Computational Neurodynamics

Exercise Sheet 2 (Unassessed)
Neural populations and Braitenberg vehicles

Programming questions

Question 1.

The Hodgkin-Huxley model we implemented in Exercise 1 was very biologically realistic, but also computationally demanding. Let us begin our journey towards large simulations of complex neural networks by implementing a few simpler neuron models.

a) Write a Python script that simulates the activity of a single Izhikevich neuron receiving a constant input current $I = 10$. Use the parameters for an excitatory (regular spiking) neuron from the lecture notes, a step size of $\delta t = 0.1$, and initial conditions $v = -65$, $u = -1$.

b) Using the lecture notes, adjust the parameters for the Izhikevich neuron so that they emulate an inhibitory neuron.

c) You will have noticed that each of these types of neurons produces very different spiking behaviour even when subject to the same, constant input current. Several important properties of the neuron's spiking response can be captured in the so-called *frequency-current* (or F-I) curve, that represents the output firing rate of a neuron (in the Y axis) for different values of its input current (in the X axis). For each of the parameter settings simulated in parts *a* and *b*, plot their F-I curves by running multiple simulations with different values of the input current (e.g. between 0 and 20) and taking the temporal average of the number of spikes in a long time interval (e.g. around 1s).

Question 2.

Now you have implemented and experimented with single Izhikevich neurons, let us add another degree of complexity by simulating populations of neurons.

a) Implement a function (or class) that simulates N Izhikevich neurons at the same time using the Euler method. This could be achieved, for example, by storing the values of the Izhikevich parameters in arrays (e.g. one array of size N for the a parameter, and so on), and then using the Euler method with Python's vectorised operations. You can feed a constant current, e.g. $I = 5$.

b) Let's add some variation in the parameters. Set up the population to contain either N excitatory neurons or N inhibitory neurons. For each neuron, draw a random number r at uniform in the interval [0, 1], and set the parameters as recommended by Izhikevich's original 2003 paper:

	Excitatory	Inhibitory
a	0.02	0.02 + 0.08 *r*
b	0.2	0.25 – 0.05 *r*
c	-65 + 15 r^2	-65
d	8 – 6 r^2	2

Simulate an excitatory and inhibitory population under constant current, and make a raster plot reproducing (qualitatively) those in the lecture notes.

c) Implement connections between the neurons. Each connection has two properties: a weight *W* and a delay *D*, determined by two N-by-N matrices respectively. When neuron *i* fires, neuron *j* receives an instantaneous pulse of current with value *W(i,j)* happening *D(i,j)* timesteps later. Reasonable values are, for example, *D(i,j)* = 5 and $W(i,j) = 50/\sqrt{K}$, where *K* is the number of incoming synapses of the post-synaptic neuron.

Make sure to test your connection delays properly. You can do this e.g. by initialising a network with two neurons and a very strong unidirectional connection, feeding external current only to neuron 1, and checking that the spike in neuron 2 occurs when expected from the connection delay.

Question 3.

Time to put some behaviour into the mix. Let's implement our very own Braitenberg vehicle. To do this, you can use three files with auxiliary code:

i) The file `environment.py` contains auxiliary code to simulate the environment a Braitenberg vehicle lives in. Specifically, it exposes functions to initialise object positions, obtain the sensor readings for a vehicle in a certain position, and update the vehicle given certain wheel speeds.

ii) The file `animate_vehicle.py` contains code to integrate the environment and the controller and animate the results. You should be able to run this file as-is to test your solution.

iii) The file `controller.py` contains the signature of two methods that you will have to implement.

Your tasks are the following:

a) Using the Izhikevich population simulator you wrote in Question 2, implement a neural controller for a Braitenberg vehicle and reproduce the behaviour shown in the lecture notes. Here are a few tips:

- All the neurons should be excitatory. 4 neurons per population is a good size to start.
- A conduction delay of 5 timesteps between the sensory and motor populations is recommended. However, if you didn't get to implement the delays in Question 2c, you can still attempt this question without them.

- For the output of the motor populations, you should take the mean firing rate – i.e. the average number of spikes per neuron per second.
- Note that there are two timesteps involved in this simulation. One is the timestep of the Euler integration (recommended 0.1ms) and the other is the timestep of the robot's movement (recommended 100ms).

b) Alter the network inside the robot so that it avoids objects rather than approaching them.

c) (for enthusiasts only) In which cases does the vehicle fail most often? Think about the cases where there is one object at each side of the robot. Using the ideas in Topic 6 (Competition) modify the neural net inside the vehicle to deal with these cases more effectively.

Written questions

Question 4.

a) Write down all the equations that comprise the leaky integrate-and-fire (LIF) neuron model. Describe the role of each equation.

b) A LIF neuron with parameters $\tau = 5\text{ms}$, $R = 1$, $\alpha = 0$, $\theta = -50\text{mV}$, and $v_r = -65\text{mV}$ is driven with a constant dendritic current of $I = 20$ from an initial voltage $v(0) = -65\text{mV}$. In the limit of a long time window, what is its average firing rate (in spikes per second)?

Question 5.

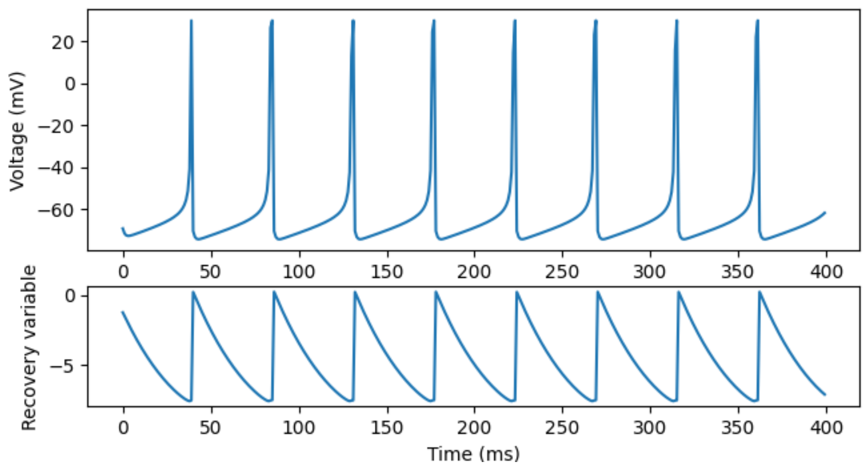
a) The Izhikevich model of spiking neurons has a voltage variable v and a recovery variable u, governed by two coupled differential equations:

$$\frac{dv}{dt} = 0.04\,v^2 + 5v + 140 - u + I$$
$$\frac{du}{dt} = a(bv - u)$$

The model has one more component that determines the temporal evolution of *v* and *u*. What is it? And why is it necessary?

b) The figure below shows a plot of the membrane potential *v* and recovery variable *u* for an Izhikevich neuron with parameters $a = 0.02$, $b = 0.2$, $c = -65$, and $d = 8$, subject to a constant dendritic current of $I = 10$.

What would be the effect of increasing *a*? Give your answer in words and sketch plots for *v* and *u*. Your answer should be qualitatively correct, but does not need to present exact values.



Question 6.

a) Describe the role of the parameter d in the Izhikevich neuron model. What is the effect of increasing the value of d?

b) We simulate an Izhikevich neuron with $a = d = 0$ and $I = 0$, and we initialise the simulation with an initial value $u(0) = 0$. What neuron model does this correspond to? What behaviours could the neuron exhibit depending on the initial voltage value $v(0)$? Give specific numerical values for any quantities of interest.