Introduction to Computational Neuroscience
Practice on Single Neuron Models

Ardi Tampuu, Ilya Kuzovkin, Raul Vicente

October 5, 2016

A request: Please track how long it will take to complete this set of exercises. Add this time to your final report.
NOTE: In this homework you are required to also submit you code.
NOTE: If you are not using Matlab/Octave, you will unfortunately need to rewrite the code in your preferred language. In the beginning of the course I warned this will happen eventually. But consider also doing this session in Matlab, as not much knowledge of the language is required for these tasks - just replace some values here and there.

In this session we will have a brief look on three different computational models of a neuron: McCulloch-Pitts, Intergrate-and-Fire and Hodgkin-Huxley.

Exercise 1: Logic gates (1pt)

On the lecture we have seen how to construct `AND`, `OR` and `NOT` logic gates using the the McCulloch-Pitts model of a neuron. Remember that:
`if sum(w.*x)<0, output is 0`
`if sum(w.*x)>=0, output is 1` (notice that 0 is included)
Your task is to construct more logical operations using MP neurons. Please construct the following two gates:

1. `NAND`
2. `XOR`

HINT 1 For the `XOR` gate you will need more than one neuron.
HINT 2 Same input can go simultaneously to several neurons (and with different weights).

Exercise 2: Integrate and Fire neuron model (2.5pt)

Integrate-and-Fire neuron (IAF neuron) accumulates voltage until it reaches the *threshold*. After that it outputs/emits a spike and resets the voltage back to the reset value. In this exercise we will model the behaviour of IAF neurons and study their properties. We will add features and behaviours to our model step by step.

The file `code/integratefire.m` contains first of all some theoretical explanation of how we are going to simulate a neuron's behaviour (read it!) by numerically solving the differential equation governing integrate-and-fire neuron.
Your task is to complete the series of `TODO`-s in the `integratefire.m`. You need to complete these TODO-s one by one in the given order and report **ALL** figures, answers, interpretations

and conclusions you will make during the work.

You will need to:

1. `TODO 1`: write how to calculate V(2), given V(1), I and C
2. `TODO 2`: write how to calculate V(3), given V(2), I and C
3. `TODO 3`: write how to calculate V(t+1), given V(t), I and C. Plot what happens and describe (describe in comparison with TODO 4)
4. `TODO 4`: replace the 0 with the conditional sentence to check if V(t) is above the threshold level Vth. Plot what happens. Describe, compare.
5. `TODO 5`: replace the 0 with an expression that would add 1 to *number_of_spikes*. Run the for loop again, report the count with default values.
6. `TODO 6`: increase the current to 10.0. Plot the behaviour and report the count.
7. `TODO 7`: Try also other currents, observe the spike count (no need to plot Vm(t) for each current). Plot it in your head and/or on a graph. What is the relationship - exponential? logarithmic? linear? less than linear?
8. `TODO 8`: add the term $noise * sqrt(dt) * randn$ to the V(t+1) equation. This input reflects he fact that neurons always receive random inputs from their neighbours. Observe the noise on Vm plot.
9. `TODO 9`: This `TODO` has its separate part of code below the for loop used so far. Your task is to change the noise level and plot 3 different raster plots (10 trials on each) with different noise levels. Instructions are at the end of the file.

One of the outputs should look something like this

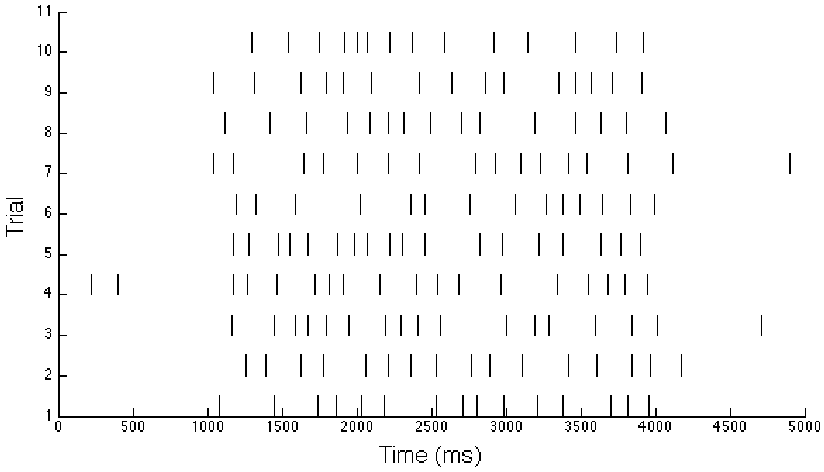

Figure 1: 10 trials of data generated using Integrate-and-Fire neuron model.

Exercise 3: Integrate and fire advanced (1.5pt)

In this exercise we will play with somewhat more realistic versions of integrate-and-fire model.

In first stage we add a leak current to the neuron. This current is caused by the ion pumps and it drives the membrane potential slowly towards its resting state. In the previous exercise there was no such current - when the input current stops, the potential remains fixed at the current value. With leak current added, the Vm now slowly returns to its resting state at -70 mV.
Another difference is that beforehand any current, no matter how small, would eventually drive the neuron to spike. In the current exercise however, weak inputs are balanced out by the leak current.

The code you have to read, understand and modify is in `leakyintegratefire.m`

Your task is to:

- find the lowest $I_{injected}$ (with 0.1 precision) that leads to a spike. Plot the behaviour at this **I** value and at the value 0.1 below (last one with no spike). Describe what happens. Describe also what happens after the stimulus is turned off.
- How does firing rate (spikes per second) differ at I = 5.0, 10.0, 25.0, 50.0 with and without (results in TODO 7 in previous exercise) leak current?

In the second stage we explore the neuron's behaviour if it receives input from incoming (*presynaptic*) spikes rather than constant injected current. Even though one incoming signal is not enough to activate our neuron, *temporal summation* of inputs happening in a short period of time can lead to the voltage reaching the threshold. The code you need to understand and modify is provided in `integratefiresynapses.m`.

For this task you have to remember that if a neuron gets enough excitatory input in a short period of time, it will activate. The same amount of inputs dispersed over a longer period will not lead to firing. Your task is again to modify code near the `TODO` signs and to answer some questions:

1. `TODO 1` The neuron only receives one incoming spike at t=300ms. Plot the membrane potential and the strength of the synaptic current in time. i) Describe how membrane potential and strength of current change over time (peak values, shape). ii) How long does it take for the current to decrease by half (half-life)? How is that related to the "taus" parameter? iii) Modify the strength of the synaptic current up to a point where one presynaptic spike is enough to produce a postsynaptic spike.
2. `TODO 2` Uncomment the line and the neuron will receive a spike every 50ms. Nevertheless it never fires. Why?
3. `TODO 3` Uncomment the line and replace t3 (time of third spike) with a number bigger than 101. What is the lowest value for t3 so that the receiving neuron does not emit a spike?

Exercise 4: Hodgkin-Huxley neuron model (1.5pt)

Hodgkin-Huxley model is considered to be the most important computational neuronal model in the neuroscience today. We have the model already implemented in the file `HHO.m`, study it. Follow the instructions in the `hodgkinhuxley.m` file and report all figures, thoughts, interpretations and conclusions you will have during the work.

The `TODO` marker will indicate places where you have to do something: complete the code, plot and report a figure, give an interpretation, etc.

1. Explain the nature of m, h and n in terms of Na+ and K+ ion channels being open or closed (discussed in lecture).
2. Add the figure at I0=5.0 to the report. Modify I0 and report the lowest I0 that still causes a spike in the beginning of simulation. And the lowest I0 that causes continuous spiking till the end of simulation. (Precision of your answer should be at least 0.1)
3. Count the number of spikes at different I0 values (code provided). **Note that** this series of simulations can take up to 5-10 min, do not panic if it takes time. Describe what you see. Why this *input-output curve* it more biologically realistic than the input-output relation of IAF? How is the non-linearity of the relationship related to modelling the ion channels and not simply resetting the membrane potential instantaneously as soon as threshold is reached?

Exercise 5*: Bonus (0.5pt)
I take the freedom to award bonus points for the appearance of the report. Some reports are really badly formatted and hard to read. So I want to reward people who make their reports more readable and my life more pleasant.

Please submit a `pdf` report with answers to the questions and comments about your solutions. Include figures, explanations. **Include the completed code files and zip the files together.** The code files are not the place to answer questions, answers must also be contained in the pdf. Upload it on the practice session page on the course website.