This is completely my own work.

Collaboration:

* Received clarification on some aspects of the questions from Jung Uk Kang

Aneesh Sachdeva

441472

L41 BME 5657

HW 1

Problem 1 – HH Model

The purpose of this experiment is to observe and analyze the dynamics of the Hodgkin-Huxley variables following stimulus. When adequate stimulus is applied, and action potential (or EPSP, excitatory post synaptic potential) is created. If the model is built correctly, the simulated membrane voltage, ion conductance, and ion currents will follow the same physiology observed in a real action potential.

The model follows the differential equations of the general Hodgkin-Huxley model (Figure 1). The equations are programmatically applied via Euler integration in python. dV/dt is obtained by rearranging the overarching differential equation in Figure 2 to be an expression of dV/dt. This results in dV/dt = (I\_stimulus – I\_ions)/membrane capacitance.

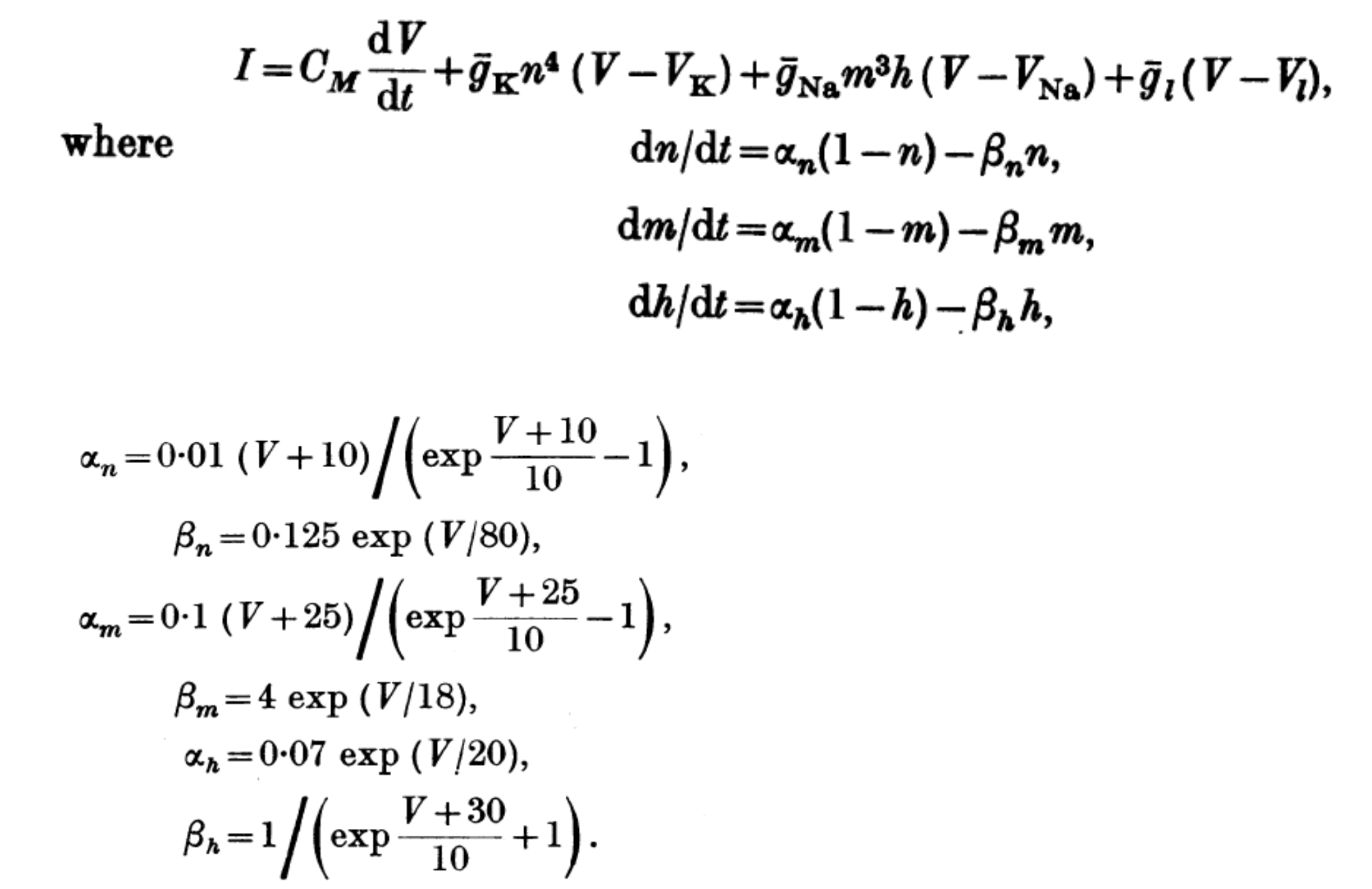


Figure 1: Hodgkin-Huxley differential equations. These are programmatically applied using Euler integration.

(continued below)

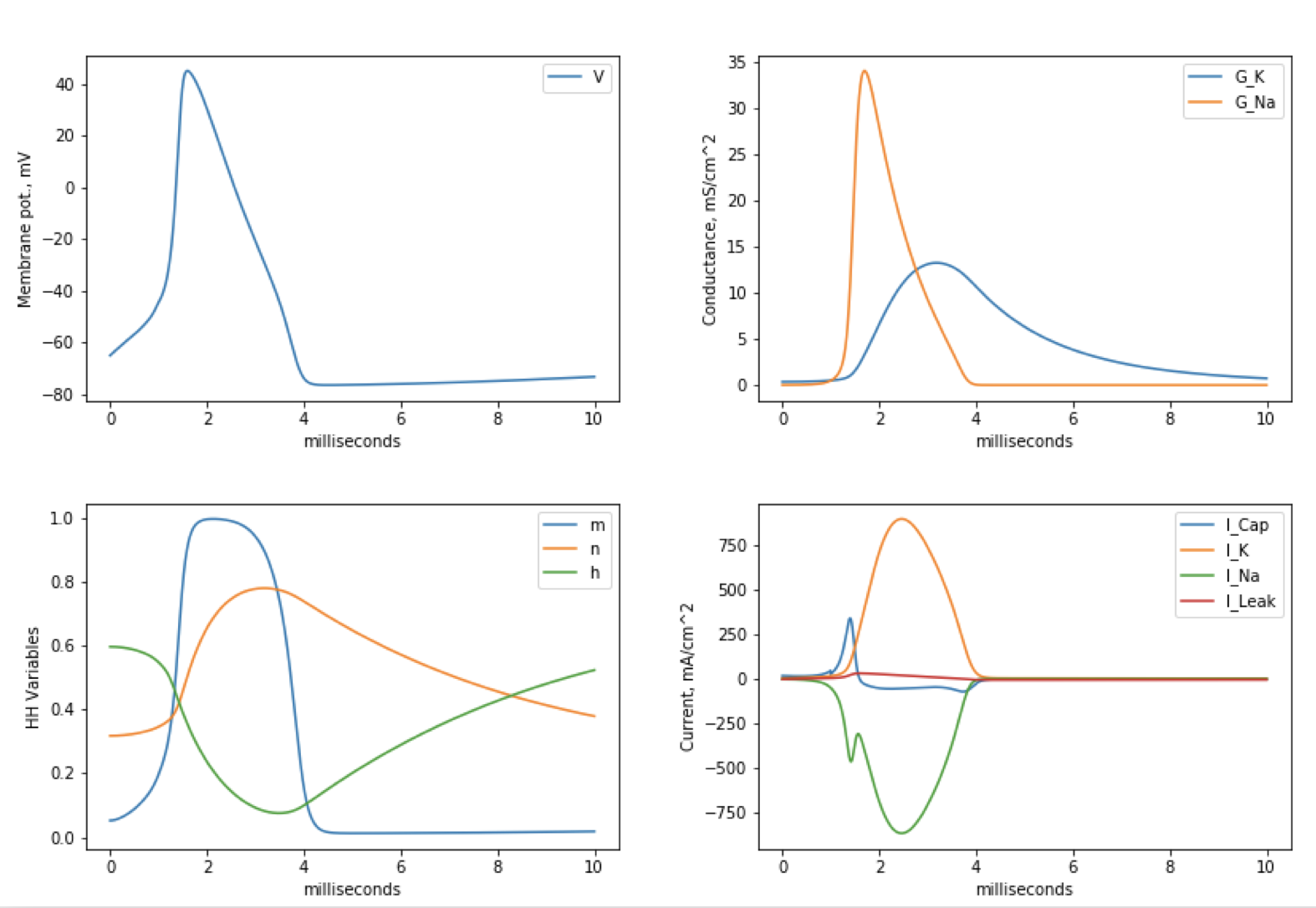


Figure 2: Dynamics of Hodgkin-Huxley variables during an action potential (AP). (Top Left) Membrane potential of neuron following 1 second of 20nA stimulus. (Top Right) Conductance of sodium and potassium during AP. (Bottom Left) Activation variables during AP. (Bottom Right) Capacitive, potassium, sodium, and leak currents during AP.

The dynamics of the Hodgkin-Huxley variables after a 1 second stimulus of 20nA are depicted in Figure 2. As expected, we see the membrane potential exponentially increase towards the sodium potential (+55mV) after passing a threshold around -50mV. This is affirmed by the sodium conductance spiking prior to potassium conductance. The membrane potential then decreases towards the potassium potential (-77mV) as sodium conductance passes its peak and potassium conductance reaches its peak. During the course of the AP we observe the capacitive and sodium current going of the cell, and sodium current going into the cell. This is confirmed by the opposite signs on the sodium and potassium currents. Leak current stays relatively flat as expected.

The results agree with our understanding of the physiology of a neuron’s action potential. Sodium channels operate faster than potassium, and so we see sodium conductance spiking before the slower potassium channels open up. Sodium first enters the cell, causing the increase in membrane potential, and then the opening of the potassium channels and inactivation of the sodium channels results in an out flux of potassium ions that drives down membrane potential. These gating and activation mechanisms are explained by the HH variables, *m, n* and *h* (figures 1 and 2). Variable *n* gates potassium conductance, which peaks shortly after the peak of *m*, which gates sodium conductance. Variable *h* assists with the transition between sodium channel inactivation and potassium channel activation. We observe the membrane potential correctly hyperpolarizing below resting potential (-65mv). Leak potential remains relatively flat throughout the AP, which is physiologically correct as potassium minutely leaks out of the cell to help maintain the electrochemical gradient needed for action potentials to occur.

Now the threshold and rebound behavior of action potentials under different conditions of stimuli is evaluated.

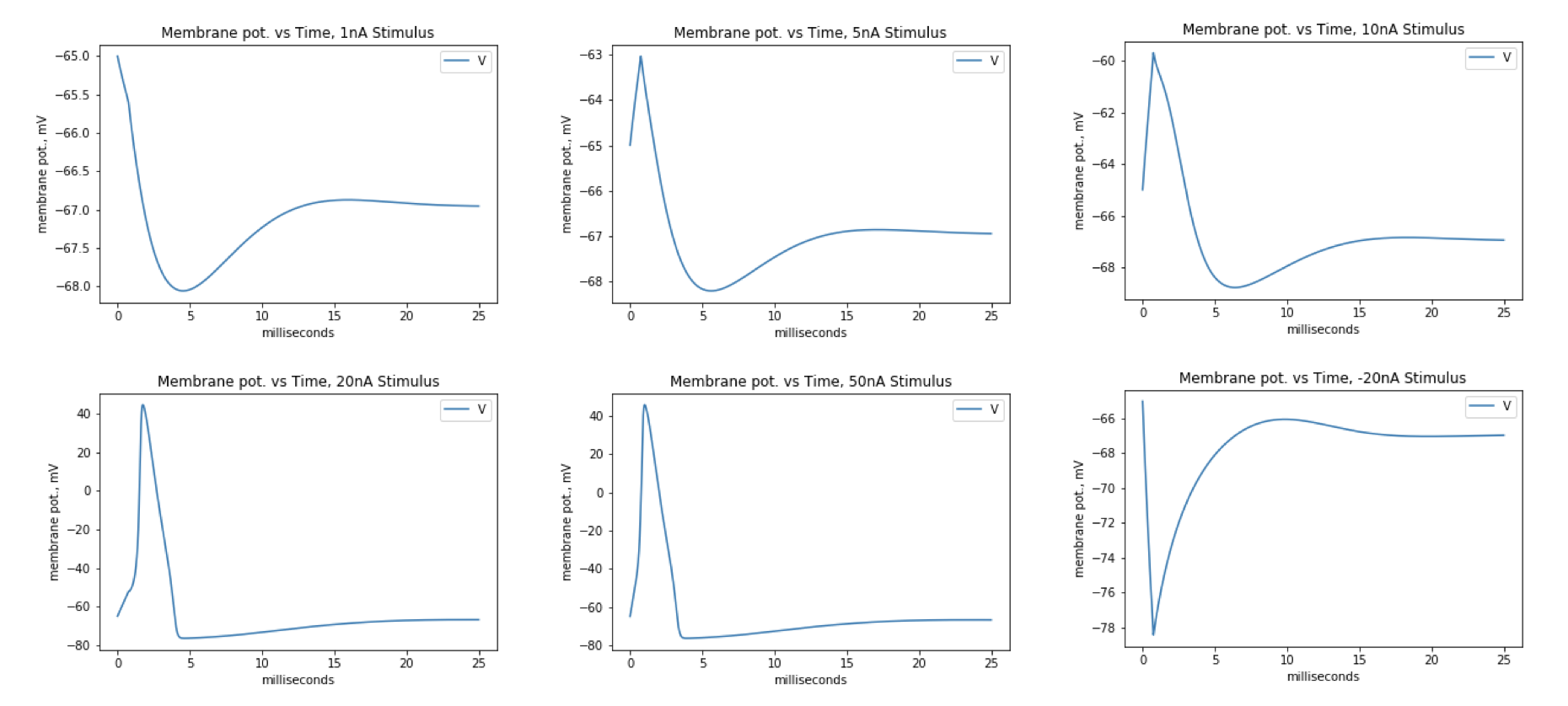


Figure : Response of a neuron to various levels of stimulation as measured by the chance in its membrane potential over time. All stimuli were applied for 750ms. Action potentials are observed to occur only after adequate stimulus is applied, and their peak potential is not dependent on the stimulus applied.

The purpose of this experiment is to evaluate the threshold and rebound behavior of action potentials. Action potentials will not occur unless the stimulus is adequate enough to elevate membrane potential above threshold.

Using the general Hodgkin-Huxley model (Figure 1) varying stimuli is applied to the simulated neuron. The only independent variable is strength of the current applied. All current is applied consistently over a 750ms duration, and the resulting membrane potential is measured over 25 seconds.

At low levels of stimulus current, the neuron may exhibit minor perturbations in membrane potential (similar to MEPs, miniature end-plate potentials) but never achieves a full action potential (Figure 3). Action potentials are observed with stimulus current >~20nA, as there is enough current to bring the membrane potential to threshold, after which the membrane potential accelerates towards the sodium potential (+55mV).

As long as stimulus is adequate to bring membrane potential to threshold, the magnitude of the stimulus is observed to have no effect on the peak or duration of the action potential (Figure 3). This agrees with the principle of threshold behavior. No matter the positive current applied, the membrane potential always slightly hyperpolarizes below resting potential (-65mV). When negative current is applied, the membrane potential spikes towards the potassium potential (-77mV), as if it was a mirrored reflection of a normal action potential across the x-axis. After negative stimulus, the membrane potential even slightly depolarizes above resting potential before settling to dynamic equilibrium.

From these results, it’s clear that action potentials can only occur when stimulus is adequate to bring membrane potential to threshold, but beyond that stronger stimulus has no additional effect. This is because the gating of ion conductivity allows follow the same dynamic behavior after the threshold is reached.

Problem 2 – IAF Neuron

The purpose of this experiment is to study the behavior of a simple integrate and fire neuron (IAF) under various frequencies of stimuli. After frequency of stimulation is increased past a certain level, the IAF neuron will stop firing action potentials.

A simple stepwise function (Figure 4) containing a differential equation is used to construct the IAF in python. The differential equation (used before the neuron hits threshold) is implemented via Euler integration. dV/dt is expressed by rearranging the overarching equation into dV/dt = (I(t) – V(t)/R) / C. A time step of 1ms is used for Euler integration, and membrane threshold is set to 5mV.

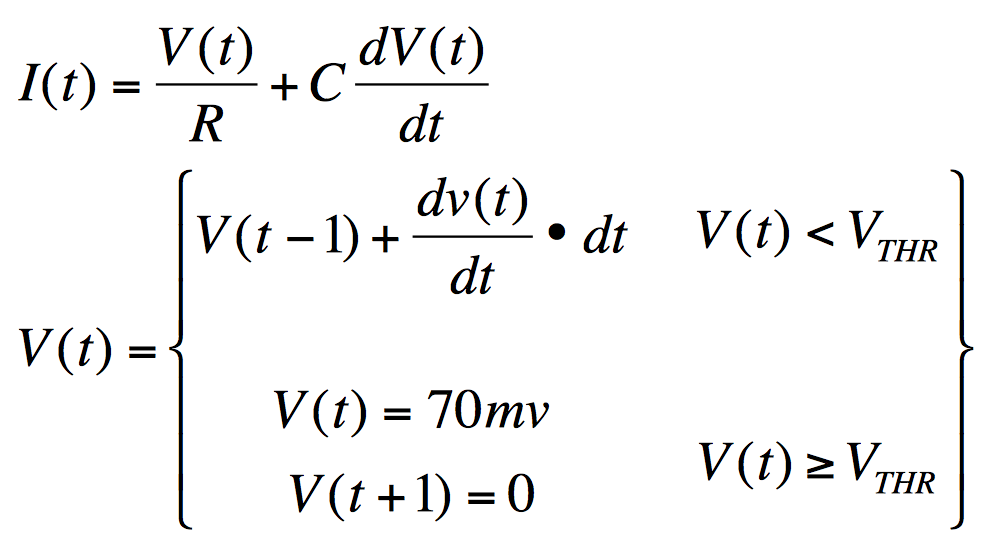


Figure 4: The stepwise function used to construct a simple IAF neuron model. When membrane potential reaches threshold, an action potential occurs instantaneously, and is returned to 0mV in the following time step. Model is implemented via python and Euler integration.

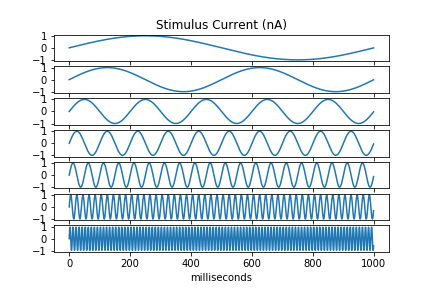


Figure 5: The various levels of frequency used to stimulate (current) the IAF neuron. From top to bottom, 1.0Hz, 2.0Hz, 5.0Hz, 10Hz, 20Hz, 50Hz, 100Hz. All stimuli are applied for 1 second.

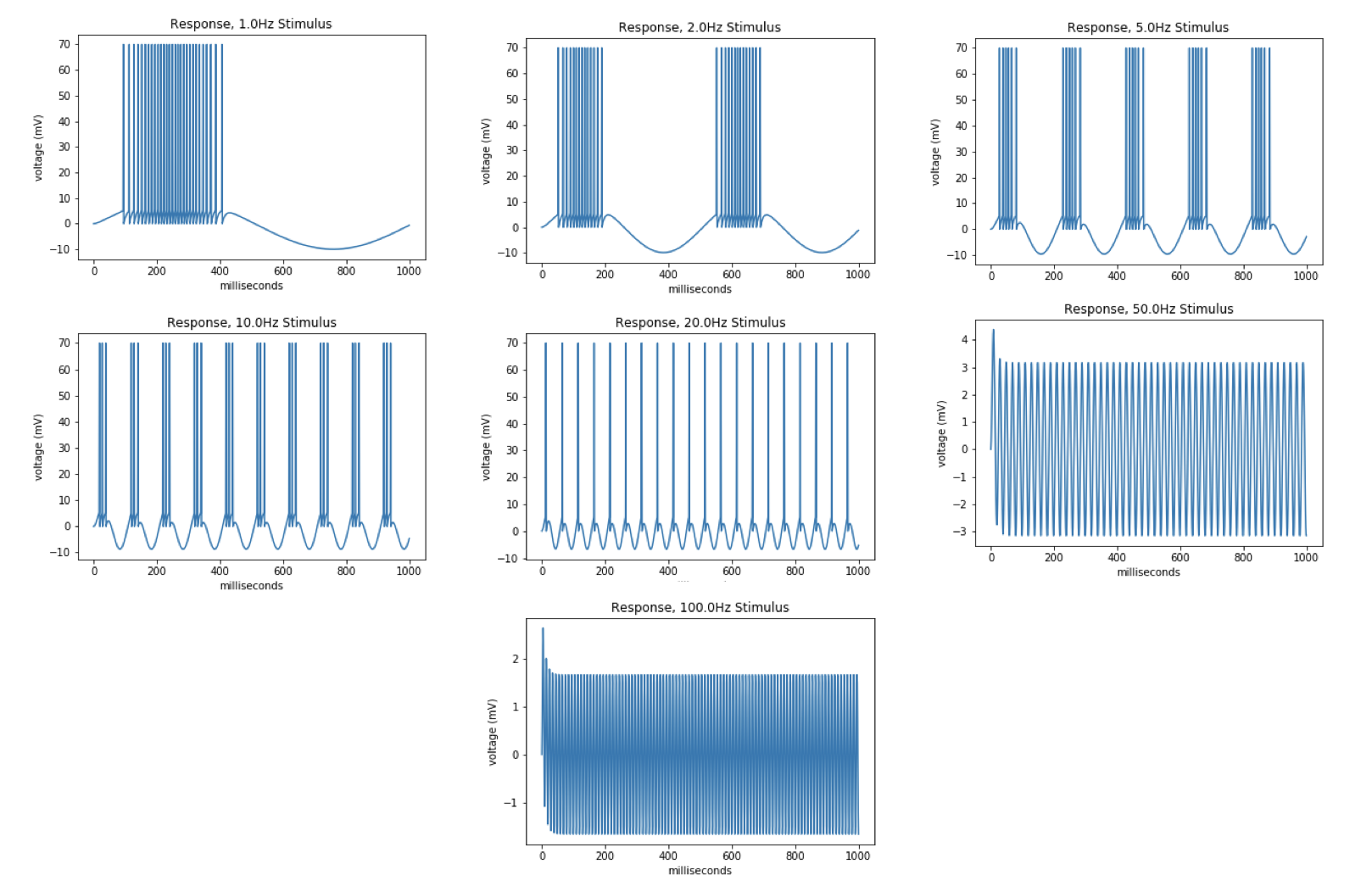


Figure 6: Response of IAF neuron membrane potential to various frequencies of stimuli. Action potentials stop occurring during 50Hz and 100Hz stimuli. All stimuli are applied for 1 second.

After applying sinusoidal stimuli of increasing frequency (Figure 5) to the IAF neuron, action potentials are observed to stop occurring at high enough frequencies (Figure 6). This behavior is due to the rapidly changing and high magnitudes of dV/dt that is a result of high frequency stimulation. Since the amplitude of the stimulus alternates rapidly between 1.0 and -1.0 for high frequencies, dV/dt correspondingly ends up alternating signs rapidly, resulting in membrane potential never reaching threshold. If membrane potential never reaches threshold then an action potential will never occur. Although an action potential doesn’t occur, the membrane potential still oscillates rapidly, in-line with the rapidly changing nature of dV/dt. At low frequencies of stimuli, however, dV/dt is calm enough for membrane potential to reach threshold, and for corresponding action potentials to occur. At these frequencies, all action potentials are observed to occur during the positive amplitudes of stimuli. This is a result of a positive dV/dt which originates from positive stimulus current (Figure 6).

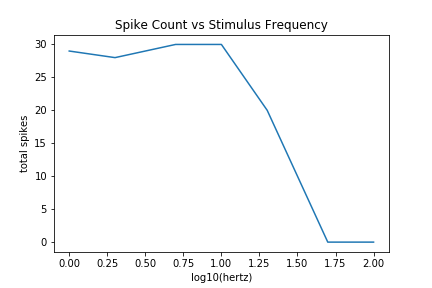


Figure 7: The relationship between stimulus frequency and total number of spikes (action potentials) over a second of stimulus.

This experiment demonstrates the relation between stimuli frequency and amplitude, and corresponding action potential frequency (or lack-of) (Figure 7). Although action potentials fail to occur when the stimulus rapidly alternates between positive and negative amplitude, when only positive stimuli is applied action potentials can occur in rapid succession. This understanding can help us extrapolate how connected neurons can inhibit and excite each other.

Problem 3 – Linear Model

The purpose of this experiment is to construct a Linear model and test its ability to predict spike trains given stimuli. After training on four trials of stimuli and response, the resulting linear filter should be able to reasonably predict the spike train of a fifth trial. This experiment is a simplified recreation of *Geffen, MN, Broome, BM, Laurent, G and Meister, M. (2009). “Neural encoding of rapidly fluctuating odors.” Neuron 61(4): 570-586*

The linear filters used to create the predicted and actual spike trains are trained via least squares regression by minimizing (Sk – R)2 where S is the stimulus matrix, k is the linear filter, and R is the discretized response matrix. Given the minimal size of the dataset, we can cleanly use the closed form linear regression solution in Figure 7 to find the filter that minimizes the residuals. The response matrix is created by discretizing the recorded spikes into 100ms bins, where each bin has a value 1 if there is a spike in that time range and 0 if not. The stimulus matrix is created by first using the same binning procedure, then moving a 2s rolling window across bins and saving each window as its own row in the stimulus matrix. In summary, the filter predicts the intensity of the response at the end of 2 seconds of stimulus.

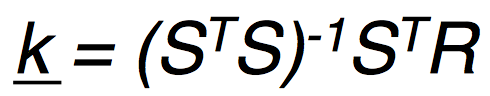


Figure 8: The closed form solution of finding the linear filter, k, which minimizes (Sk - R)^2, where S is the stimulus matrix, k is the linear filter, and R is the response matrix.

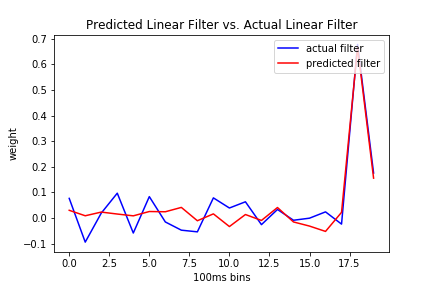


Figure 9: Each filter is created from minimizing (Sk - R)^2, where S is stimulus, k is the linear filter, and R is the discretized response matrix. The predicted filter is trained on an average of the response matrices from trials 1-4. The actual filter is trained on the response matrix from trial 5.

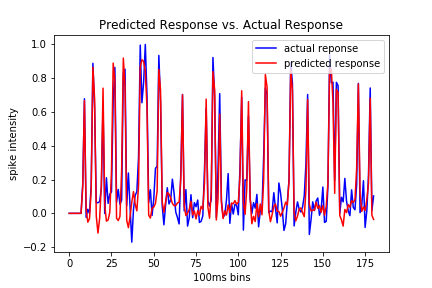


Figure 10: Comparison between predicted spike train and actual spike train from trial 5. Each response is created by multiplying a linear filter and stimulus matrix. The linear filter for the predicted response is created using trials 1-4.

The predicted filter and actual filter for trial 5 (Figure 9) are similar in shape, with the predicted filter having smoother weighting during 0-1.7s. Given that the predicted filter is trained from an *average* of the responses from trials 1-4, it makes sense that the filter will be less sensitive to perceived noise in the response, as non-systematic noise usually smooths after being average. The predicted and actual spike train responses (Figure 10) are similar, however the predicted response often falls short on the magnitude of peak amplitudes compared to the actual response. This result agrees with the similarity of the filters and the relatively low and consistent (smoother) weighting in the predicted filter; with smaller magnitude of weights there should be smaller response intensities produced. The shape of the filter and the accuracy of the predictions also agree with the notes from John Pillow on slide 41 of lecture 4.

This model can be improved by adding non-linearity (i.e. ReLU or sigmoidal) to amplify the intensities of the spikes, however with respect to the simplicity of the linear filter, the accuracy we obtained is reasonable. The success is due in large to the design of the stimulus and response matrix; under the assumption that stimulus and response are i. i. d., there should be some distribution of p(response | stimulus lag). A disadvantage to the simple linear model however is that it equally weights old stimulus (in the 2 second window) and recent stimulus, relative to the response. This assumption is most likely not accurate, as I would expect recent stimulus to have more weighting on the response. To test this, an exponentially weighted (by stimulus timing) linear regression model should be explored. Other areas for exploration include experimenting with bin size, stimulus lag size and the number of neurons used.

Sources referenced:

<https://stackoverflow.com/questions/6811183/rolling-window-for-1d-arrays-in-numpy>

* Referenced for assistance with creating the rolling window for stimulus matrix creation.