Bioelectric Homework #4

Matlab Program:
See the appendix for matlab code.

Bi-directional Action Potential:

Minimum Cathodic Currents to stimulate Bi-Directional AP
The model was simulated using an extracellular monopole source with a 0.8ms anodic square wave pulse followed immediately by a 0.2ms cathodic pulse. In each simulation, the amplitude of the anodic current was set to $\frac{1}{4}$ the amplitude of the cathodic current. The simulation was run for the following perpendicular distances (z) between stimulus electrode and axon: 1mm, 2mm, 4mm and 8mm. For each distance the minimum amplitude of cathodic stimulation required to elicit a bi-directional AP was obtained trial and error using a recursive approach that halved the problem size each recursion until a threshold was converged upon with fidelity of a single single μ-amp. The following values were obtained.

Distance z:	Threshold Cathodic Current Amplitude
1mm	145 μ-amps
2mm	501 μ-amps
4mm	2114 μ-amps
8mm	11243 μ-amps

These results were as expected. They demonstrate how sensitive the axon is to closer stimulation electrodes. Careful placement of stimulus electrodes could provide selective stimulation by electrode location alone.

Surface Plot for z=1mm:

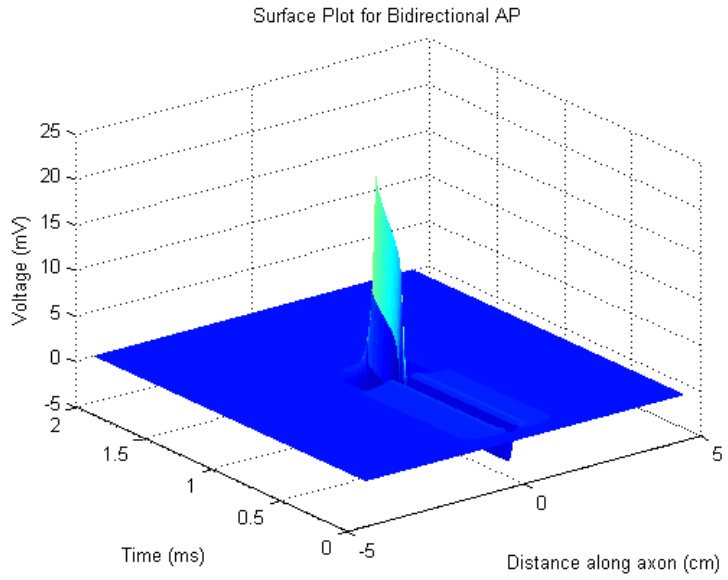

Figure 1: Bidirectional Action Potential for electrode with stimulus amplitude just below threshold 1mm from axon.

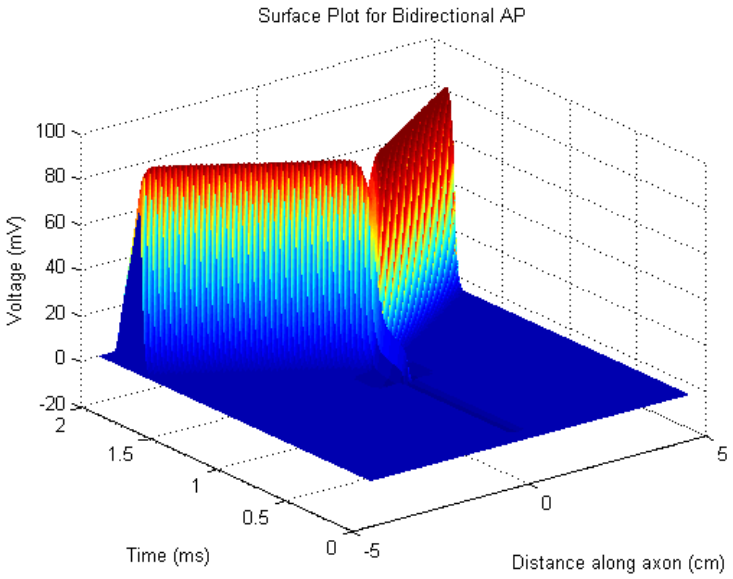

Figure 2: Bidirectional Action Potential for electrode with stimulus amplitude just above threshold 1mm from axon.

Bi-Directional AP Movie:
An MPEG-4 video of the bi-directional AP was uploaded to BlackBoard.

Voltage and Current Snapshots:
Defining the instant when the action potentials have "propogated" half-way down their respective sides of the axon can be interpreted in two ways. The first is the instant the middle point of each half of the axon sees a voltage above 0. And the other is when the peak of the action potential reaches that point on the axon. Provided below are 2 sets of 3 images. The upper 3 images correspond to the membrane voltage, membrane current and longitudinal axonal current for the first definition of the instant from above. The lower 3 images represent the same graphs for the instant when the peak of the AP has reached half-way to each end of the 9cm axon. All graphs are plotted over distance x. These results were NOT expected, please see the next page for an explanation.

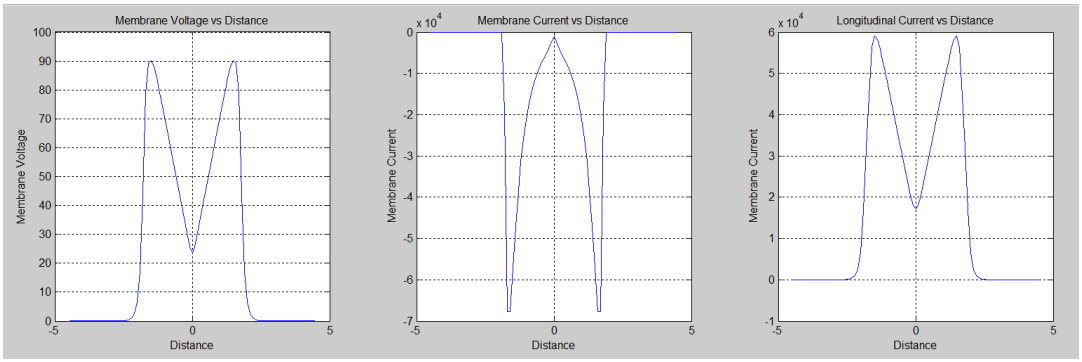

Figure 3: Membrane Voltage & Current and Longitudinal Axonal Current at the instant the APs have propagated half way down the to the axon's ends.

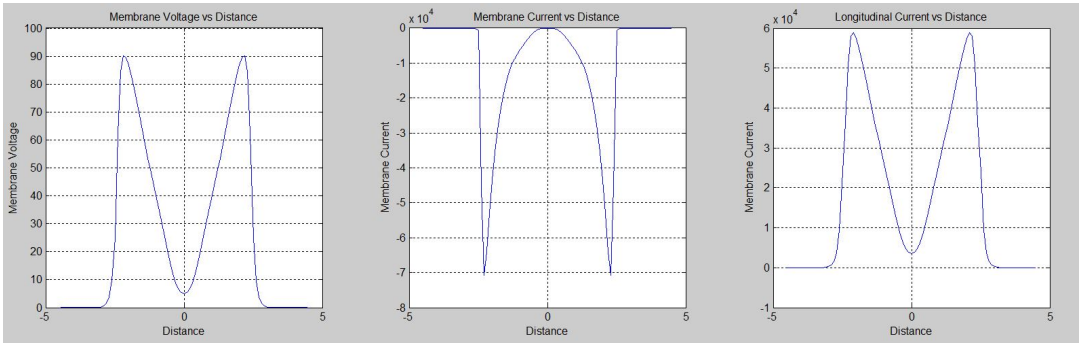

Figure 4: Membrane Voltage & Current and Longitudinal Axonal Current at the instant the AP peaks have propagated half way down the to the axon's ends.

These results were not expected. For the membrane current, the negative current indicates an influx of sodium which does make sense,being strongest under the peaks of the APs where sodium is rushing in. However I would have expected a small positive current (indicating an efflux) before and after the AP peak, where the sodium flows out before and after the propagating barrier. Something like this:

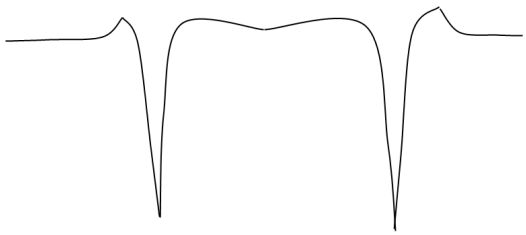

Figure 5: Expected Membrane Current.

Then, with the longitudinal current, a slight backwards current after the propagation front meant I expected it to look more like this:

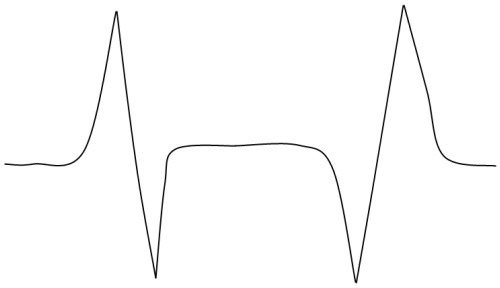

Figure 6: Expected Longitudinal Current.

I believe that the equations I was using were valid (see matlab code) so I was unable to track down the reasons for this discrepancy between expected and simulated results.

Nodes of Ranvier allowing inward membrane current:
Calculating this value seemed irrelevant given my completely unexpected results for the membrane current graphs. However the method is quite simple and I can describe it here. The inward sodium current would manifest as a negative current on the graph of membrane current vs. distance. Additionally, each inter nodal length was provided as 1mm. Assuming negligible nodal length that yields a number of nodes equivalent to the number of points in x. So simply calculating the number of points in x valued below 0 (negative current) yields the number of nodes of Ranvier allowing inward current. To do this is matlab, a simple find command on vector x with a comparison to 0 and then a length command to find the number of points would yield an answer.

Uni-Directional Action Potential:
To elicit a uni-directional action potential, 2 electrode were used. The first stimulus electrode released a cathodic current stimulus of -300μ-amps a distance of 1mm from the axon. The second stimulus electrode released an anodic current stimulus of 300μ-amps a distance of 1mm from the axon and from a stimulus electrode placed 1mm away from the axon (z=1mm) and 1mm to the side of the stimulating cathode. Depending on the side the anode is placed, the action potential's propagation will be blocked in one of the two directions. A single version of this is shown below with the action potential propagating to the right.

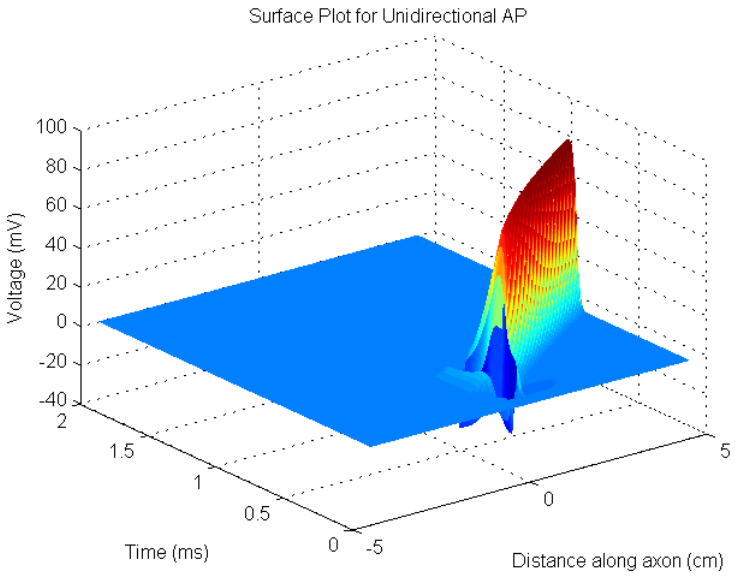

Figure 7: Surface Plot of Uni-Directional AP propagating the rightward direction.

Uni-Directional AP Movie:
An MPEG-4 video of the uni-directional AP was uploaded to BlackBoard.

MATLAB Code (Bi-Directional AP):
The code used for the bi-directional AP is shown below:

```
% this does not account for temperature effects

clear all; close all; clc;



% Define constants:
dt = 0.001;              % the step size for the timestep
t = 0:dt:2;              % the total duration of the calculations is 2 ms
                         % Note: any stimulus should be removed after 1.1 ms

dx = 0.1;                % step size for the distance along the axon.
                         % In class Moran, stated the x vector should be
                         % 900 elements long...
                         % but this is not congruent with using a step size
                         % equal to the internodal gap (1mm) which makes 90
                         % slices for a 9cm long axon.

x = -4.5:dx:4.5;         % the vector of slices to be used
                         % 9cm total, centered around 0

Re = .350;
Ri = .150;
Cm = 2.5;
a = 10E-4;
l = 1E-4;

gNa = 1445;
gL = 128;
Ena = 115;
El = -0.01;

%%%% vary the values for the cathodic_pulse_current_ampltitude until the
%%%% surface plot shows a generated AP... hone in a threshold.

% cathodic_pulse_current_amplitude = 144; %before min threshold for z = 1mm
cathodic_pulse_current_amplitude = 145; %after min threshold for z = 1mm
% cathodic_pulse_current_amplitude = 500; %before min threshold for z = 2mm
% cathodic_pulse_current_amplitude = 501; %after min threshold for z = 2mm
% cathodic_pulse_current_amplitude = 2113; %before min threshold for z = 4mm
% cathodic_pulse_current_amplitude = 2114; %after min threshold for z = 4mm
% cathodic_pulse_current_amplitude = 11242; %before min threshold for z = 8mm
% cathodic_pulse_current_amplitude = 11243; %after min threshold for z = 8mm


% Define the stimulation current as a vector the same length as time. To
% make a simple square pulse, a constant value for a fixed time is sufficient.
% Include an anodic pulse from 1ms to 9ms (8ms total)
% Include a cathodic pulse from 9ms to 11ms (2ms total)
% The anodic pulse should be 1/4 the magnitude of the cathodic pulse.
Ist = zeros(size(t));
Ist(100:900) = 0.25*cathodic_pulse_current_amplitude;  % 0.8ms anodic pulse
Ist(900:1100) = -cathodic_pulse_current_amplitude; % 0.2ms cathodic pulse


% z represents the perpendicular distance between the axon and the
% stimulating electrode. (make sure that the corresponding cathodic current
% ampltitude is used for the right distance "z" or the script will error out.
z = 0.1;
% z = 0.2;
% z = 0.4;
% z = 0.8;


% Define a matrix to quantify the Extracellular Potential at every point
% along the length of the axon "length(x)" for every instant in time
% "length(t)"
```

```
Ve = zeros(length(t), length(x));
for i = 1:length(t)
    for j = 1:length(x)
        Ve(i,j) = (Re * Ist(i))/(4*pi*sqrt(x(j)^2 + z^2));
    end
end

% Define the matrix for membrane current which will only be used for 1
% timestep (prob 3)
im = zeros(1,length(x));
% Define the matrix for longitudinal axonal current which will only be
% used for 1 timestep (prob 3)
i_longitudinal = zeros(1,length(x));

%Define matrices to hold the membrane potential, m and h
V = zeros(length(t), length(x));
m = zeros(length(t), length(x));
h = zeros(length(t), length(x));

%boolean to keep track of the special plots, so we don't repeat them
should_plot_voltage = 1;
should_plot_voltage2 = 1;
index_half_way_to_end = 68;


%loop through time... at each point in time, consider every point along the
%axon in x with a nested inner loop
for i = 1:length(t)

    % now loop through each section of the
    for j = 2:(length(x)-1)
        Vm = V(i,j);        % define a temp variable to hold this timestep's
                            % membrane voltage for the current slice in the axon
                            % just to help keep things neat later.

        % define the alpha constants based off the formulas provided in the hw
        alpha_m = (97+0.363*Vm)/(1+exp((31-Vm)/5.3));
        beta_m = alpha_m/exp((Vm-23.8)/4.17);
        beta_h = 15.6/(1+exp((24-Vm)/10));
        alpha_h = beta_h/exp((Vm-5.5)/5);

        % define the change in m for the current timestep, then udpate the
        % next timestep's m value by adding the change
        dmdt = -(alpha_m + beta_m)*m(i,j) + alpha_m;
        m(i+1,j) = m(i,j) + dmdt*dt;

        % define the change in h for the current timestep, then update the
        % next timestep's h value by adding the change
        dhdt = -(alpha_h + beta_h)*h(i,j) + alpha_h;
        h(i+1,j) = h(i,j) + dhdt*dt;

        % define the change in membrane voltage for the current timestep,
        % then update the next timestep's membrane voltage value by adding
        % the change.
        dVdt = (-gNa*m(i,j)^2*h(i,j)*(Vm-Ena) - gL*(Vm-El) + ((2*a*dx)/(4*Ri*l))*(((V(i,j-1)-2*Vm+V(i,j+1))/dx^2) + (
        V(i+1,j) = V(i,j) + dVdt*dt;

    end

    %Plot the voltage along the axon at the instant the AP propogates
    %halfway to the ends of the axon.
    if(V(i,index_half_way_to_end)>1)
        if(V(i,index_half_way_to_end)>90 && should_plot_voltage2 == 1)
            %plot membrane voltage
            figure;
            subplot(1,3,1);
            plot(x,V(i,:));
            grid on; xlabel('Distance'); ylabel('Membrane Voltage');
            title('Membrane Voltage vs Distance');

            %plot membrane current
            subplot(1,3,2);
```

```
            for j = 2:(length(x)-1)
                im(j) = -gNa*m(i,j)^2*h(i,j)*(Vm-Ena) - gL*(Vm-El);
            end
            plot(x,im(1,:));
            grid on; xlabel('Distance'); ylabel('Membrane Current');
            title('Membrane Current vs Distance');

            %plot longitudinal axonal current
            subplot(1,3,3);
            for j = 2:(length(x)-1)
                i_longitudinal(j) = ((2*a*dx)/(4*Ri*l))*((V(i,j-1)-2*Vm+V(i,j+1))/dx^2);
            end
            plot(x,i_longitudinal(1,:));
            grid on; xlabel('Distance'); ylabel('Membrane Current');
            title('Longitudinal Current vs Distance');


            should_plot_voltage2 =0;
        elseif (should_plot_voltage == 1)
            %plot membrane voltage
            figure;
            subplot(1,3,1);
            plot(x,V(i,:));
            grid on;  xlabel('Distance'); ylabel('Membrane Voltage');
            title('Membrane Voltage vs Distance');

            %plot membrane current
            subplot(1,3,2);
            for j = 2:(length(x)-1)
                im(j) = -gNa*m(i,j)^2*h(i,j)*(Vm-Ena) - gL*(Vm-El);
            end
            plot(x,im(1,:));
            grid on; xlabel('Distance'); ylabel('Membrane Current');
            title('Membrane Current vs Distance');

            %plot longitudinal axonal current
            subplot(1,3,3);
            for j = 2:(length(x)-1)
                i_longitudinal(j) = ((2*a*dx)/(4*Ri*l))*((V(i,j-1)-2*Vm+V(i,j+1))/dx^2);
            end
            plot(x,i_longitudinal(1,:));
            grid on; xlabel('Distance'); ylabel('Membrane Current');
            title('Longitudinal Current vs Distance');

            should_plot_voltage = 0;
        end
    end

end

% plot the voltage in both time and x... showing the AP.
figure
surf(x, [0 t], V), title('Surface Plot for Bidirectional AP'), xlabel('Distance along axon (cm)'), ylabel('Time (ms)'
shading flat

% Make a video of the AP.

% video = VideoWriter('BidirectionalAP---Z_1mm.mp4');
% open(video);
% figure
% for i = 1:length(t)
%     plot(x, V(i,:));
%     axis([-4.5 4.5 -25 100]);
%     title('Bidirectional Action Potential'), xlabel('Distance along axon (cm)'), ylabel('Voltage (mV)');
%     mov(i) = getframe(gcf);
%     writeVideo(video, mov(i));
% end
```

MATLAB Code (Uni-Directional AP):
The only noteworthy changes to the bi-directional code are outlined below:

```
Ist1 = zeros(size(t));
Ist2 = zeros(size(t));
Ist1(100:200) = -300;
Ist2(100:500) = 300;
Ve = zeros(length(t), length(x));
% Ve1 = zeros(length(t), length(x));
% Ve2 = zeros(length(t), length(x));

z = 0.1;

for i = 1:length(t)
    for j = 1:length(x)
        Ve1 = (Re * Ist1(i))/(4*pi*sqrt(x(j)^2 + z^2));
        Ve2 = (Re * Ist2(i))/(4*pi*sqrt((x(j)+0.1)^2 + z^2));  %only propogate to right
%         Ve2 = (Re * Ist2(i))/(4*pi*sqrt((x(j)-0.1)^2 + z^2));  %only propogate to left
        Ve(i,j) = Ve1 + Ve2;
    end
end
```