CS 443: Brain Inspired Computing

Assignment # 1 - Simulating Neuron Models

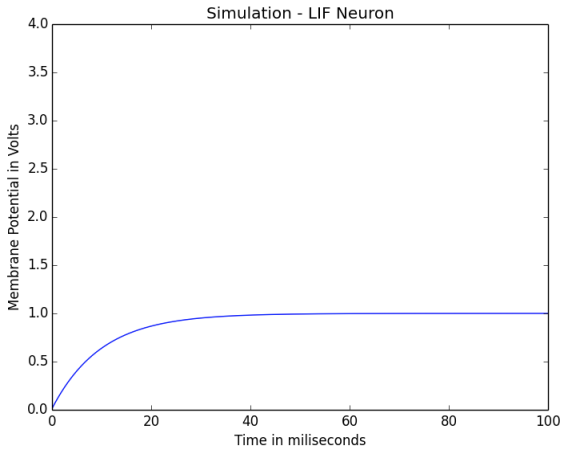
Shashank Sharma
NetID: ss2795

Questions:

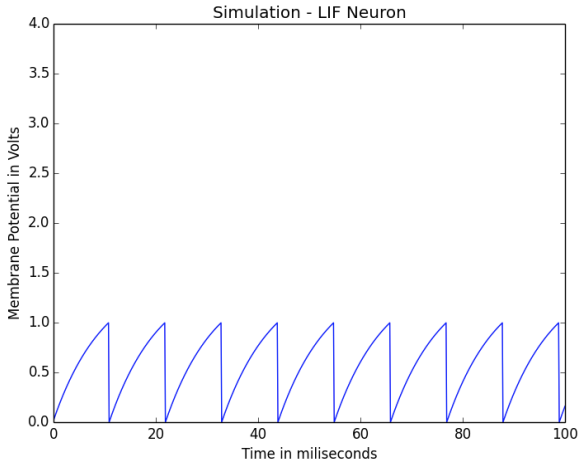
1. **If an IF/LIF neuron is fed a constant low input:**
    a. If an integrate and fire neuron is constantly fed a very low input, it is expected that the neuron will eventually fire, albeit it might take some time, as the input is extremely low. This is because the integrate and fire neuron has no mechanism to take into account how long ago a given input occurred, and hence, given a constant input, the total membrane potential will eventually reach a given threshold, and eventually fire. After firing, the membrane potential would then reset, and the process would start again. Since the potential is linearly dependant on the input, and the input is constant, the neuron would continue to fire at a constant rate.
    b. If a leaky integrate and fire neuron is fed a constant low input, we would would not expect it to fire, as the LIF neuron model takes into account how long ago inputs occurred, and therefore if the constant leak is higher than constant input, the membrane potential would remain at 0 after each step, never reaching the threshold, and therefore, never firing.
2. **If an IF/LIF neuron is fed a constant high input:**
    a. If an IF neuron is constantly fed a constant high input, we would expect it to fire, as before, but at a higher rate, as the firing rate of an IF neuron is linearly related to the input current, and therefore, if the current increases, the firing rate also increases.
    b. If a LIF neuron is constantly fed a constant high input, we would expect it to actually fire, as the input is more than likely higher than the leak rate, and therefore, will continue to build up the membrane potential until the neuron actually fires. The neuron would then reset, and then the process would start, causing the LIF neuron to fire at a constant rate, since once again, the input current and firing rate are linearly dependant.
3. **The limitations of an LIF neurons are:**
    a. Firing rate is linearly dependant on input current, and does not take into account the refractory period that occurs in real neurons after they are fired, and therefore, neurons continue to spike at a constant firing rate, in addition to not being able to model adaptation
    b. After firing, the neuron is reset completely to an initial state, which ensures that previous spikes are completely forgotten, which, again, is not representative of how neurons actually work, as there is no form of adaption, or no way to have a post-synaptic potential lower than the resting potential.
    c. Neurons generally receive inputs from various different neurons, not a single current input, which cannot be modeled using the basic LIF neuron formula
    d. It is an extremely simple model, which helps computationally, but does not help create an accurate model of how neurons work. There are other models that require just a little more computational power, but are significantly better.

Programming:

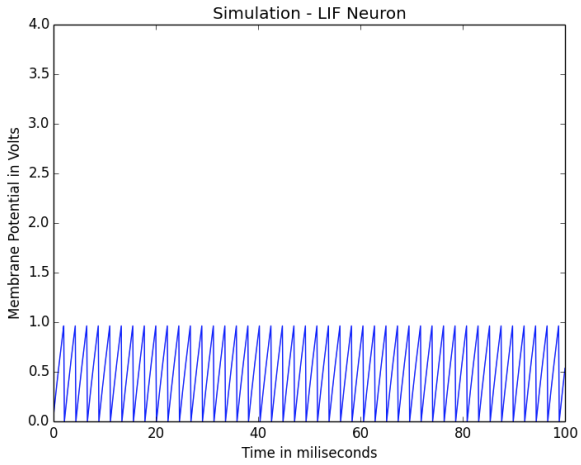
1. **LIF Neuron Simulation (Code Zipped with project):**
   a. Constant input of 1 V (Input too low to fire, since "leak" is higher than potential gain)



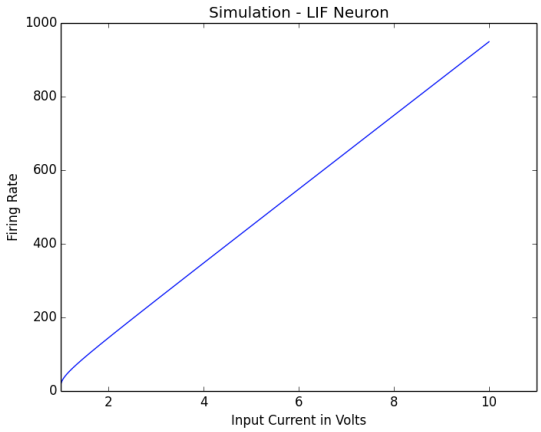
   b. Constant input of 1.5 V (Input sufficiently high to fire, fires at constant rate)



   c. Constant input of 5V (Input extremely high, fires at high, constant rate)



2. **Firing Rate of LIF Neuron based on various input currents:**



3.

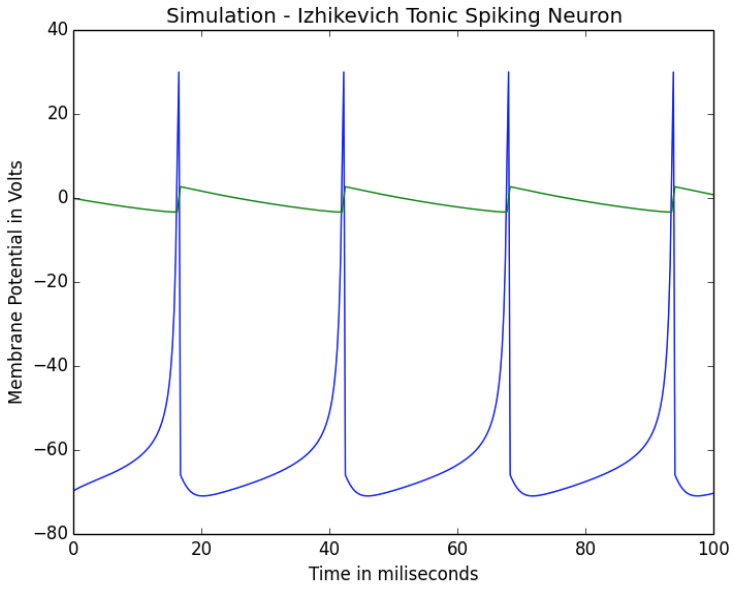
a. As you increase the input current in volts, the firing rate increases linearly. This is because the LIF neuron model bases its membrane potential directly on input voltage. Since change in membrane potential happens linearly based on input current, and the threshold remains constant for any constant input current, the firing rate for any given input current is constant, and grows linearly. The graph was created using the derivation of the firing rate formula from the initial LIF Model formula, giving us:

b. $T = \tau m^*(\ln(RI/RI-Vth))$

c. Firing rate = 1/T

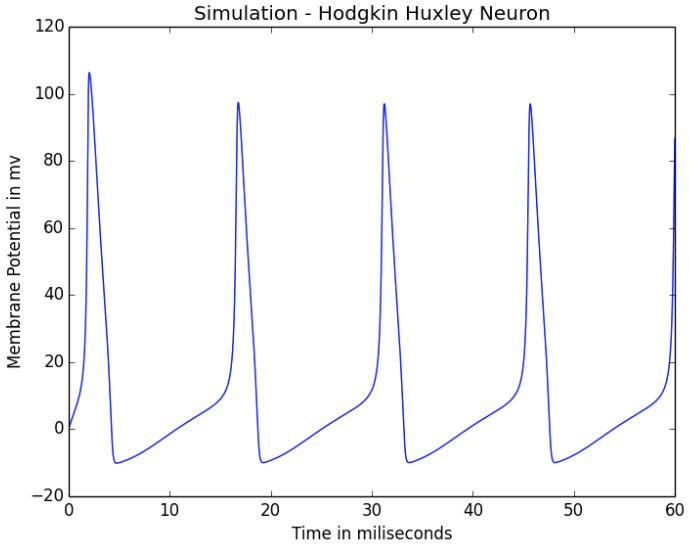
4. **Simulating Izhikevich Neurons**

a. Simulating a Izhikevich Tonic Spiking Neuron



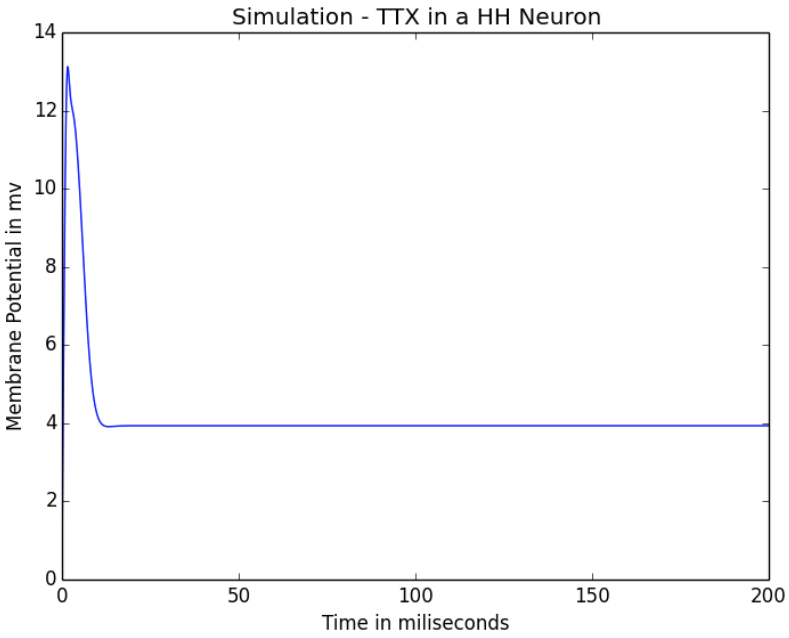
5. **Simulating a Hodgkin-Huxley Neuron:**

a. Used http://neurdon.wpengine.com/2011/01/26/neural-modeling-with-python-part-2/ as a source to get various channel functions for the simulations. Using these channels, and the given formula in the HH paper, you get this simulation:



6. TTX Simulation:

a. For both the TTX and Pronase simulations, I modified the previous HH simulation



7. Pronase Simulation