec as gridspec
fig = plt.figure(figsize=(20, 8))
gs1 = gridspec.GridSpec(10, 5)
gs1.update(left=0.01, right=0.5, wspace=9)
ax1 = plt.subplot(gs1[:-1, :])
ax1.plot(Neuron_statemon.t/ms, Neuron_statemon.V[0]/mvolt,'C1',lw='2
   ')
ylabel('V mV')
legend();
ax2 = plt.subplot(gs1[-1, :])
ax2.plot(Neuron_statemon.t/ms, Neuron_statemon.I[0]/(uamp/cm**2),'C2
   ',lw='2')
```

```
xlabel('t (msec)')
ylabel('I uamp/cm^2')
legend();
show()
```

Listing 6: Simulating a Neuron

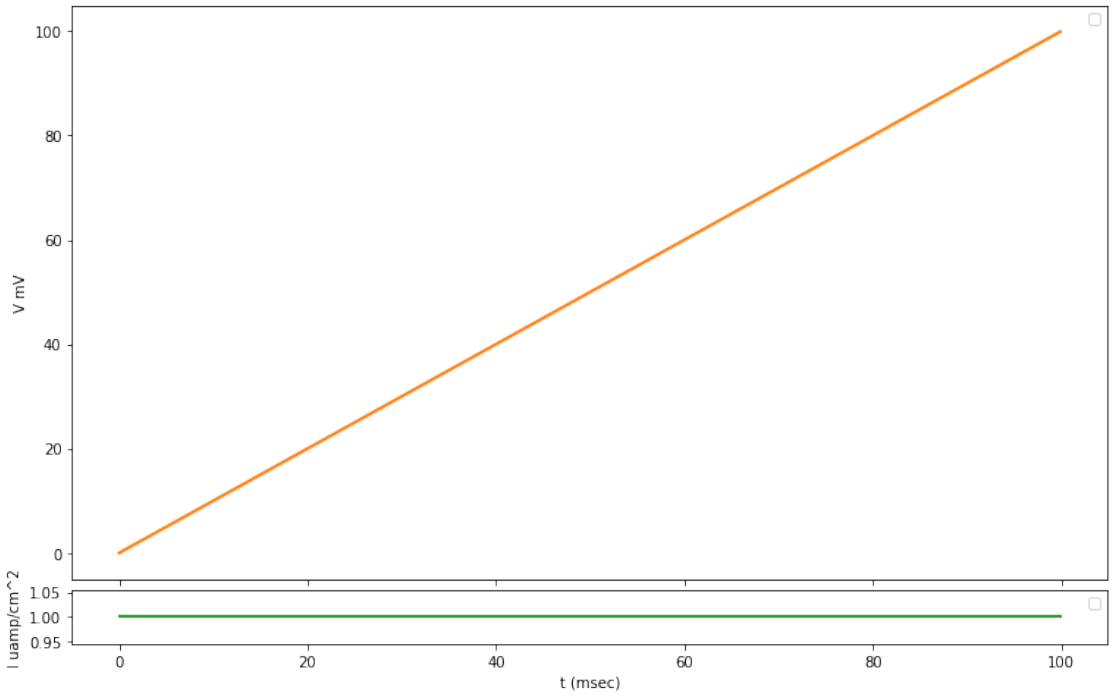


Figure 1: Response to a constant input current of 1 μamp/cm^2

2.1.7 Simulating Different Currents Over Time

To see the response of the membrane potential to a varying current input, it is only needed to change the *Neuron.I* value and run the code again. For instance, a 50 millisecond current pulse can be created as shown bellow.

```
runtime = 100*ms
Neuron.I = 1*uamp*cm**-2
run(runtime)
Neuron.I = 1*uamp*cm**-2
run(50*ms)
Neuron.I = 0*uamp*cm**-2
run(100*ms)
```

Listing 7: Varying Input

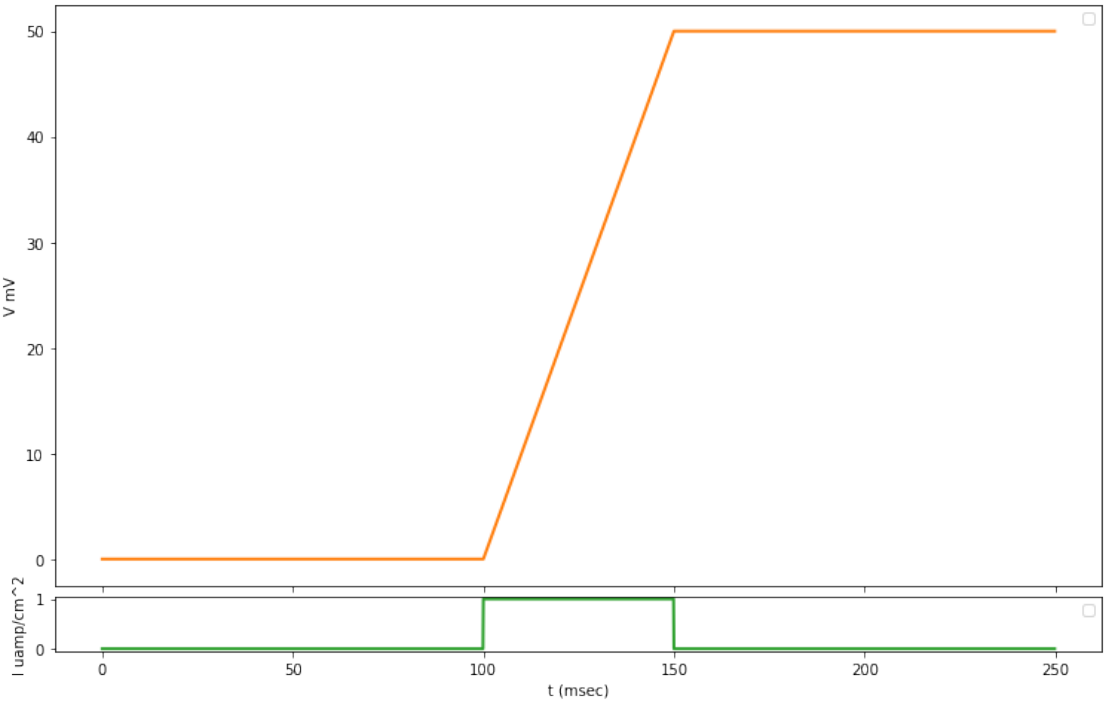


Figure 2: Response to a 50 ms current pulse of 1 μamp/cm^2

By implementing a threshold potential at which a spike occurs, the neuron model in this example becomes a spiking neuron model known as the Integrate-and-Fire model. It is the basis for present spiking neuron models which will be discussed in the next chapter.

3 Integrate-and-Fire Model

In 1907, Louis Lapicque was able to mathematically represent a neuron (Eq.(2)) by taking the time derivative of the law of capacitance ($Q = CV$) (Izhikevich, 2007; Gerstner et al., 2014).

$$C_m \frac{dV_m(t)}{dt} = I(t) \tag{2}$$

where, C_m is the membrane capacitance and V_m is the membrane potential.

It is clear that this is very similar to the example shown in Ch. 2. However, the Integrate-and-Fire (I&F) model increases its membrane potential with time, as a result of an input current, until a threshold potential is reached V_T (Dayan and Abbott, 2005; Kandel, Schwartz, and Jessell, 2000). When this threshold is reached a spike occurs and the membrane voltage is reset to the resting potential V_R. After the voltage is reset the model continues.

3.1 Implementing the Integrate-and-Fire Model on the Brian2 Simulator

Starting from the example in the previous chapter (Ch. 2), two parameters need to be added. Namely, a threshold potential and a resting potential.

3.1.1 Threshold

While the threshold parameter is implemented as a string in the code, it is a condition in the *NeuronGroup* function as shown bellow.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
Vt = -40*mvolt      #Threshold
eqs = '''
    dV/dt = I/Cm : volt
    I : amp*meter**-2
'''
N = 1
Neuron = NeuronGroup(N, eqs, threshold='V>Vt', method = 'euler')
Neuron_statemon = StateMonitor(Neuron, variables=['V','I'], record =
    True)
runtime = 100*ms
Neuron.I = 1*uamp*cm**-2
Neuron.V = -70*mvolt
run(runtime)
```

Listing 8: Integrate-and-Fire Model

The reader can notice that the threshold as well as the *Neuron.V* values are negative. This should be so in a neuron as cells have negative resting potentials.

3.1.2 Reset

Analogously to the threshold a reset parameter is given as shown bellow.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
Vt = -40*mvolt      #Threshold
Vr = -65*mvolt      #Reset Voltage
eqs = '''
    dV/dt = I/Cm : volt
    I : amp*meter**-2
'''
N = 1
Neuron = NeuronGroup(N, eqs, threshold='V>Vt', reset='V=Vr', method
    = 'euler')
Neuron_statemon = StateMonitor(Neuron, variables=['V','I'], record =
    True)

runtime = 100*ms
Neuron.V = -70*mvolt
Neuron.I = 1*uamp*cm**-2
run(runtime)
```

Listing 9: Integrate-and-Fire Model

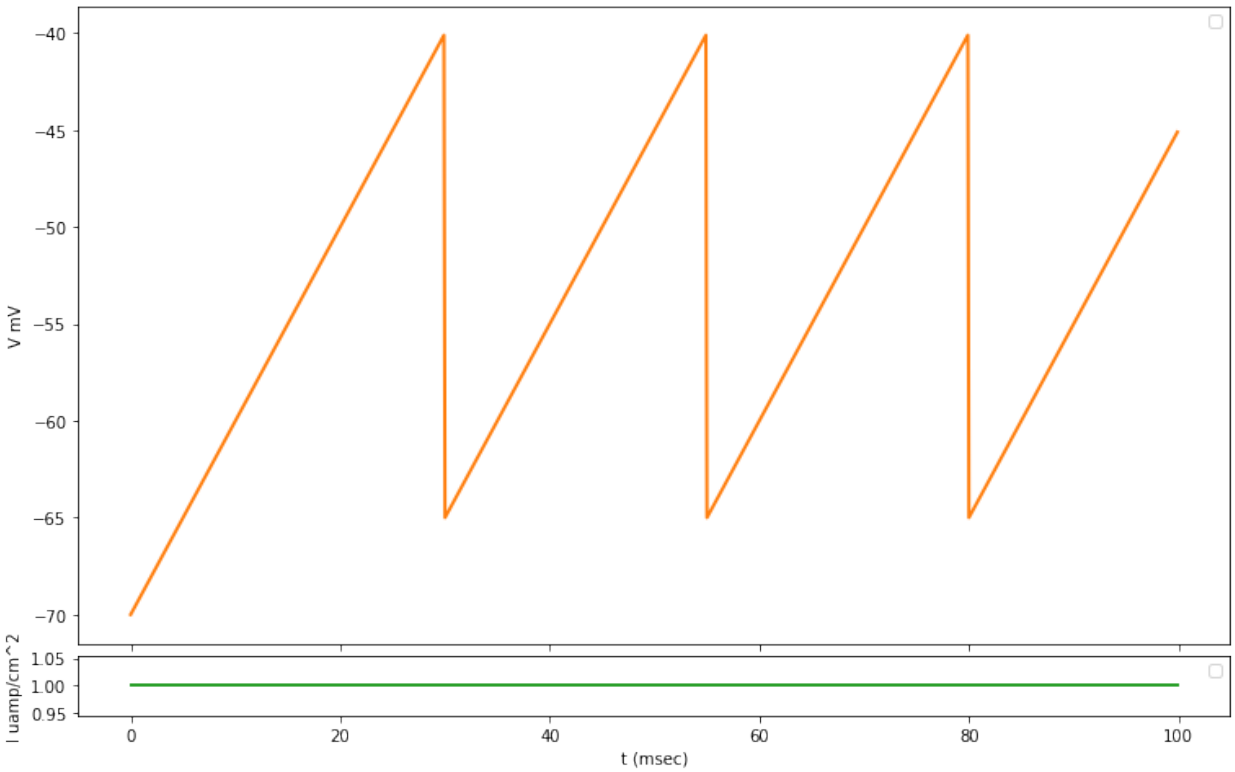


Figure 3: Integrate-and-Fire Model

3.2 Limitations of the Integrate-and-Fire Model

This model can be improved by including a refractory period that prevents the neuron to fire for a certain amount of time after the spike (Dayan and Abbott, 2005). This period can be manually added to the *NeuronGroup* function as follows:

```
Neuron = NeuronGroup(N, eqs, threshold='V>Vt', reset='V=Vr',
   refractory = 2*ms, method = 'euler')
```

Nevertheless, the I&F model has a big flaw: it has no time-dependent memory (Gerstner et al., 2014). This means that if this neuron receives input current so that the membrane voltage does not reach the threshold, it will keep this voltage forever until the next spike (Fig. 4). To solve this memory problem, the leaky integrate-and-fire model was developed (Izhikevich, 2007). This is the object of the next chapter.

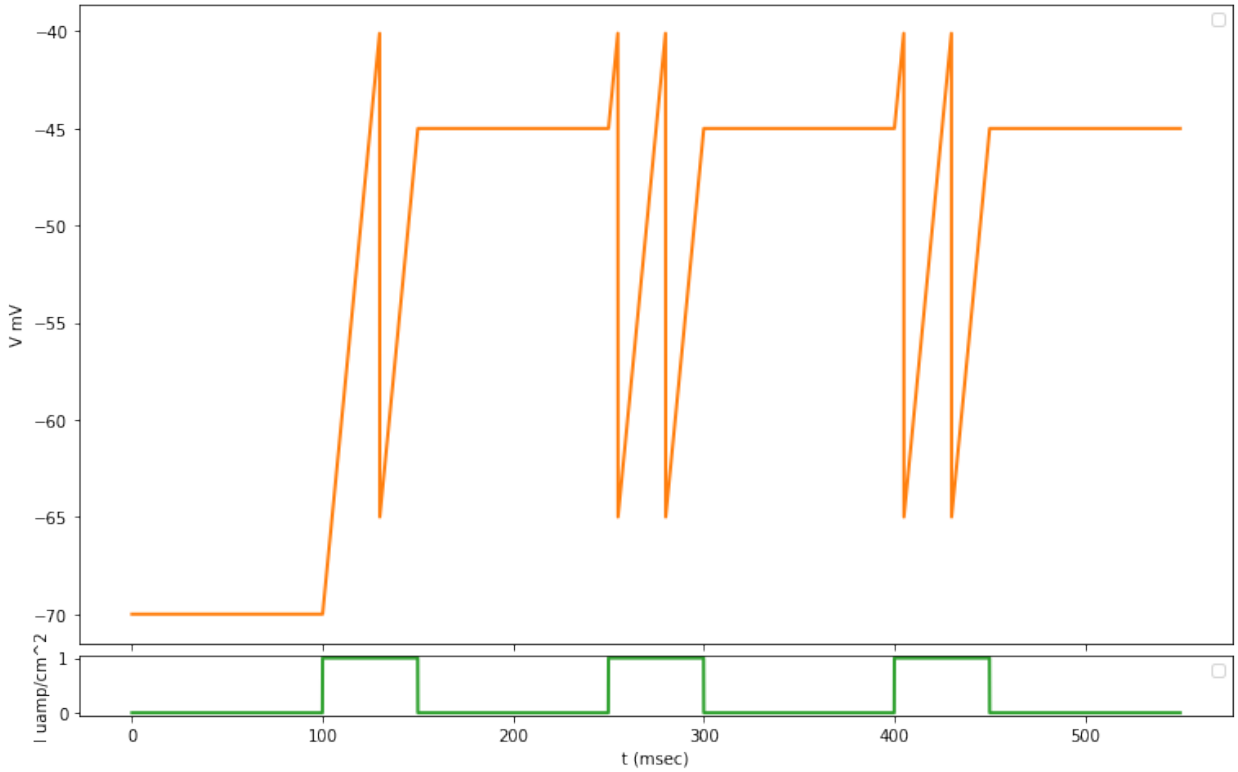


Figure 4: Integrate-and-Fire Model: keep this voltage forever until the next spike

4 Leaky Integrate-and-Fire Model

This model reflects the diffusion of ions occurring through the membrane while the cell is not in equilibrium (Dayan and Abbott, 2005). This is achieved by introducing a leak term $I_l(t) = \frac{V_m(t)}{R_m}$. The introduction of this leak term causes the cell to fire only when $I(t)$ is bigger than a new threshold $I_{th} = \frac{V_{th}}{R_m}$ (Izhikevich, 2007). Hence, if the current does not reach this new threshold current, the leak term will leak out any change in potential (Fig. 5).

$$C_m \frac{dV_m(t)}{dt} = I(t) - \frac{V_m(t)}{R_m} \tag{3}$$

where R_m is the membrane resistance.

4.1 Implementing the Leaky I&F Model on the Brian2 Simulator

The membrane resistance parameter needs to be defined as well as the new leak current I_l.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
Cm = 1.0*uF/cm**2  #The membrane capacitance
Vt = -40*mvolt      #Threshold
Vr = -65*mvolt      #Reset Voltage
Rm =  10*Mohm*cm**2       #Membrane resistance
eqs = '''
    dV/dt = (I-I_l)/Cm : volt
    I_l = V/Rm : amp*meter**-2
    I : amp*meter**-2
'''
N = 1
Neuron = NeuronGroup(N, eqs, threshold='V>Vt', reset='V=Vr',
   refractory = 2*ms, method = 'euler')
Neuron_statemon = StateMonitor(Neuron, variables=['V','I'], record =
    True)
```

Listing 10: Leaky Integrate and Fire Model

4.2 Limitations of the Leaky Integrate-and-Fire Model

The models presented in Ch. 3 and 4 are highly simplified as many aspects from neuronal dynamics and physiology are neglected (Connors and Gutnick, 1990). Neurons have different modes of spiking (Izhikevich, 2007). They may be regular or fast spiking as well as bursting and stutter spiking (Connors and Gutnick, 1990; Dayan and Abbott, 2005). However, models that do not have memory beyond the most recent spike are not able to describe bursting modes. Moreover, adaptation is not captured by the model. As the voltage is reset following each spike, there is no memory of the spikes before the most recent one (Connors and Gutnick, 1990; Izhikevich, 2007). Another limitation is that, independently of the state of the postsynaptic neuron, the input is integrated linearly (Gerstner et al., 2014). Finally, post-inhibitory rebound (rebound spikes) behavior is also not reproduced by this model (Dayan and Abbott, 2005).

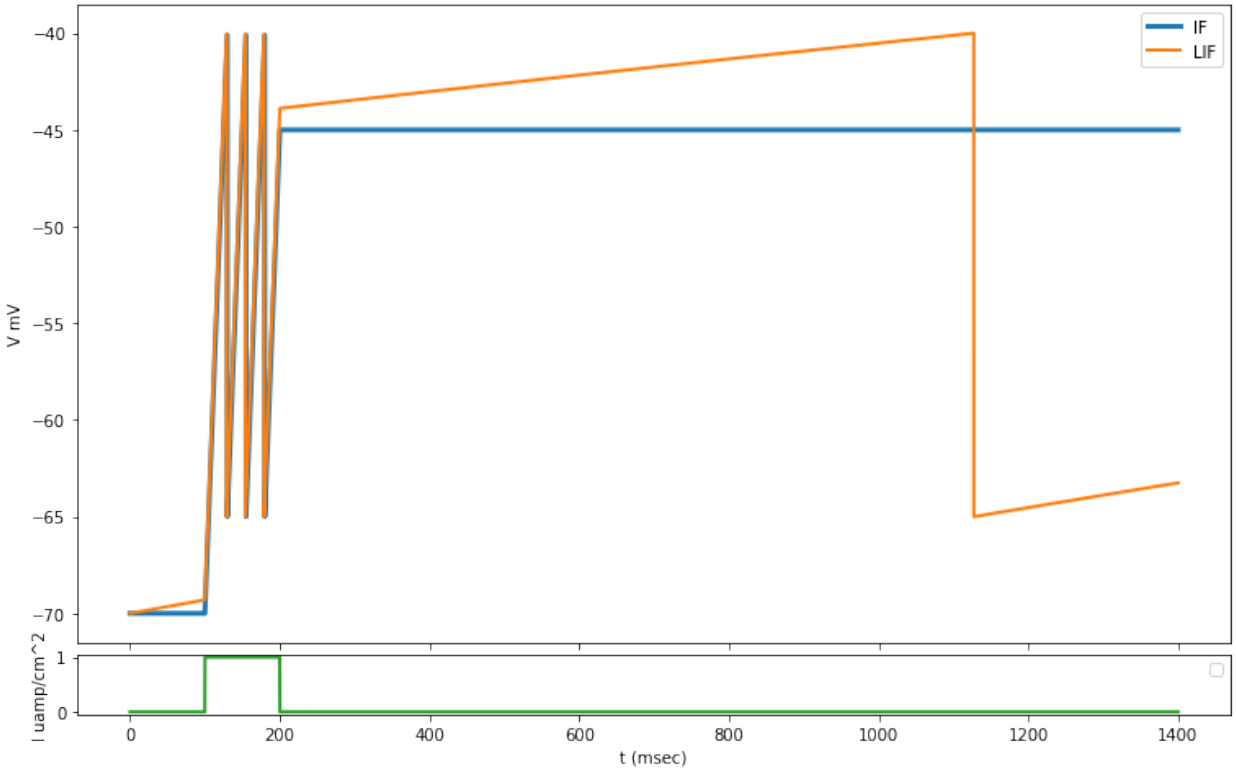


Figure 5: Leaky Integrate-and-Fire Model and Integrate-and-Fire Model

5 Hodgkin-Huxley Model

The equation developed by Lapicque (Eq.(2)) is the mathematical description of current flowing through the lipid bilayer (Dayan and Abbott, 2005). Furthermore, when looking at the different ion channels of the membrane, the current passing through the i^{th} ion channel can be described as follows:

$$I_i = g_n(V_m - V_i) \tag{4}$$

where, V_m is the membrane potential, g_n is the channel conductance and V_i is the reversal potential of the ion channel (Izhikevich, 2007). Furthermore, some ion channels have a voltage gating mechanism and thus, the channel conductance in these cases are a function of both time and voltage (Kandel, Schwartz, and Jessell, 2000). To describe these mechanisms Hodgkin and Huxley, through a series of voltage clamp experiments, developed a four dimensional model (HH Model) - described by a set of four ordinary differential equations (ODE) (Hodgkin and Huxley, 1952).

$$C_m \frac{dV_m}{dt} = \overline{g}_K n^4 (V_m - V_K) + \overline{g}_{Na} m^3 h (V_m - V_{Na}) + \overline{g}_l (V_m - V_l) - I \tag{5}$$

where $\overline{g}_i$ is the maximal conductance, n and m are activation variables (potassium and sodium channels respectively) and h is the inactivation variable for the sodium channel. The activation and inactivation variables are dimensionless quantities between 0 and 1 that represent the channel conduction as a function of voltage (Gerstner et al., 2014). In other words, these variables represent how many channels are opened. Thus, when multiplied with the maximal conductance, the voltage-gated channel conductance is represented. While Eq.(5) shows the equation for the total membrane potential, the ordinary differential equations for the activation and inactivation variables are shown in Eq.(6):

$$\begin{aligned} \frac{dn}{dt} &= \alpha_n(V_m)(1-n) - \beta_n(V_m)n \\ \frac{dm}{dt} &= \alpha_m(V_m)(1-m) - \beta_m(V_m)m \\ \frac{dh}{dt} &= \alpha_h(V_m)(1-h) - \beta_h(V_m)h \end{aligned} \tag{6}$$

While in the original paper by Hodgkin and Huxley (Hodgkin and Huxley, 1952), the functions of α and β are given, it is out of the scope of this project. Modern models use similar definitions for the activation and inactivation variables as newer experimental results have allowed to improve the functions for α and β.

5.1 Implementing the Hodgking-Huxley Model on the Brian2 Simulator

The parameters can be found in the original paper (Hodgkin and Huxley, 1952)[3]. An example of parameters is shown bellow. When it is not defined the defaultclock is 0.25 milliseconds, in this example a 0.025 milliseconds.

```
from brian2 import*
import matplotlib.pyplot as plt
import brian2.numpy_ as np
start_scope()
dtt = 0.025*msecond
Cm = 281 * pF
gl = 30 * nS
El = -70.6 * mV
EK = -90*mV
ENa = 50*mV
g_na = 40.0*uS
g_kd =  6.0*uS
VT = -50.4 * mV
```

Listing 11: Hodking-Huxley Parameters

Eq.(5) is implemented as shown below:

```
eqs_V = '''
dv/dt = (gl*(El-v) - g_na*(m*m*m)*h*(v-ENa) - g_kd*(n*n*n*n)*(v-EK)
   + I - Igap)/Cm : volt
I : amp
Igap : amp
Vcut : volt
'''
```

Listing 12: Main Equation

Eq.(6) is implemented as found in an example in the Brian2 Documentation.[4]

```
eqs_m = '''
dm/dt = 0.32*(mV**-1)*(13.*mV-v+VT)/
    (exp((13.*mV-v+VT)/(4.*mV))-1.)/ms*(1-m)-0.28*(mV**-1)*(v-VT
   -40.*mV)/
    (exp((v-VT-40.*mV)/(5.*mV))-1.)/ms*m : 1

'''
eqs_n = '''
dn/dt = 0.032*(mV**-1)*(15.*mV-v+VT)/
    (exp((15.*mV-v+VT)/(5.*mV))-1.)/ms*(1.-n)-.5*exp((10.*mV-v+VT)
   /(40.*mV))/ms*n : 1
'''
eqs_h = '''
dh/dt = 0.128*exp((17.*mV-v+VT)/(18.*mV))/ms*(1.-h)-4./(1+exp((40.*
   mV-v+VT)/(5.*mV)))/ms*h : 1
'''
```

Listing 13: Activation/Inactivation Dynacmis

The four equations are added to *eqs_HH* that is used to generate a neuron group. Finally, the neuron is simulated and plotted.

[3]Examples of this model can be found in https://brian2.readthedocs.io/en/latest/index.html#

[4]https://brian2.readthedocs.io/en/latest/examples/IF_curve_Hodgkin_Huxley.html

```
eqs_HH = eqs_V
eqs_HH += eqs_m
eqs_HH += eqs_n
eqs_HH += eqs_h
HH = NeuronGroup(N_cells, eqs_HH, threshold='v>Vcut', method='euler'
    ,dt=dtt)
HH.v = El
HH.Vcut = -40.4*mV
HH_Statemon = StateMonitor(HH, variables = ['v','I'], record=True,dt
    =dtt)
```

Listing 14: Create Neuron Group

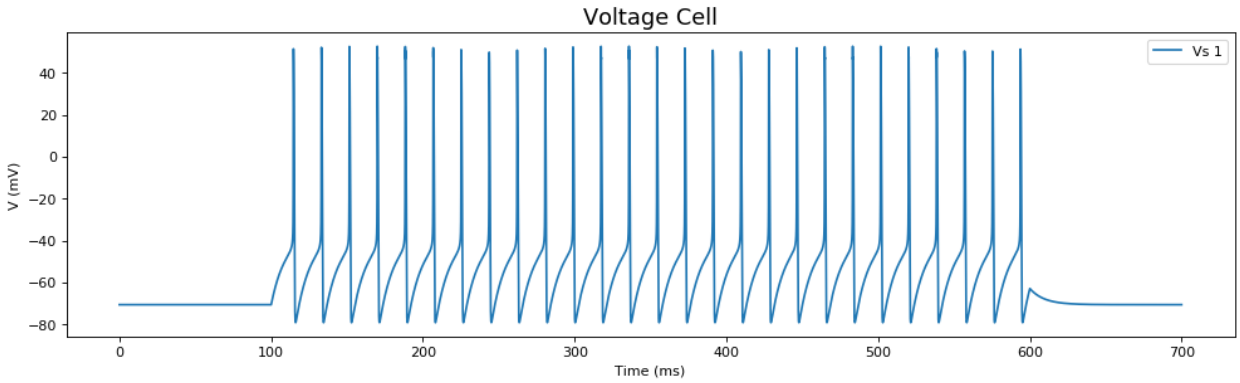


Figure 6: Response of Hodgkin-Huxley Neuron Model

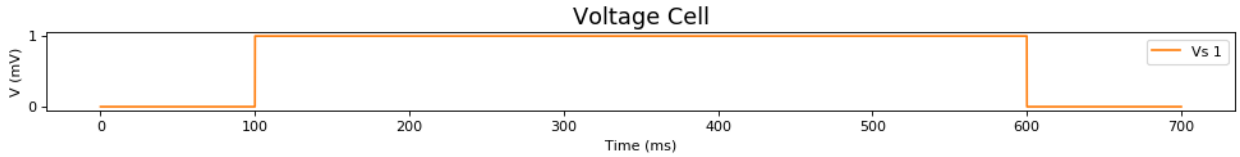


Figure 7: Applied Current

5.2 Limitations of the Hodgking-Huxley Model

The Hodgkin-Huxley model describes only three types of ion channels. However, this model can be extended to include other ion channel types (see Ch. 7)(Gerstner et al., 2014). Nevertheless, the Hodgkin-Huxley model is computationally expensive as there are 4 ODEs to be solved. For research that needs the basic response of a neuron and does not particularly need a very detailed neuron model, it is of particular interest to use a simpler model that is able to approximate closely the response of HH model. Such a model is found in the Adaptive Exponential Integrate-and-Fire spiking neuron model (AdEx) which is discussed in the next chapter.

6 Adaptive Exponential Integrate-and-Fire Model

This two dimensional model describes the dynamics of the membrane potential with a first ODE that includes an exponential voltage dependence and a second ODE to which voltage is coupled that describes adaptation of the neuron (Izhikevich, 2007).

$$\begin{aligned} \frac{dV}{dt} &= \frac{1}{C_m} g_L \cdot (E_L - V) + g_L \cdot \Delta_T \cdot e^{\left(\frac{V - V_T}{\Delta_T}\right)} + I_{gap} - w \\ \frac{dw}{dt} &= \frac{a\,(V - E_L) - w}{\tau_w} \end{aligned} \tag{7}$$

where Δ_T is the slope factor, V_T is the the threshold potential, w is the adaptation variable, a is the adaptation coupling factor and τ_w is the adaptation time constant.

The adaptation variable w has a decay time so that a memory-like mechanism is found. When a spike happens both the membrane potential and the adaptive variables are reset, however, while the first resets to the resting potential, the second decays until either reaching 0 or the next spike time (Dayan and Abbott, 2005). Furthermore, one can see that when constantly inject current without generating a spike, V_T is the maximum voltage that can be reached. The slope factor Δ_T is a quantification of the sharpness of the spike. This can be seen as the sharpness of the sodium activation curve if the activation time constant is ignored (Goodman and Brette, 2008). In fact, this model becomes a I&F model at the limit of zero slope factor (with a fixed threshold V_T).

6.1 Implementing the AdEx Model on the Brian2 Simulator

Implementing the parameters as found in the original paper (Goodman and Brette, 2008).

```
from brian2 import*
start_scope()
dtt = 0.025*msecond
N_PC = 1
C = 281 * pF
gL = 30 * nS
taum = C / gL
EL = -70.6 * mV
VT = -50.4 * mV
DeltaT = 2 * mV
Vcut = VT + 5 * DeltaT
tauw = 144*ms
a = 4*nS
b = 0.0805*nA
Vr = -70.6*mV
```

Listing 15: Example of Adaptive Exponential Integrate and Fire Model

Eq.(7) is implemented as shown below. Then, a neuron group is generated and, finally, the neuron is simulated and plotted.

```
eqs_PC = """
dv/dt = (gL*(EL - v) + gL*DeltaT*exp((v - VT)/DeltaT) I - w)/C :
   volt
```

```
dw/dt = (a*(v - EL) - w)/tauw : amp
I : amp
"""
PC = NeuronGroup(N_PC, model=eqs_PC, threshold='v>Vcut', reset="v=Vr
    ; w+=b", method='euler', name = 'PC',dt=dtt)
PC.v = EL
PC_statemon = StateMonitor(PC, ['v', 'Igap', 'I','w'], record=True)
PC_spikemon = SpikeMonitor(PC)#, 't', record=True)
```

Listing 16: Main Equation

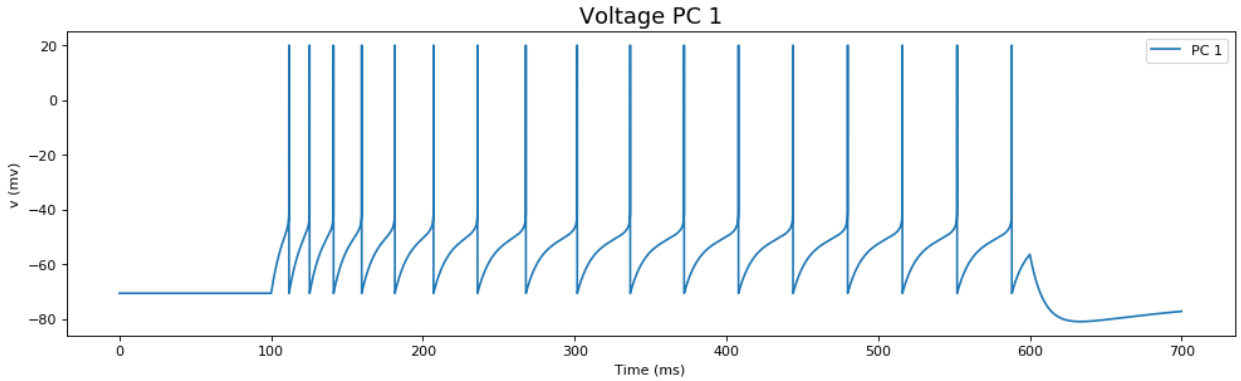


Figure 8: Response of Adaptive Exponential Neuron Model

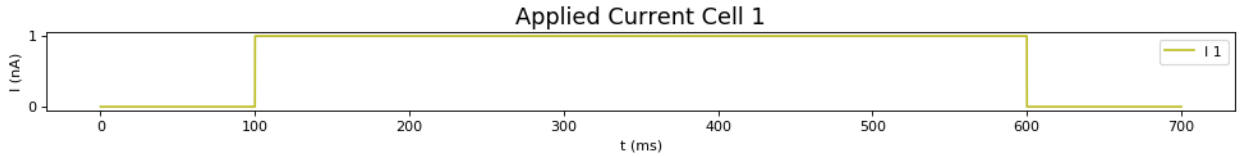


Figure 9: Applied Current

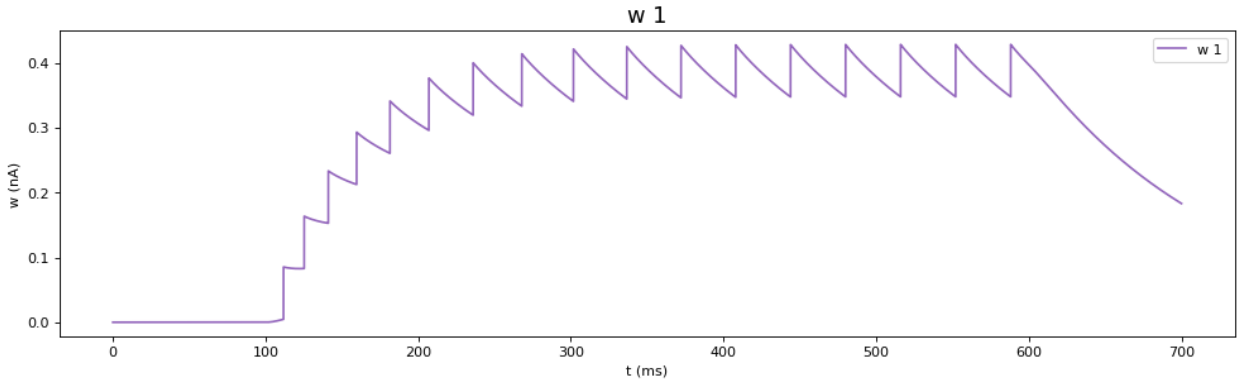


Figure 10: Adaptation Variable w

6.2 Comparing with Hodgkin-Huxley Model

When comparing to the Hodgkin-Huxley model, one sees that the exponential term describes the voltage dependent activation of the sodium channel (activation is assumed instantaneous). The AdEx model describes almost exactly the response of the HH model as shown in Fig. 11.

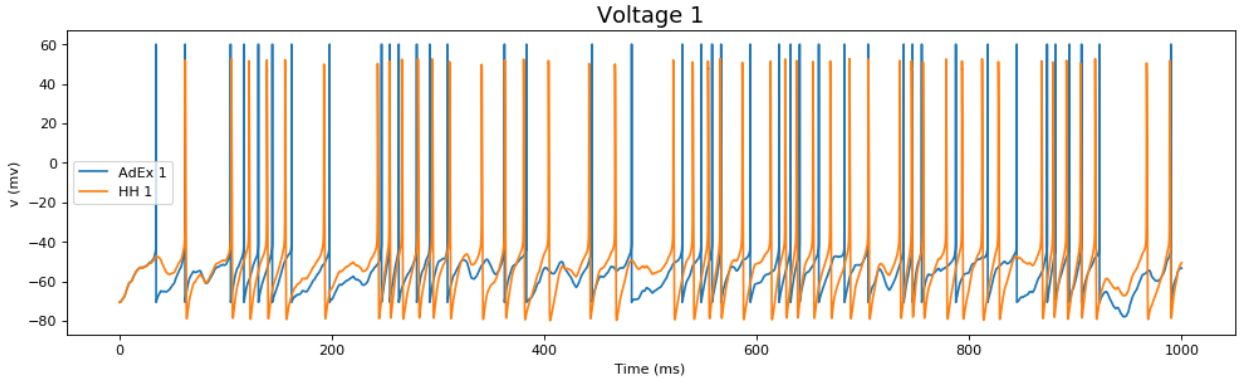


Figure 11: Comparing AdEx and Hodgkin-Huxley

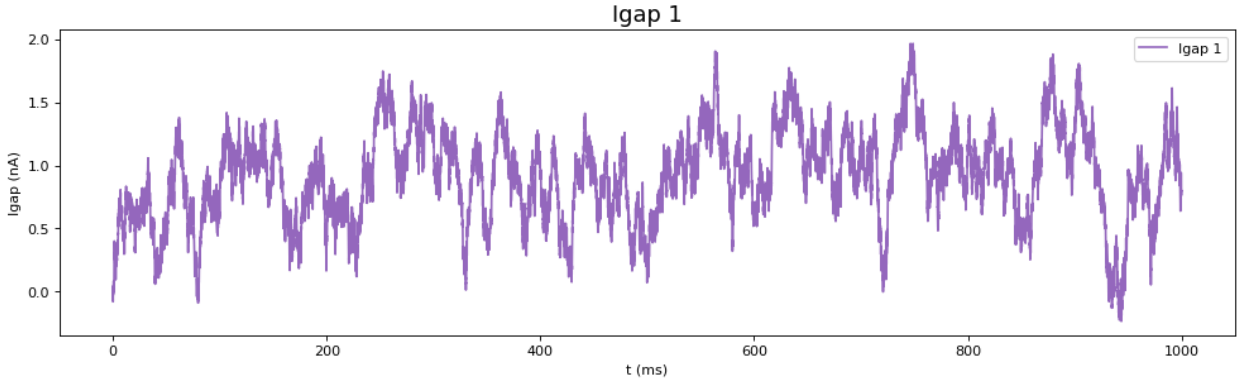


Figure 12: Applied Current

7 Modeling Synapses

In Brian2, the *Synapses* function is used to create synapses between different neuron groups. The following example uses a newly created neuron group as noise and the AdEx neuron group used in Ch. 6. First, a neuron group is created following that represents a current signal with stochastic noise using the Ornstein-Uhlenbeck stochastic process.

7.1 The Ornstein-Uhlenbeck Stochastic Process in Brian 2

The current created (Eq.(8)) is defined by a value I_0 which represents the maximum value that I should approach, and τ_{Noise} which represents how fast I approaches I_0 (this is seen in Fig. 13).

$$\frac{dI}{dt} = \frac{(I_0 - I)}{\tau_{Noise}} \tag{8}$$

```
N_noise = 1
tau_noise = 10*ms
I0 = 1.0 *nA

eqs_noise = '''
dI/dt = (I0 - I)/tau_noise : amp
'''
Noise = NeuronGroup(N_noise, eqs_noise, threshold = 'True', method='
    euler',name = 'Noise',dt=dtt)
Noise_statemon = StateMonitor(Noise, 'I', record=True,dt=dtt)
```

Listing 17: Random Noise Input

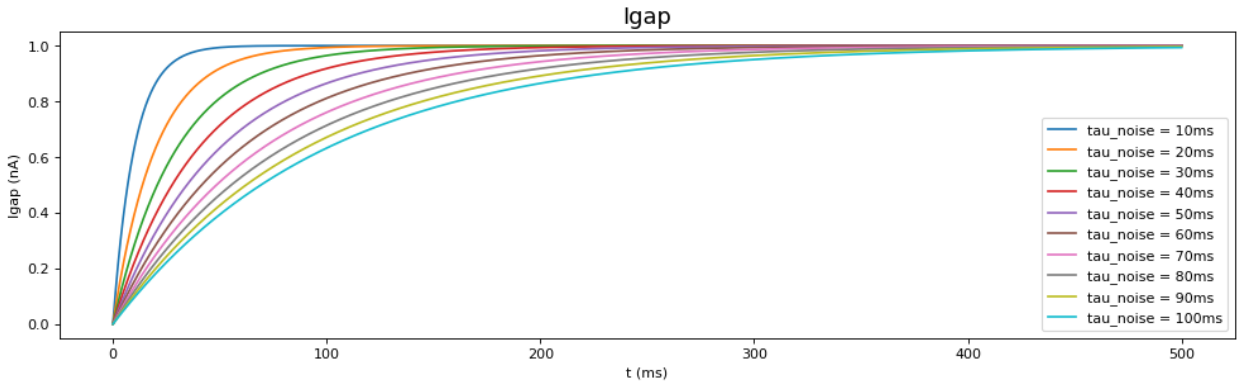


Figure 13: Effect of τ on Current Generated

The component from the OU stochastic process[5] $\frac{\sigma\xi}{\sqrt{\tau_{Noise}}}$ is added to the original noise (Eq.(9)). Two new parameters are found: ξ wich is a Gaussian random variable that scales with $\frac{1}{\sqrt{second}}$ and has mean 0 and standard deviation 1; and σ that represents how much offset or amplitude the signal has from its original position. The OU component is also affected by τ_{Noise}. The effect of τ_{Noise} and σ on the signal is shown in Fig. 14 and 15.

$$\frac{dI}{dt} = \frac{(I_0 - I)}{\tau_{Noise}} + \frac{\sigma\xi}{\sqrt{\tau_{Noise}}} \tag{9}$$

```
N_noise = 1
tau_noise = 10*ms
I0 = 1.0 *nA#amp
sigma = 0.5*nA#amp
eqs_noise = '''
dI/dt = (I0 - I)/tau_noise + sigma*xi*tau_noise**-0.5 : amp
'''
Noise = NeuronGroup(N_noise, eqs_noise, threshold = 'True', method='
    euler',name = 'Noise',dt=dtt)
Noise_statemon = StateMonitor(Noise, 'I', record=True,dt=dtt)
```

Listing 18: Random Noise Input

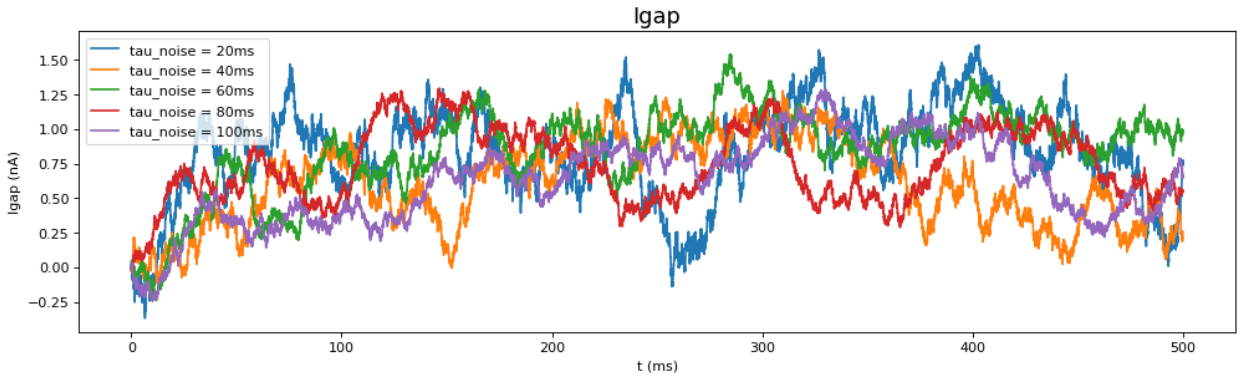


Figure 14: Effect of τ on Noise

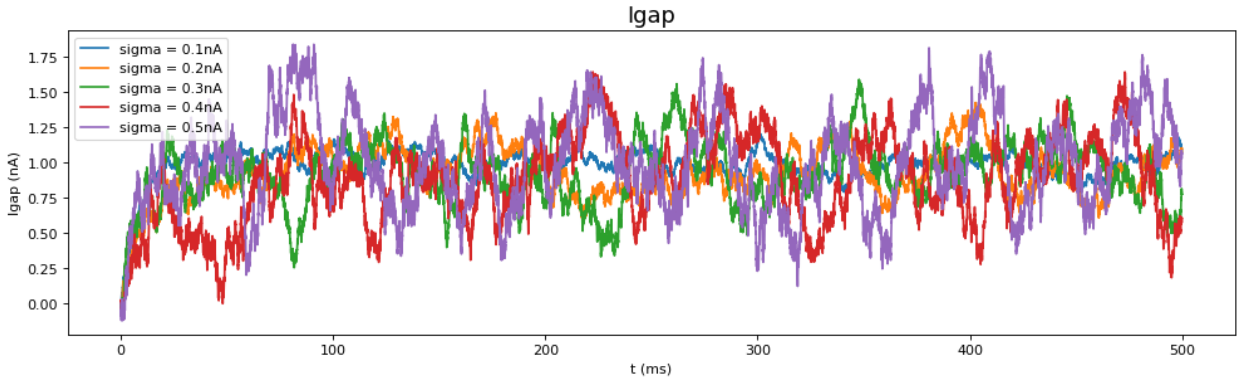


Figure 15: Effect of σ on Noise

[5]As found in https://brian2.readthedocs.io/en/latest/user/models.html

7.2 An Example to Get Started with Synapses

The synapse is created by using the *Synapses* function. Analogously to the *NeuronGroup* function, the equation of the synapse needs to be defined. The presynaptic neuron is defined first, followed by the postsynaptic neuron. In this case the presynaptic neuron is the Noise and the postsynaptic one is the AdEx (called PC in this example). This example simulates a random parallel fiber input to a Purkinje Cell (Fig. 16 and 17).

```
eqs_syn = '''
    weight : 1 # gap junction conductance
    Igap_post = weight* (I_pre) : amp (summed)
'''
S = Synapses(Noise, PC, eqs_syn,name = 'PC_Noise_Synapse',dt=dtt)
S.connect(j='i')
S.weight = '(j+2)*-1'
```

Listing 19: Creating the Synapse

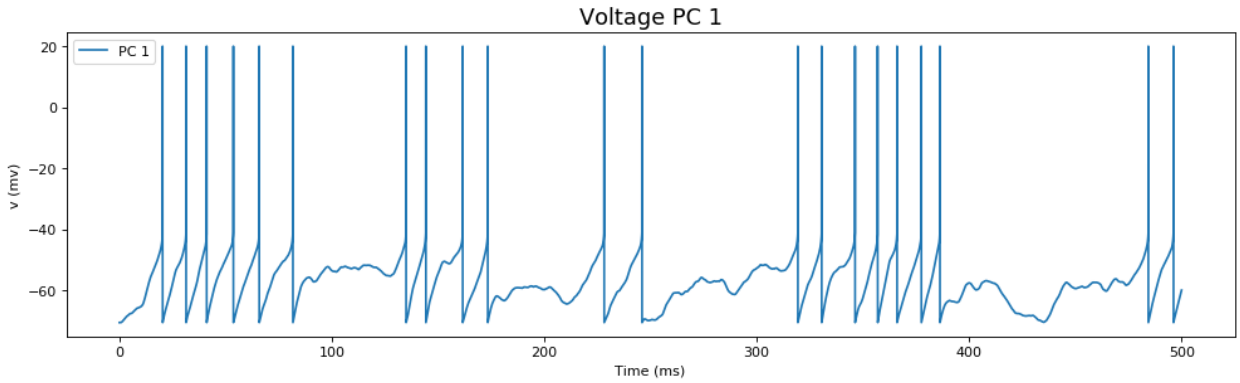


Figure 16: Response of Adaptive Exponential Neuron Model

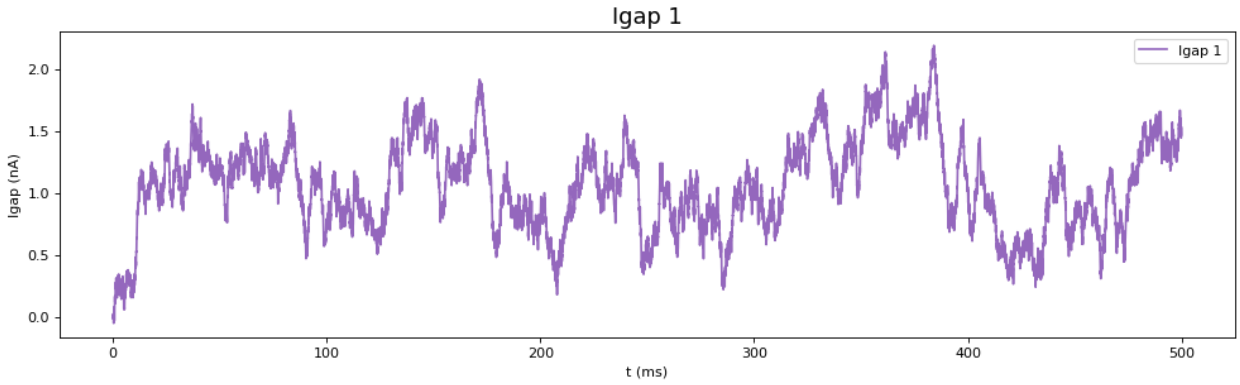


Figure 17: Applied Current

The connection between the neurons is made. The i neurons represent the i^{th} presynaptic neuron while the j represent the postsynaptic neurons. In this example, the connection is chosen to be $j =' i'$ which means that each presynaptic neuron will be connected to its postsynaptic counterpart in the same order ($i = 1$ with $j = 1$ and so on). The connectivity can be seen (Fig. 18) by defining the following function[6]:

```
def visualise_connectivity(S):
    Ns = len(S.source)
    Nt = len(S.target)
    figure(figsize=(10, 4))
    subplot(121)
    plot(zeros(Ns), arange(Ns), 'ok', ms=10)
    plot(ones(Nt), arange(Nt), 'ok', ms=10)
    for i, j in zip(S.i, S.j):
        plot([0, 1], [i, j], '-k')
    xticks([0, 1], ['Source', 'Target'])
    ylabel('Neuron index')
    xlim(-0.1, 1.1)
    ylim(-1, max(Ns, Nt))
    subplot(122)
    plot(S.i, S.j, 'ok')
    xlim(-1, Ns)
    ylim(-1, Nt)
    xlabel('Source neuron index')
    ylabel('Target neuron index')
```

Listing 20: Synapse Connectivity

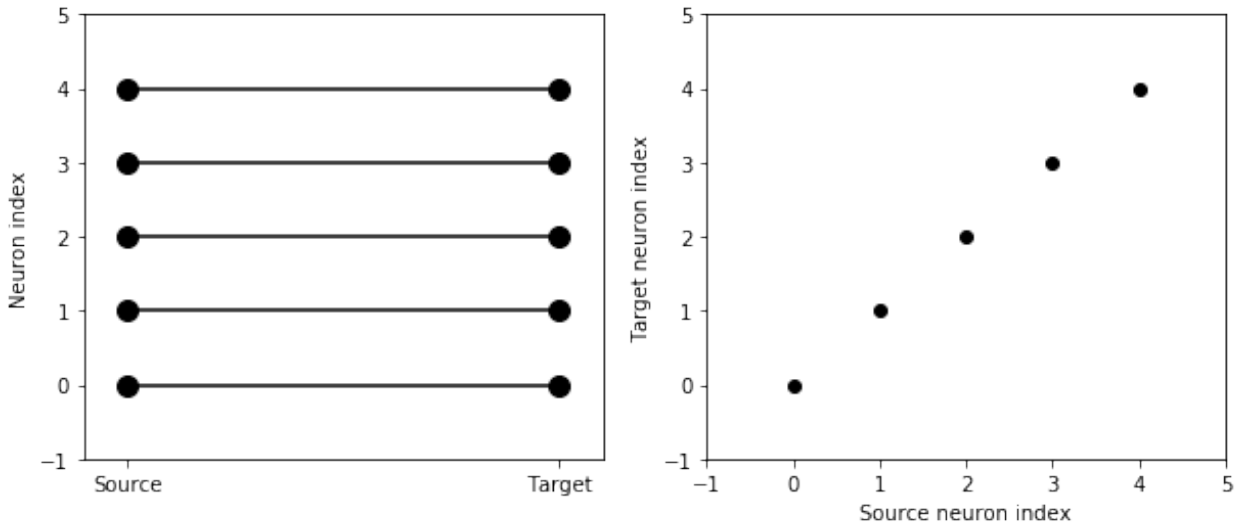


Figure 18: Connectivity of the Synapse

[6]As found in: https://brian2.readthedocs.io/en/latest/resources/tutorials/2-intro-to-brian-synapses.html

The *weight* that is defined represents the synaptic weight. In other words, it is the amount of presynaptic signal that is given to the j^{th} postsynaptic neuron. The effect of the synaptic weight on the postsynaptic neuron is shown in Fig. 19.

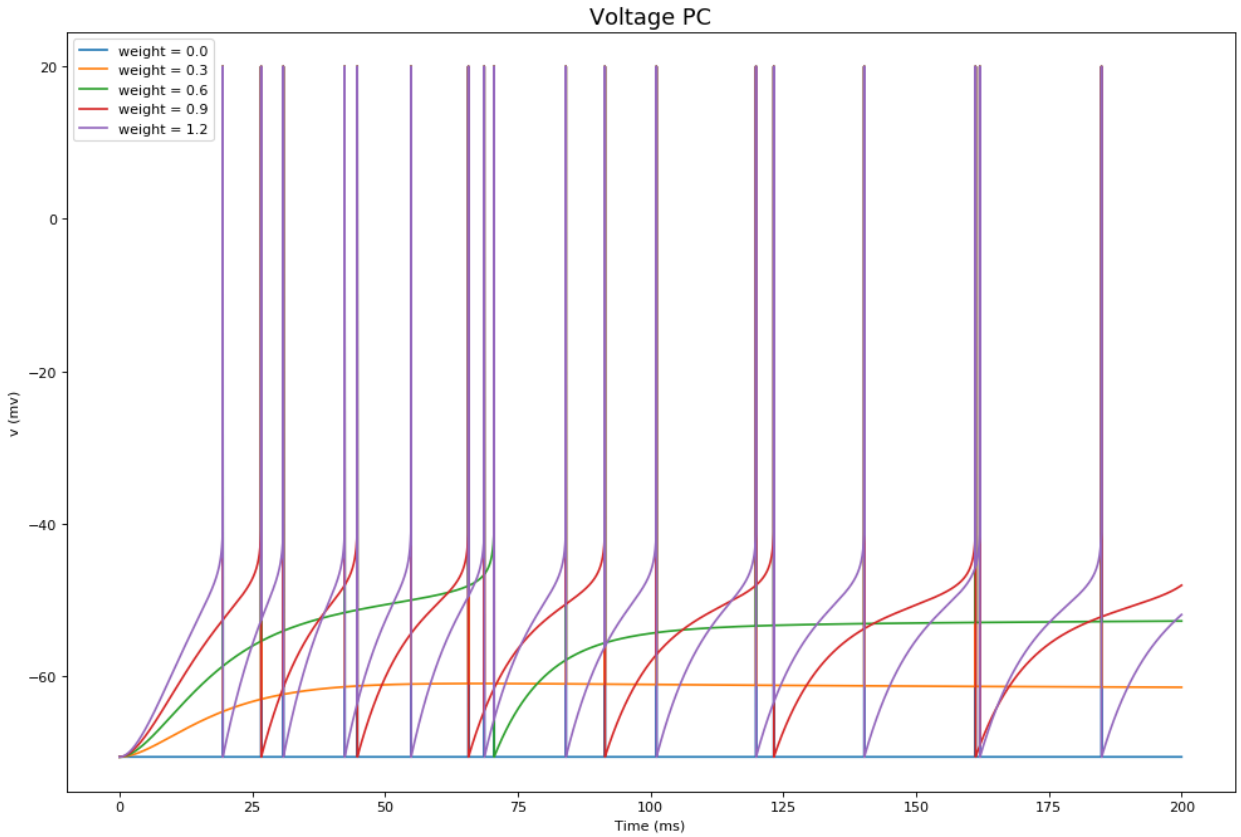


Figure 19: Effect of the synaptic weight on the postsynaptic neuron

It is clear from Fig. 19 that the neuron does not spike if the weight is lower than 0.5 (blue and yellow lines). As the synaptic weight increases, the current entering the neuron is higher and thus, it starts to spike with higher frequency.

7.3 Complementary Definitions for the *Synapses* Function

More can be defined for the *Synapses* function. For instance, equations can be given to describe the way each neuron react following a pre- or post- synaptic spike. For instance, if the presynaptic neuron spikes, the adaptation variable of the AdEx postsynaptic neuron can be updated. This means that following the spike of the presynaptic neuron, the AdEx will take longer to get to the second spike. Analogously defining what happens after a postsynaptic spike can be done using *on_post = '...'*.

```
S = Synapses(Noise, PC, eqs_syn, on_pre = 'w+=b', name = '
    PC_Noise_Synapse',dt=dtt)
```

Listing 21: Pre- or post-synaptic spikes

Articles

Goodman, Dan and Romain Brette (2008). "NEUROINFORMATICS Brian: a simulator for spiking neural networks in Python". In: 2.1.

Hodgkin, A.L. and A.F. Huxley (1952). "A quantitative description of membrane current and its application to conduction and excitation in nerve". In: *The Journal of physiology* 117.

Books

Connors, B. W. and M. J. Gutnick (1990). *Intrinsic firing patterns of diverse cortical neurons*. 13th ed. Trends in Neuroscience.

Dayan, Peter and L.F. Abbott (2005). *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press.

Gerstner, Wulfram et al. (2014). *Neuronal Dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Izhikevich, Eugene M. (2007). *Dynamical systems in neuroscience : the geometry of excitability and bursting*. MIT Press, p. 441. ISBN: 0262090430.

Kandel, E. C., J. H. Schwartz, and T. Jessell (2000). *Principles of neural science*. 4th. Elsevier.

Purves, D. et al. (2008). *Neuroscience*. 4th. Sinauer Associates.

Shepherd, G M. (2004). *The Synaptic Organization of the Brain*. 5th ed. Oxford University Press.

Thompson, R. F. (1993). *The brain*. 2nd. W. H. Freeman and Company.