Program Presentation

I implemented a version of the neural network described in the Brunel paper from 2000. We made the following assumptions:

- There are only 2 types of neurons: inhibitory and excitatory and the excitatory/inhibitory postsynaptic potentials have the same value for all the neurons respectively (J_E and J_I)
- Each neuron receives C_E exitatory connections and C_I inhibitory connections from neurons chosen randomly in the network
- A neuron can receive two or more connections for a single neuron
- It is not possible for a neuron to be connected to itself
- The activity of other areas in the brain (background noise) is simulated by an exitatory poisson distribution, sent randomly to all the neurons in the network

For the implementation I chose to attribut an index (int) to each neuron. I made an array<deque<int>> to store the indexes of all the neurons that receive an input from the neuron at a particular line.

The network also has an array<Neuron*> which stores pointers pointing to all the neurons making up the network.

The buffer is an array of size D (=delay) + 1. When the buffer is filled with an excitatory postsynaptic potential, the value is put at the position D in the buffer. Then, it automatically moves forward one case per timestep. That way we make sure that the delay is respected and the need to use the current time and the modulo operator is eliminated.

To obtain the different states (A, B, C and D) described in the paper, it is necessary to change to constants eta and g to the appropriate values in the constants.hpp file.

State	**Brunel simulation**	**This implementation (average)**
B (eta = 4, g = 6)	60.7 [Hz]	62.2 [Hz]
C (eta = 2, g = 5)	37.7 [Hz]	33.5 [Hz]
D (eta = 1, g = 5)	5.5 [Hz]	7.08 [Hz]

The values we obtained differ a little, which is normal since there are a lot of random elements in the network and therefore it is not possible to obtain an exact value. Also, the states described are for a range of eta and g values.

Using googletest, I also tested the different functions of a single neuron and of a whole network in the neuron_unittest.cpp file. In order for the test to work, it was necessary to create a bool poisson_ as a class attribute of the class Neuron, this deactivates the background nois for this particular neuron. Also, I would like to mention that the function “fill_buffer” should be private but was put in public because it is used for testing.

On Github, all the final cpp and hpp files are in the src folder, the other files are older versions with all the commits history.

Compilation of the program

To compile the program you have to go to https://github.com/jeromel05/cppcourse-brunel and dowload the zip archive. Unzip it, create an empty build folder inside. Go to https://github.com/google/googletest and dowload the zip archive. Inside the googletest-master folder you will find a folder named googletest. Rename it “gtest” and put it in the src folder. You also need to install cmake.

In the terminal, from the <cppcourse-brunel-master/build> directory, type the commands “cmake ../src” and then “make”. After that you can run the program by typing “./projet_neuron”.

To change the runtime of the program, you need to change the argument of the update function called in the main.cpp file.

For testing type: “./neuron_unittest”