BIOMEDE 517 - Neural Engineering - Labs 3 and 4 - Dr. Cindy Chestek

Implications of Electrostatic Modeling of Neurons and Heads

Kushal Jaligama

1 - Introduction

Epilepsy, Parkinsons, and general tremor patients exhibit overactive neuronal signaling and can be aided through stimulating nervous tissue to block excessive signal propagation. By understanding the conductivity environments of neural tissue, one can effectively identify what kinds of currents should be applied in order to regulate the firing of neurons in certain brain regions. This laboratory walks through simulating different head models in COMSOL, as well as analysis on how to stimulate and record signals from brain tissue. Exploring the electrical interactions between different brain regions reveals insight onto resultant signals. Modeling these interactions is important because treatment of tremor conditions should be localized to the correct brain regions to stop or promote signal propagation in another region.

2 - Methods

2.1 - Assumptions, Designs, and Complexities of COMSOL Head Models

In this laboratory, three different head models were explored. The first model was constructed with parameters from (Moffit and McIntyre, 2005), modeled as four different regions in COMSOL, namely the skull, scalp, cerebrospinal fluid, and brain. These parts had conductivities of 8000, 8500, 9000, 10,000 um, respectively, and were scaled 10x to resemble a human head. A single point current with a charge of 1 amp acted as an electrode. This configuration allowed COMSOL to calculate voltages in the isotropic conductivity environment of a scaled rat head model from (Moffit and McIntyre 2005), which were then exported to MATLAB for analysis. This data is discussed in section 3.1 and visualized in (apdx: Fig 4). The next iteration of this model added two point current sinks 3 mm above and below the source current, giving the model a configuration representing that of Deep Brain Stimulation. This model utilized an isotropic conductivity environment.

Typical conductivity environments are not uniformly isotropic, as they were in the previous model. One can simulate an anisotropic model of conductivity by incorporating tensor data from diffusion-weighted MRI scans. In this case, tensor data of the thalamus from scans of a patient with essential tremors were converted using MATLAB from diffusion tensors to conductivity tensors (Tuch, et. al. 2001) and then incorporated into the COMSOL environment. Each voxel in space uses this tensor matrix:

$$\begin{bmatrix} S_{11} & S_{12} & S_{13} \\ S_{21} & S_{22} & S_{23} \\ S_{31} & S_{32} & S_{33} \end{bmatrix}, \qquad where\ S_{ij} = S_{ji}$$

2.2 - Assumptions, Designs, and Complexities of Analytical Neural Recordings and Stimulations

To gather simple extracellular recordings, a layer V pyramidal neuron and a layer II cortical neuron were modeled in the NEURON software suite. Analysis of these compartmental currents in MATLAB revealed that an electrode at distances of

50, 100, 200, and 300 um away from the hillock could detect external voltages produced as action potentials propagated through compartments. These voltages can be calculated using the distance between an electrode and a compartment, magnitude of stimulus, and conductivity (apdx: Table 1, #1). The superposition of voltage is the sum of all external voltages produced by each compartment, which is equivalent to the net voltage trace that a chosen point (at which an electrode is located) would record from firing of the neuron.

To build on this model, ten neurons were randomly distributed over a 100 um cube centered around a recording site at the origin. The electrode recorded the superposition of voltage over all compartments of all neurons, which have a variable firing rate between 2 and 50 Hz over a 1s period. The model also ensures that action potentials within a single neuron that do not overlap, and that action potentials from multiple neurons do not happen at exactly the same time. This model was made more complex through thermal noise, which is generated by thermal motion of ions. Thermal noise can be calculated using resistances of the electrode and surrounding tissue (apdx: Table 1, #2). A simple pink noise waveform was added via MATLAB to the data samples to simulate this thermal noise. Often, inserting an electrode into conductive tissue causes scarring, resulting in a “dead zone” immediately around the electrode. Measurements of voltage at the electrode were taken both including and excluding neural tissue within 50 um of the electrode.

The next model contained a neuron with its hillock placed 50 um from the origin in the Y direction and an electrode placed at the origin. To account for the conductivity environment, voltage gradient information exported from the single point current simulation described in section 2.1 was used to calculate a K matrix that contains voltages detected around a 1A current in space (apdx, Table 1, #3). This matrix was used to scale the current at each compartment of the neuron, yielding the superposition of voltage produced by the compartmental currents.

Stimulating neural tissue by injecting an extracellular current causes an intracellular current to flow through a neuron. Each compartment in the neuron has a characteristic external voltage based on its current (apdx: Figure 13). In a previous laboratory, the NEURON software tool was used to simulate simple multi-compartmental axons. Each of these axons had a necessary threshold current that would trigger an action potential along the axon. The Warman Equation (Warman 1992) (apdx: Table 1, #4) yields the intracellular currents for each compartment of the neuron, based on the external voltage generated by each compartment and a constant representing the internodal conductance of the axon, namely $g_a=3x10^{-5}$.

3 - Results

3.1 - Relationship of External Voltages and Distance from Single Point Source Current

COMSOL calculated voltage traces from a point source current using an equation that relates voltage and distance (apdx: Table 1, #1). At various distances, one can see that the resultant voltage of a point charge varies with distance along a curve that resembles y = 1/x (apdx: Fig 1). Essentially, the voltage generated by a current decays rapidly as the distance from the stimulus increases.

3.2 - Implications of Increasing Model Complexity and Second Spatial Derivative of Voltage

Deep Brain Stimulation models also have characteristic voltage gradients (apdx: Fig 5). These voltages can once again be modeled with the Voltage and Distance Relationship Equation (apdx: Table 1, #1), and it is evident that in cases where there are multiple points of current, voltage decays as distance from the source current increases (apdx, Fig 2).

Plotting the second derivative of the voltage, perpendicular to the line created by the source and sink points shows a region of hyperpolarization in the vicinity of the source current and regions of depolarization in the vicinity of the sink currents (apdx: Fig 3). This curve represents the Activation Function of a neuron, which displays the intracellular current response as a result of the extracellular stimulus.

3.3 - Results of Anisotropic Deep Brain Stimulation

Creating a deep brain stimulation model with an anisotropic conductivity environment changes the electric field. Analyzing data from a patient's diffusion-weighted MRI scan in MATLAB reveals that voltage decays as distance from the stimulus current increases (apdx: Fig 9). The second spatial derivative (apdx: Fig 10), perpendicular to the electrode, reveals the Activation Function of a neuron. Anisotropic models follow the same rules as isotropic models, but the tensors that describe the conductivity environment change the electric field at each voxel. The anisotropic voltage gradient in COMSOL can be seen in (apdx: Fig 12).

3.4 - Results of Analytical Model of Neural Recording

Placing an electrode at a point perpendicularly 50 um away results in a greater voltage peak when measuring an action potential than when the measurement is done from 300 um away (apdx: Fig 6, Fig 7). In this Fig, as an electrode is placed at a further and further distance, the peak of the action potential superlinearly decreases. This decaying characteristic is true regardless of neuron diameter and follows the Voltage and Distance relationship (apdx: Table 1, #1).

3.5 - Electrode Recordings of Multiple Neurons

Randomly distributing ten neurons causes the compartments of each neuron to have a characteristic external voltage. Plotting the resultant superposition of voltage over time yields (apdx: Fig 8). Thermal noise is represented as the overlaid green waveform. One can see that including spikes from the dead zone causes randomized larger spikes to appear throughout the waveform. This happens because external voltages are generated by the greater number of neurons in close proximity, ultimately influencing the superposition of voltage.

3.6 - Extracellular Voltage Based on Conductivity Environment

After simulating a 1A current in a conductivity environment in COMSOL, a resultant voltage trace matrix along a volume conductor is generated. This voltage matrix is related to the conductivity of each point, and can be used as a scalar to calculate the voltage of an extracellular point based on the currents of a neuron's compartments. The waveform in (apdx: Fig 11) shows that the voltage over time at an electrode 50 um from the axon hillock resembles an action potential.

3.7 - Minimum Current Necessary to Fire an Axon 1 mm Away

The minimal current necessary to fire a an axon from a distance of 1 mm can be calculated by dividing the maximum current along the compartments of the axon by the threshold current required to fire the neuron by direct injection. In this situation, the minimum current required to fire an axon from 1 mm away is -185.6662 mA.

3.8 - Effects of Increasing Model Complexity and Implications for DBS

Changing parameters such as conductivity of a region, magnitude of stimuli, size and location of electrodes, and noise in electrostatic models of the brain leads to greater accuracy of stimulus-response data. These parameters can be scaled and applied to an ever-increasing number of neurons. Increasing the accuracy and scale of these models allows researchers to more effectively predict the effects Deep Brain Stimulation. Observing real world conductivities gives insight into the which regions of the brain will be affected by DBS electrodes, and how significant those effects are. Factoring thermal noise into the model shows that electrodes should be designed with impedances to gather clean recordings of neurons. Changing electrode size and location impacts which parts of the neural tissue will have scarring and how recordings change based on neuron density. These all affect DBS designs, changing the level of care that patients with movement and psychiatric disorders receive.

4 - Discussion

After understanding electrostatic models of neural tissue, one can begin to effectively model the manner in which different regions of the brain interact with one another. Stimulus currents at one location result in external voltages at other parts of neural tissue, so the findings of this lab could lead to electrode design that allows current flow at regions otherwise physically difficult to pinpoint. Machine learning could use these models to generate a map of region interactions, giving insight about neural processes and their development.

References

Moffitt, Michael A., and Cameron C. Mcintyre. “Model-Based Analysis of Cortical Recording with Silicon Microelectrodes.” *Clinical Neurophysiology*, vol. 116, no. 9, 2005, pp. 2240–2250., doi:10.1016/j.clinph.2005.05.018.

Tuch, D. S., et al. “Conductivity Tensor Mapping of the Human Brain Using Diffusion Tensor MRI.” *Proceedings of the National Academy of Sciences*, vol. 98, no. 20, 2001, pp. 11697–11701., doi:10.1073/pnas.171473898.

Warman, E.n., et al. “Modeling the Effects of Electric Fields on Nerve Fibers: Determination of Excitation Thresholds.” *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 12, 1992, pp. 1244–1254., doi:10.1109/10.184700.

BIOMEDE 517 - Neural Engineering - Labs 3 and 4 - Dr. Cindy Chestek

Appendix

Kushal Jaligama

Table 1 - Equations Used for Electrostatic Modeling

#	Equation Name	Equation
1	Voltage and Distance Relationship	$V_{ext} = \frac{I_0}{4\pi\sigma r}$
2	Thermal Noise as a Result of Ion Thermal Motion When Recording From Electrodes	$V_n^2 = 4kTR\Delta f$
3	External Voltage as Result of K Matrix Generated from Variable Conductivity Environment	$V_{ext}(x,y,z) = I_0(x,y,z) \times K(x,y,z)$
4	Warman, Grill, and Durand Equation to Calculate Equivalent Intracellular Currents based on Injected Extracellular Current	$I_{\text{int}}(n) = G_a[V_e(n-1) - 2V_e(n) + V_e(n+1)]$

Figure 1 - Relationship Between Voltage and Distance for a Single Point Current Source

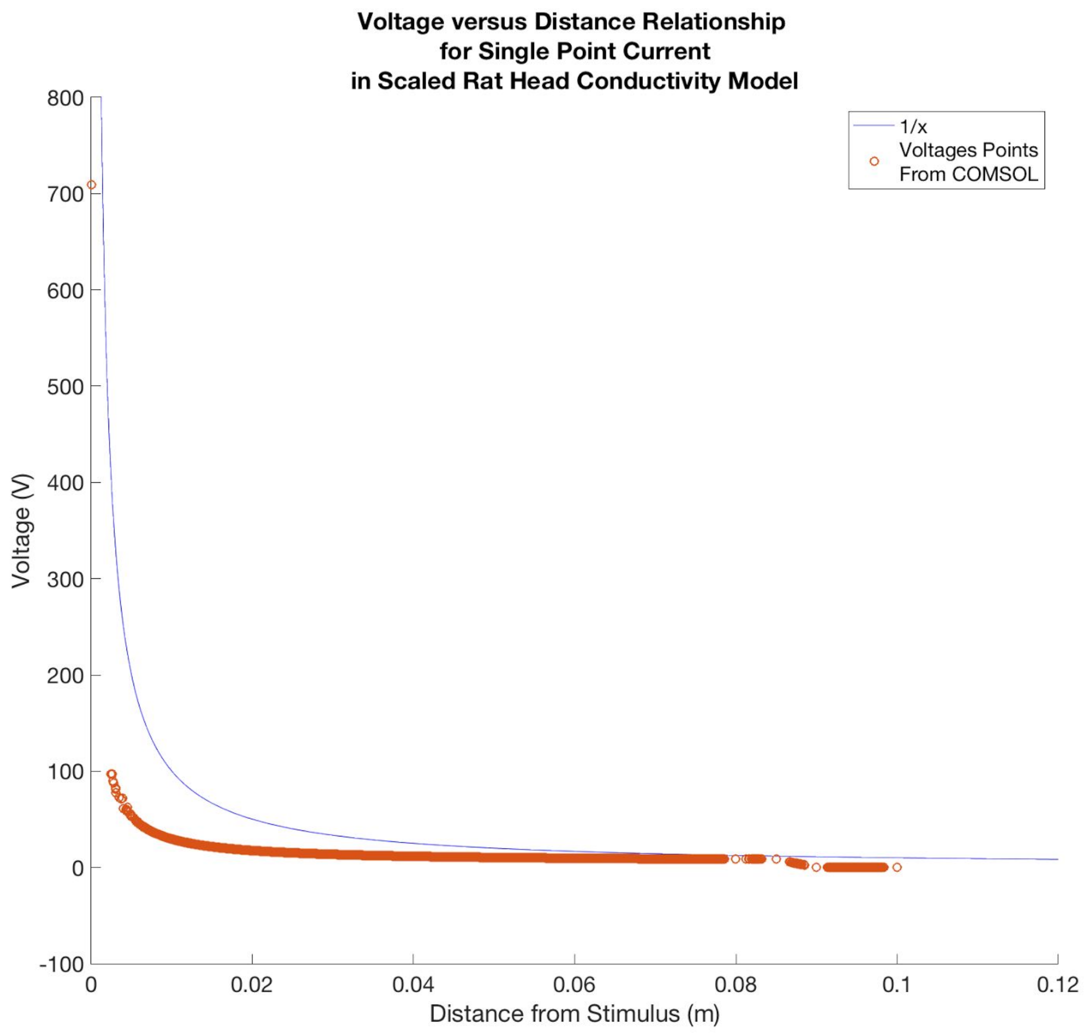


Figure 2 - Relationship between Voltage and Distance for a Deep Brain Simulation

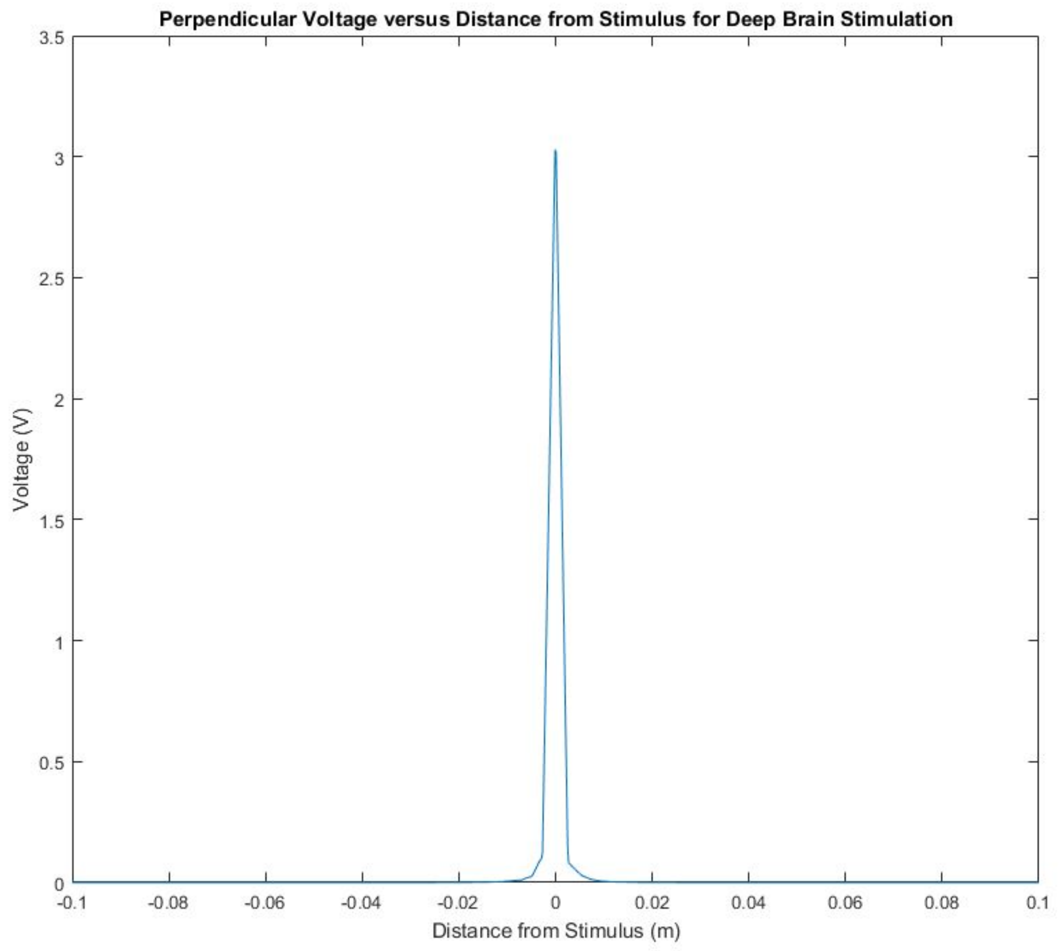


Figure 3 - Second Spatial Derivative of Deep Brain Stimulation

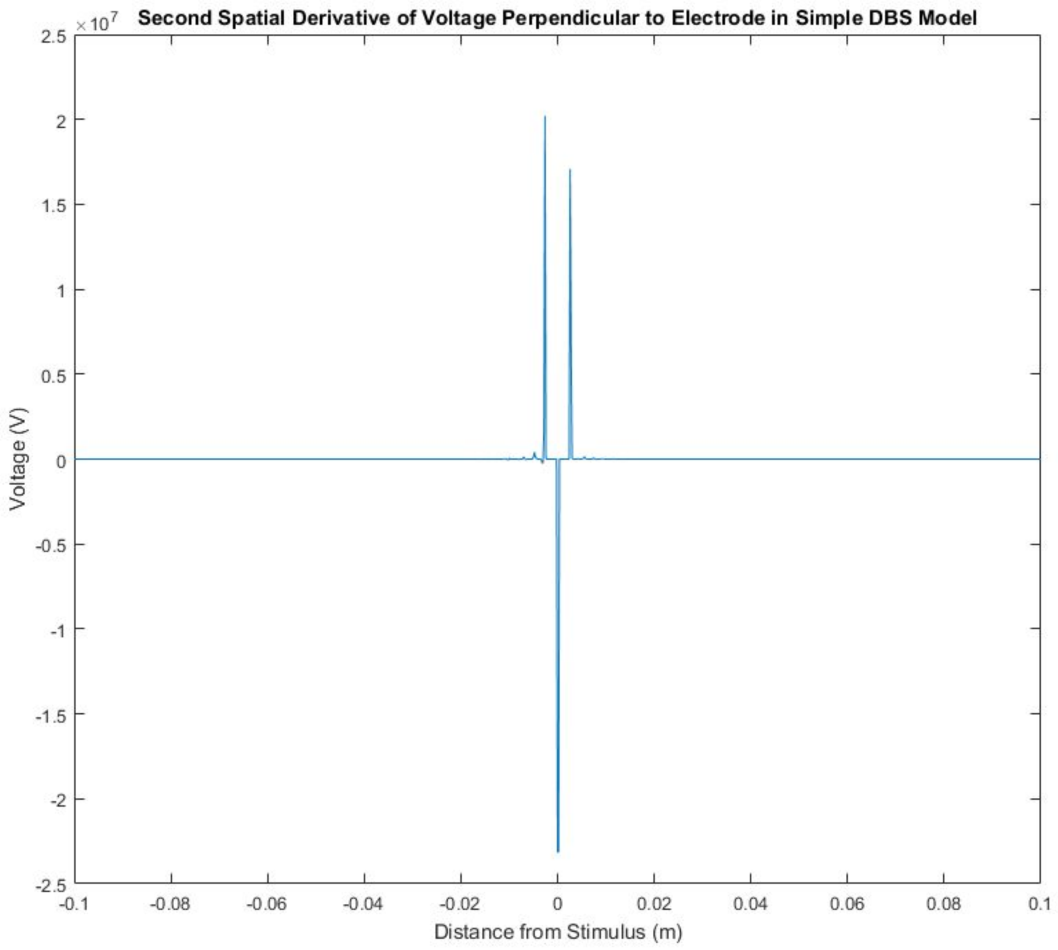


Figure 4 - Voltage Gradient of Isotropic COMSOL Model with 1 Source and 2 Sink Currents

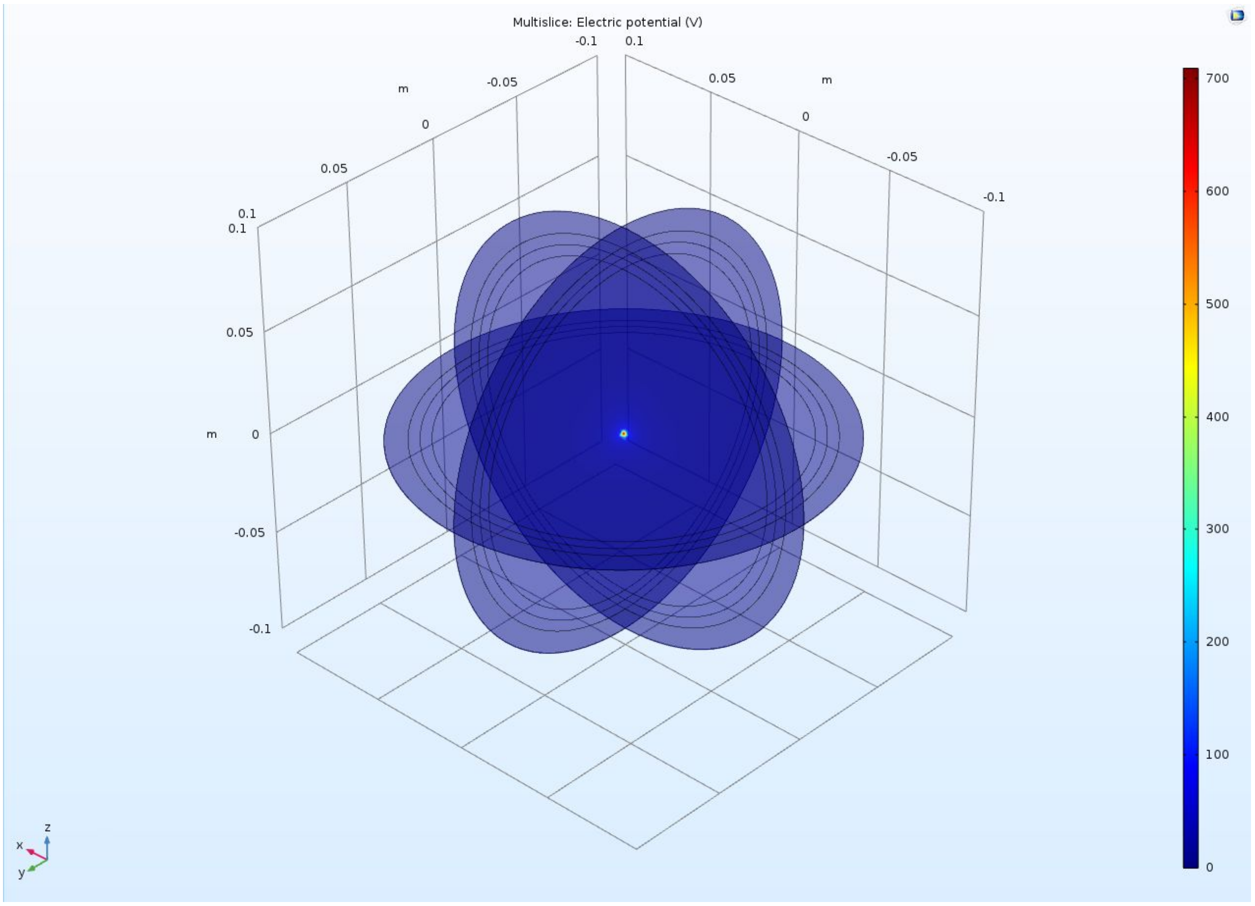
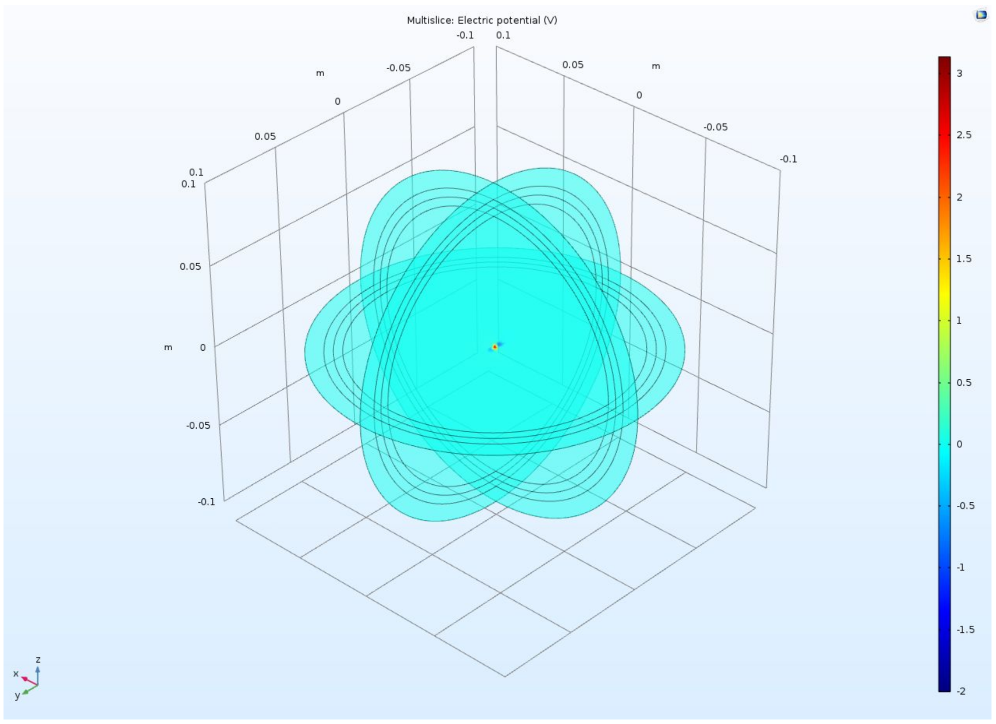


Figure 5 - Voltage Gradient of Anisotropic COMSOL Model with 1 Source and 2 Sink Currents



Zoomed In

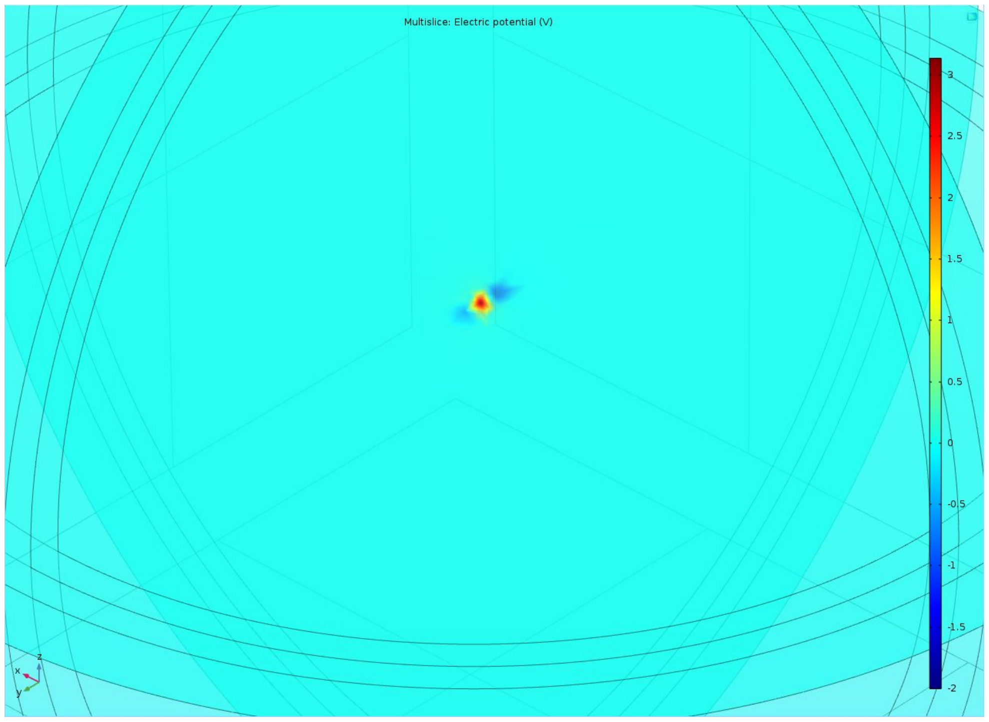


Figure 6 - Large Layer V Pyramidal Neuron - Superposition of Voltage with Electrodes at Various Distances

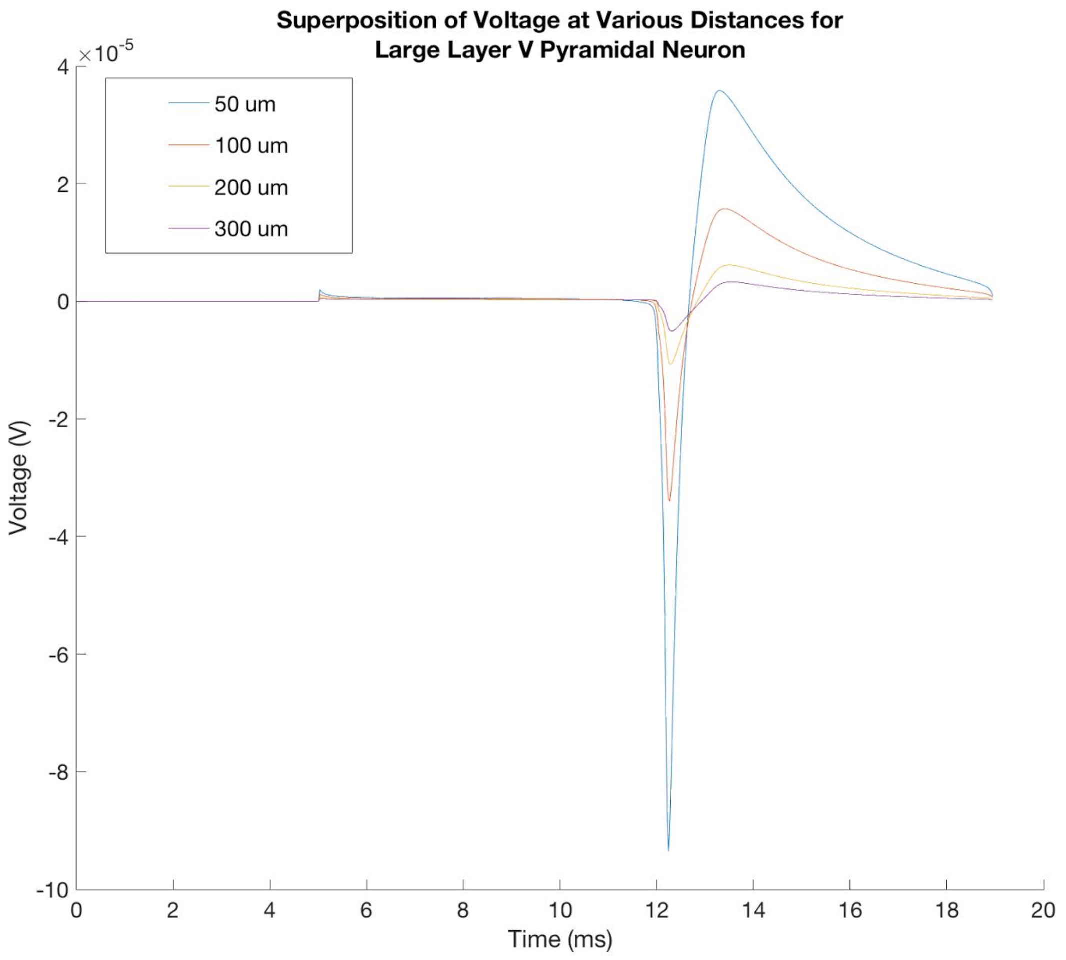


Figure 7 - Small Layer II Cortical Neuron - Superposition of Voltage with Electrodes at Various Distances

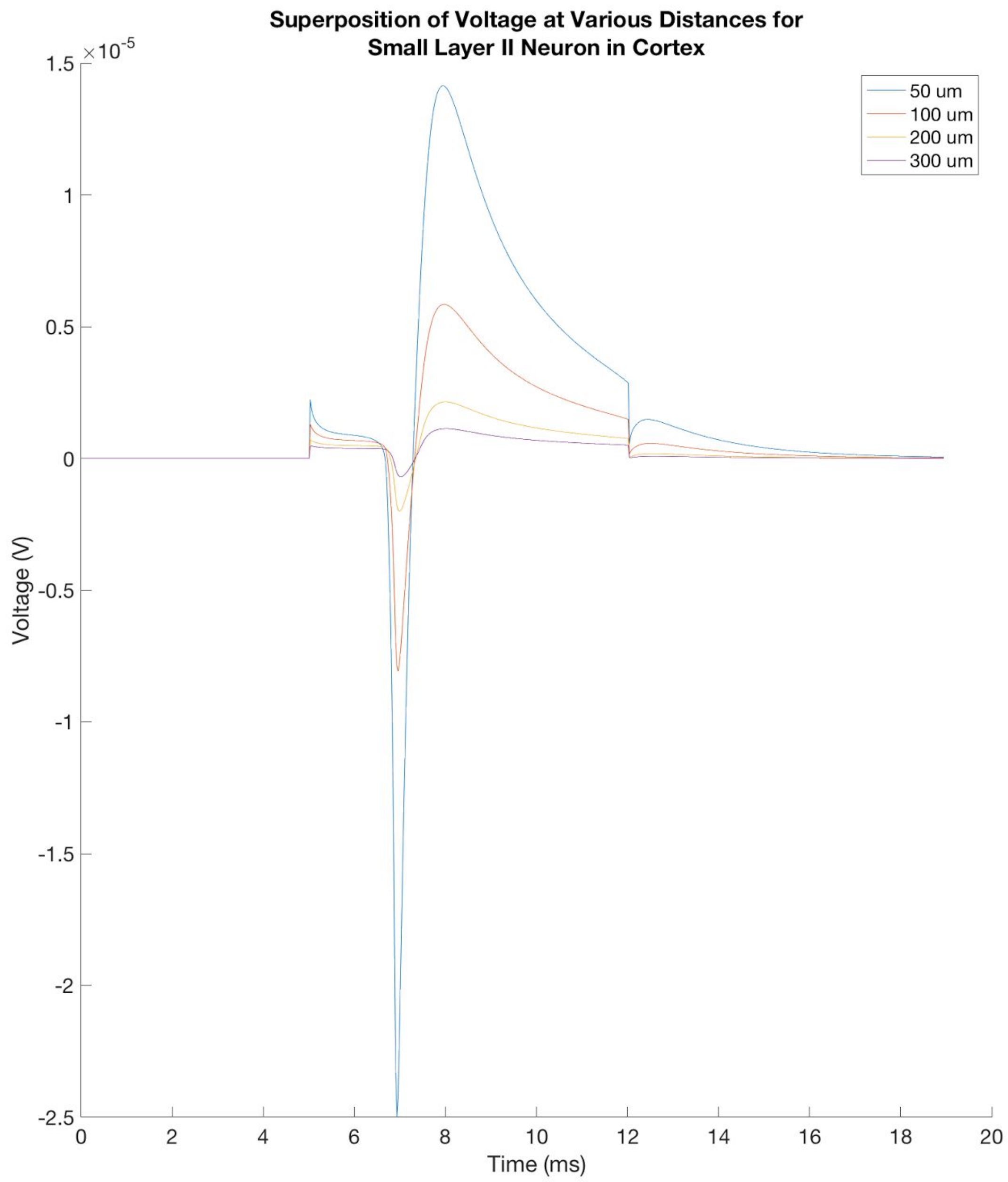


Figure 8 - Net Potential of Ten Neurons With and Without Deadzone

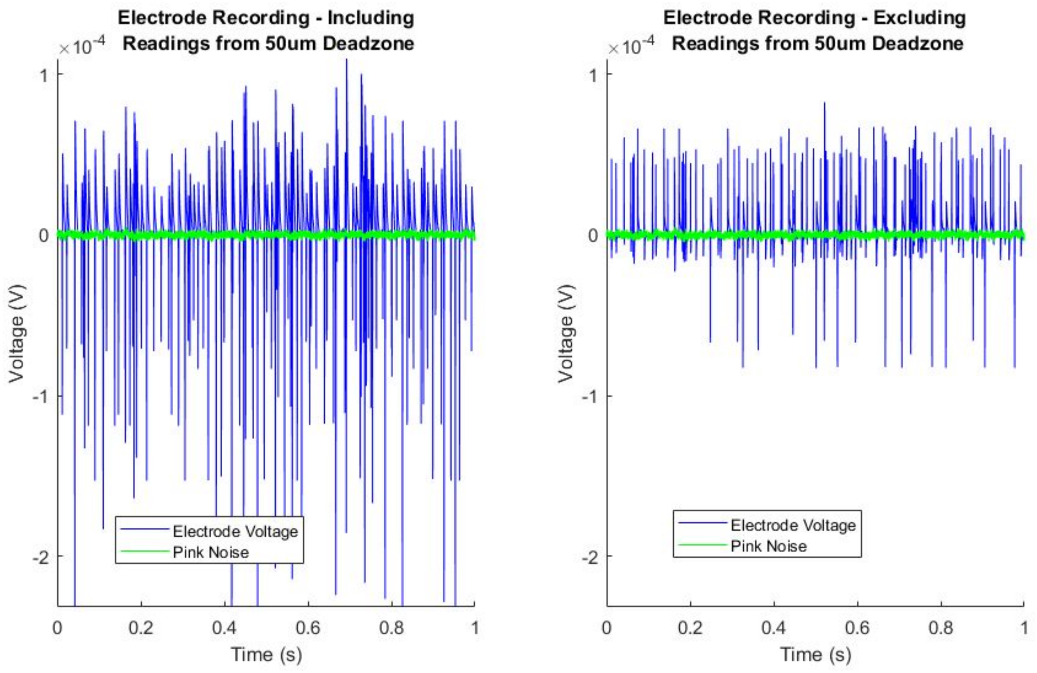


Figure 9 - Voltage at Varying Distances in Anisotropic DBS Model

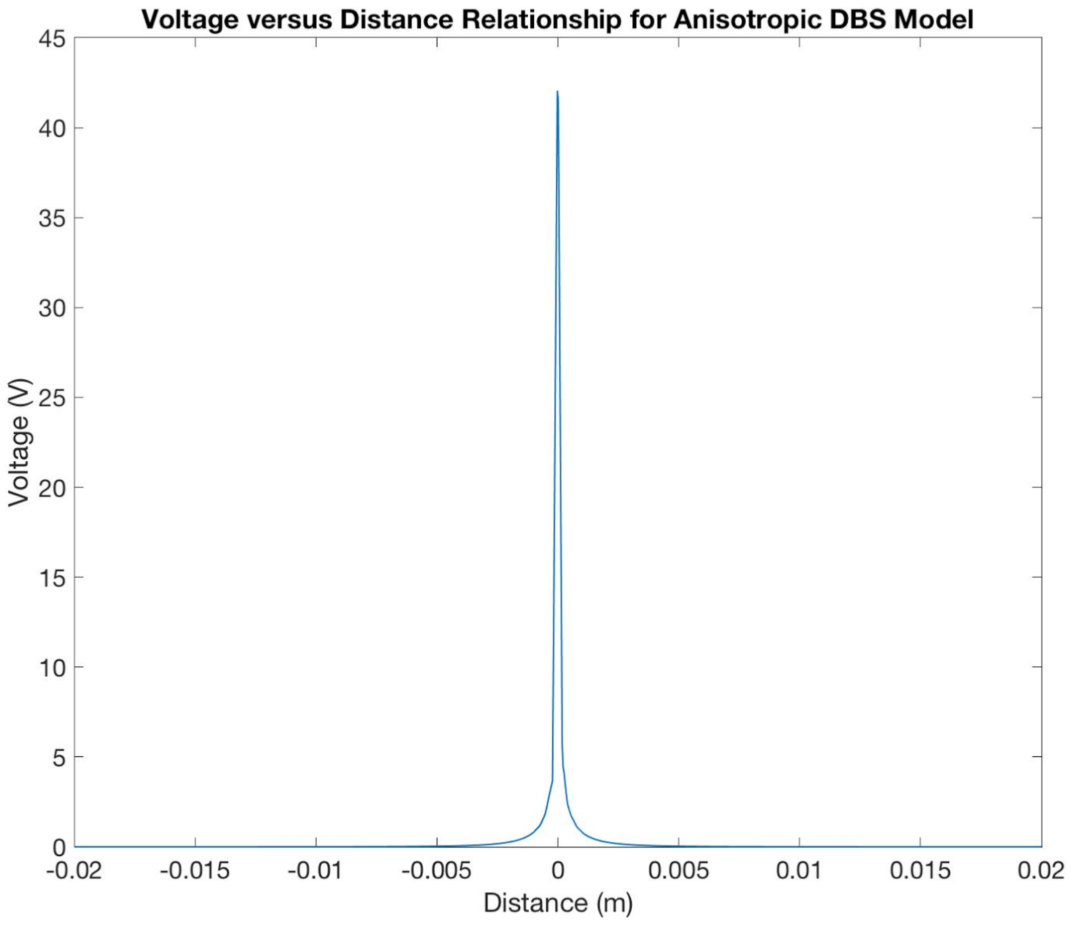


Figure 10 - Second Spatial Derivative of Voltage Perpendicular to Electrode in Anisotropic DBS Model

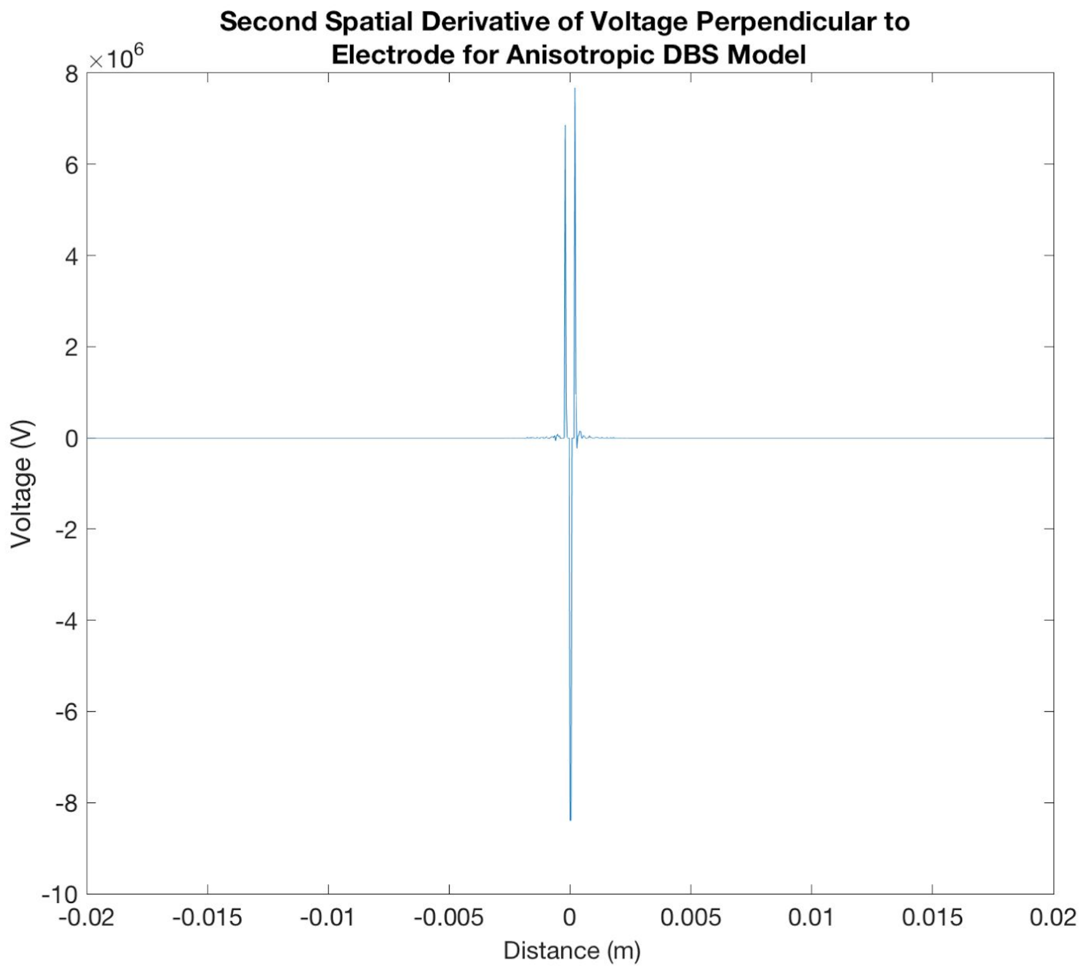


Figure 11 - Extracellular Voltage Waveform Based on K Matrix

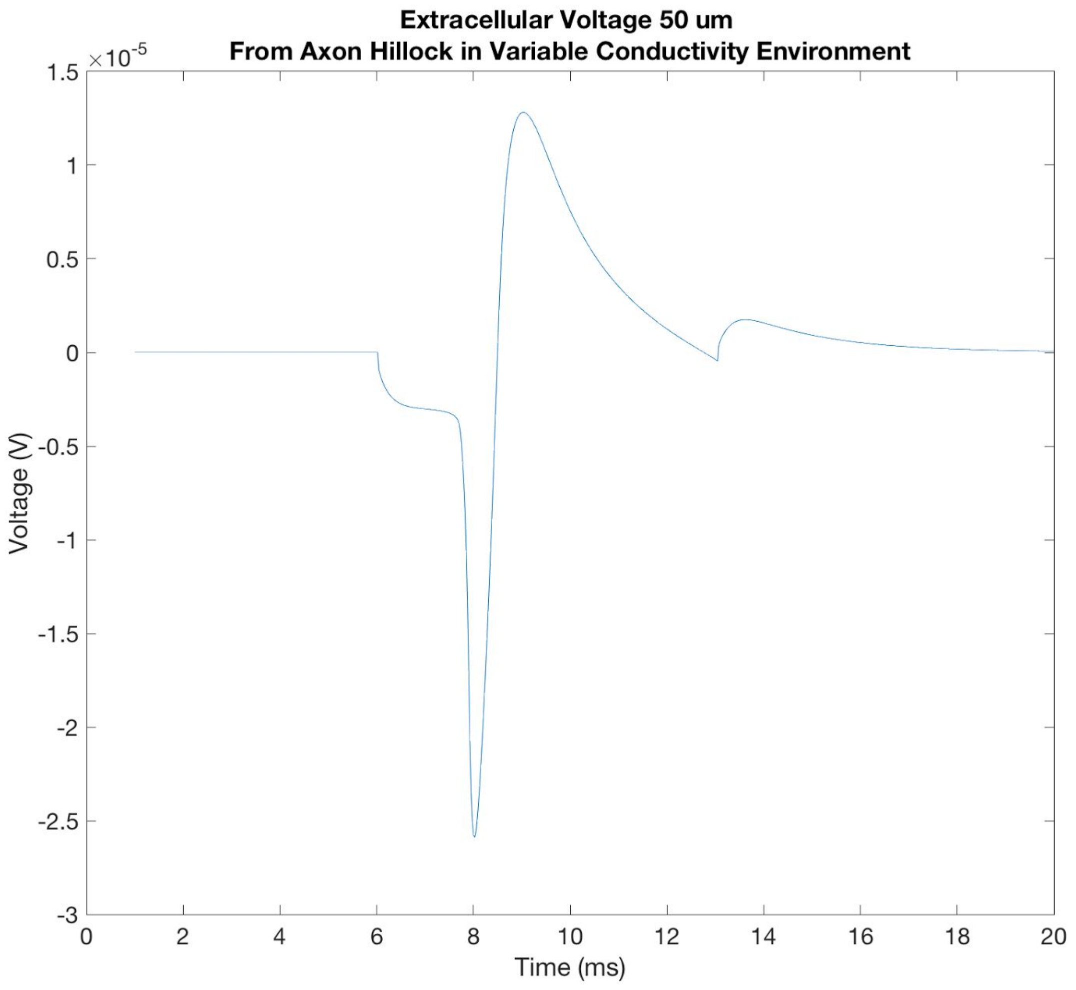


Figure 12 - Voltage Gradient of Anisotropic COMSOL Model with 1 Source and 2 Sink Current

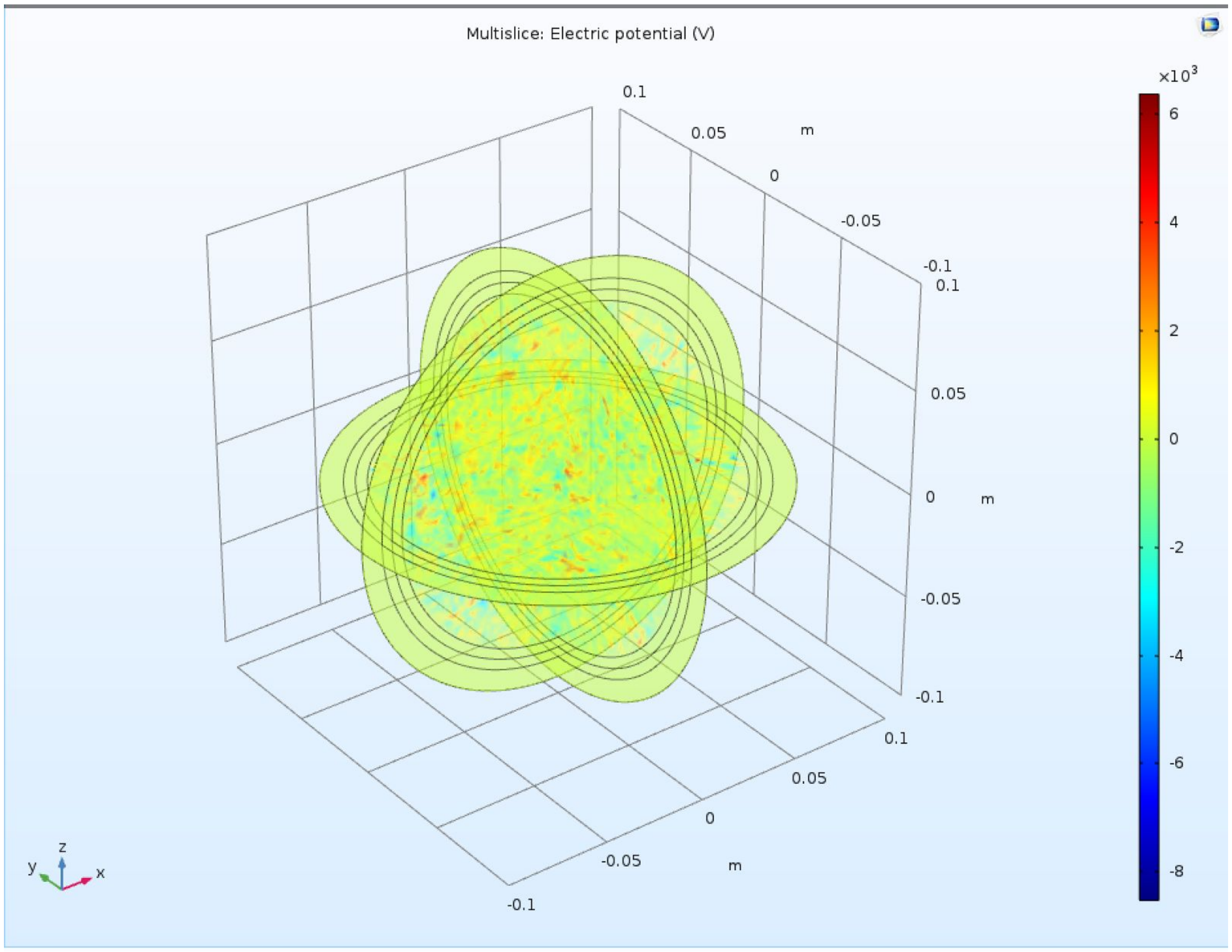
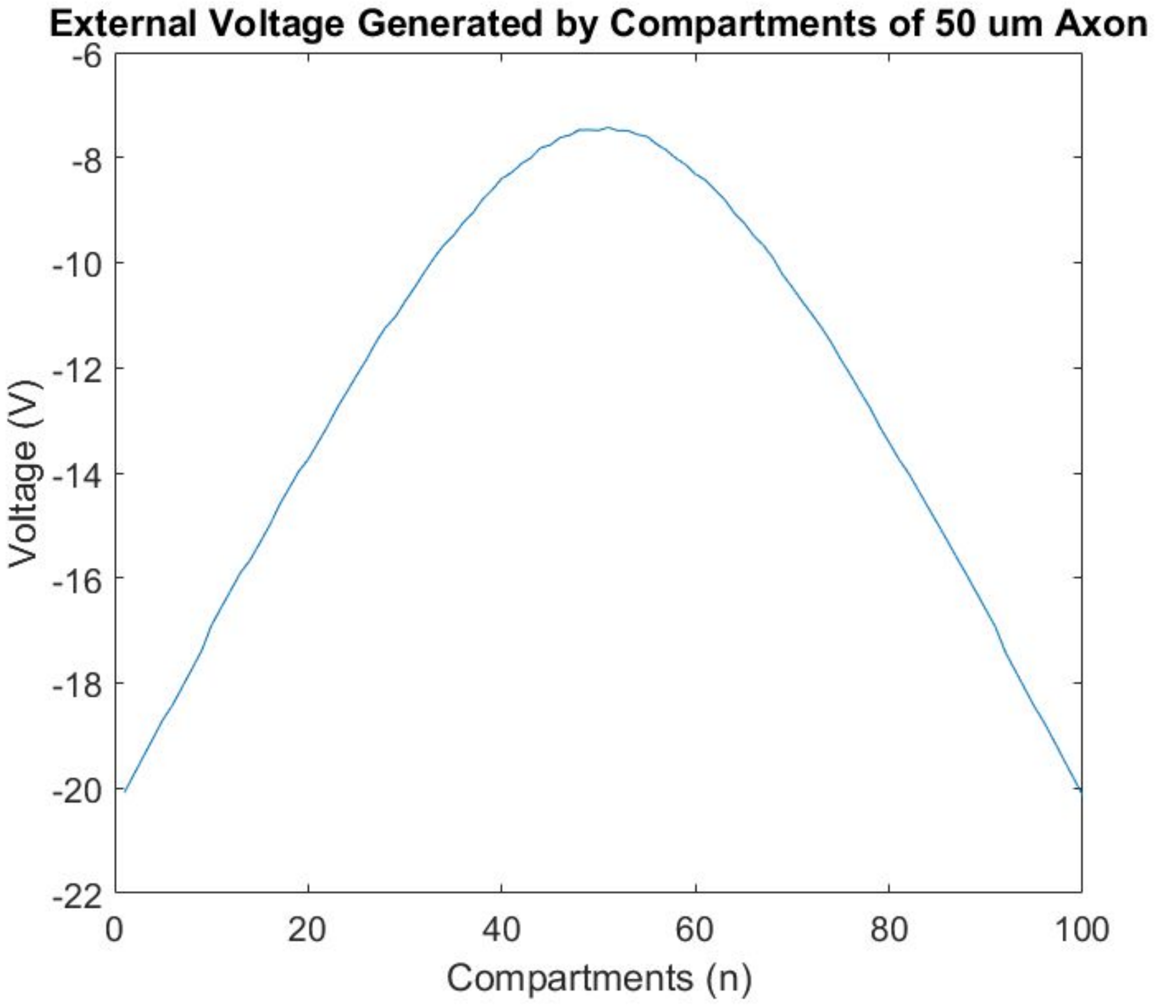


Figure 13 - Compartmental External Voltages 1 mm Away from a 50 um Axon



MATLAB Code Utilized for Analysis of Data in Lab 3

```matlab
% BIOMEDE 517 - Lab 3
% Kushal Jaligama

% Part 2 - Create a plot of voltage versus distance from stimulus for
% single point current source
Part2Data = importdata('part2export.txt', ' ');
Voltage = zeros(size(Part2Data(:,1)));
DistanceFromStimulus = zeros(size(Part2Data(:,1)));

for i = 1:size(Part2Data)
   Voltage(i) = Part2Data(i,4);
   X = Part2Data(i,1);
   Y = Part2Data(i,2);
   Z = Part2Data(i,3);
   DistanceFromStimulus(i) = sqrt((X-0)^2 + (Y-0)^2 + (Z-0)^2);
end

figure(1)
hold on
syms f(x)
f(x) = 1/x;
fplot(f, 'r')
scatter(DistanceFromStimulus, Voltage)
hold off


% Part 3 - Create a plot of voltage versus distance from stimulus for a
% Deep Brain Stimulation (DBS) model
Part3Data = importdata('part3export.txt');
DBSVoltages = zeros(size(Part3Data(:,1)));
DBSDistances = zeros(size(Part3Data(:,1)));

for i = 1:size(Part3Data)
  DBSVoltages(i) = Part3Data(i, 4);
  X = Part3Data(i,1);
  Y = Part3Data(i,2);
  Z = Part3Data(i,3);
  DBSDistances(i) = sqrt((X-0)^2 + (Y-0)^2 + (Z-0)^2);
end

% subplot(4,1,2)
% scatter(DBSDistances, DBSVoltages)

% Interpolate the data
interpolated = scatteredInterpolant(Part3Data(:,1), Part3Data(:,2), Part3Data(:,3), Part3Data(:,4));
x_coordinates = [transpose(linspace(-0.1, 0.1, 1000)), zeros(1000, 1), zeros(1000, 1)];
x_voltages = interpolated(x_coordinates);

figure(2)
plot(x_voltages);

% Interpolate data and calculate second spatial derivative of
% voltage in direction going away from the active contact and perpendicular
% to the probe (x-direction)

% [v(i+1)-2v(i)+v(i-1)] / (delta_z)^2)this is the second deriv eq

delta_z = 0.001;
for i = 2:size(x_voltages) - 1
   x_voltages_second_deriv(i) = (x_voltages(i+1) - 2*x_voltages(i) + x_voltages(i-1)) / delta_z^2;
end

figure(3)
plot(linspace(-0.1, 0.1, 999), x_voltages_second_deriv)
```

```
% Part 4 - Apply patient-specific anisotropic DBS model
% Export conductivity tensors to a form COMSOL can read, scale data
% importTensorData('tensor_data.mat')
% tensor_scalar = 0.844;
% x = x - thalamus_center(1);
% y = y - thalamus_center(2);
% z = z - thalamus_center(3);
% S11 = S11 * tensor_scalar;
% S12 = S12 * tensor_scalar;
% S13 = S13 * tensor_scalar;
% S22 = S22 * tensor_scalar;
% S23 = S23 * tensor_scalar;
% S33 = S33 * tensor_scalar;
%
% fileID = fopen('L3P4ScaledOut.txt', 'wt');
% fprintf(fileID, '%x y z S11 S12 S13 S22 S23 S33');
% format_string = "%7.6f %7.6f %7.6f %7.6f %7.6f %7.6f %7.6f %7.6f %7.6f\n";
% fprintf(fileID, format_string, x, y, z, S11, S12, S13, S22, S23, S33);

% Read the voltage data from COMSOL simulation with anisotropic params
Part4Data = importdata('part4export.txt');

% Interpolate the data
interpolated = scatteredInterpolant(Part4Data(:,1), Part4Data(:,2), Part4Data(:,3), Part4Data(:,4));
anisotropic_x_coordinates = [transpose(linspace(-0.02, 0.02, 1000)), zeros(1000, 1), zeros(1000, 1)];
anisotropic_x_voltages = interpolated(anisotropic_x_coordinates);

figure(4)
plot(linspace(-0.02, 0.02, 1000), anisotropic_x_voltages);

% Interpolate data and calculate second spatial derivative like above
for i = 2:size(x_voltages) - 1
    anisotropic_x_voltages_second_deriv(i) = (anisotropic_x_voltages(i+1) - 2*anisotropic_x_voltages(i) + anisotropic_x_voltages(i-1)) / delta_z^2;
end

figure(5)
plot(linspace(-0.02, 0.02, 999), anisotropic_x_voltages_second_deriv);
```

MATLAB Code Utilized for Analysis of Data in Lab 3

```
% BIOMEDE 517 - Neural Engineering
% Lab 4
% Kushal Jaligama

% Part 1
% Load currents_<big/small>.mat
importfile('currents_big.mat');

% Calculate voltage at a point 50um perpendicularly away from neuron.
% This is based on the current traveling through each compartment. This
% external voltage varies over time, as action potential propogates.
% The external voltage is the sum of the v measurements of every
% compartment, at each time frame.

% Time Axis
times_axis = linspace(0, 18.95, 758);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 50 0];
voltageSuperposition50 = calcVext(currents,XYZ,electrodeXYZ);
hold on
plot(times_axis, voltageSuperposition50);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 100 0];
voltageSuperposition100 = calcVext(currents,XYZ,electrodeXYZ);
plot(times_axis, voltageSuperposition100);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 200 0];
voltageSuperposition200 = calcVext(currents,XYZ,electrodeXYZ);
plot(times_axis, voltageSuperposition200);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 300 0];
voltageSuperposition300 = calcVext(currents,XYZ,electrodeXYZ);
plot(times_axis, voltageSuperposition300);
hold off

% Repeat for smaller neuron

% Load currents_<big/small>.mat
importfile('currents_small.mat');

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 50 0];
voltageSuperposition50 = calcVext(currents,XYZ,electrodeXYZ);
figure(2);
hold on
plot(times_axis, voltageSuperposition50);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 100 0];
voltageSuperposition100 = calcVext(currents,XYZ,electrodeXYZ);
plot(times_axis, voltageSuperposition100);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 200 0];
voltageSuperposition200 = calcVext(currents,XYZ,electrodeXYZ);
plot(times_axis, voltageSuperposition200);

% currents mat object is organized as time on X axis, compartment on Y axis
electrodeXYZ = [0 300 0];
voltageSuperposition300 = calcVext(currents,XYZ,electrodeXYZ);
```

```
plot(times_axis, voltageSuperposition300);
hold off


% Part 2 - Fire 10 neurons for 1s between 2-50Hz, dt = 0.025ms
% Put the ten neurons at intervals along this variation
% inside the 100um cube
variation_coordinates = [zeros(10,1), transpose(linspace(-40, 40, 10)), zeros(10,1)];
electrodeXYZ = [0 0 50];
new_coordinates = zeros(5310, 3);
for i=1:10
  for j=1:531
    % Place a neuron's compartment at new coordinate
    new_coordinates(j + 531 * (i - 1),:) = XYZ(j,:) + variation_coordinates(i,:);
  end
end

% Generate a current at 40Hz for 1s, this lines up with dt = 0.025
% This is good because the currents_<big/small> objects are at dt = 0.025s
% Do this by creating an array of neuron currents for 40,000 timesteps at 0.025 dt
new_currents = zeros(5310, 40000);
% for i=1:10
%     offset = (i-1) * 1000; % Time steps * 0.025ms = 5ms offset for each neuron
%     for t=1+offset:758:40000
%         % Put the currents for each neuron, separated by some offset so
%         % that no neuron has an action potential at the same time
%         new_currents(531 * i - 530 : i*531, t:t+757) = currents;
%     end
% end

t = 1; % ms
while t<40000
  % Randomly pick which neuron fires next
  next_neuron = randi(10);
  % Check that the neuron is not firing
  while (new_currents(next_neuron*531,t) ~= 0)
    next_neuron = randi(10);
  end
  fire_this = next_neuron;
  new_currents((fire_this*531-530):(fire_this*531),t:t+757) = currents;
  % Randomly decide how long to wait before the neuron fires again
  delay = randi(20*40);
  t = t + delay;
end

new_currents = new_currents(:, 1:40000);

figure(5)
plot(linspace(1,20, 40000), new_currents);

% Account for the dead zone of 50um around the electrode
new_coordinate_distances = sqrt((new_coordinates(:,1)-50).^2 + (new_coordinates(:,1)-50).^2 + (new_coordinates(:,1)-50).^2);
% Store indices which of the coordinates are further than 50 um away
alive_zone = find(new_coordinate_distances>=50);

% Calculate the voltage at the electrode for compartments only outside
% deadzone
electrode_voltage_with_dead_zone = calcVext(new_currents(alive_zone,:), new_coordinates(alive_zone,:), (100)*[.5 .5 .5]);
% Calculate the voltage at the electrode for all compartments
electrode_voltage = calcVext(new_currents, new_coordinates, (100)*[.5 .5 .5]);

% Limit the bounds for plotting
electrode_voltage = electrode_voltage(1, 1:40000);
electrode_voltage_with_dead_zone = electrode_voltage_with_dead_zone(1, 1:40000);

% Generate pink noise
```

```
cn = dsp.ColoredNoise('Color','pink','SamplesPerFrame',40000);
recordedNoise = cn();
recordedNoise = recordedNoise / max(abs(recordedNoise));
recordedNoise = recordedNoise * 5e-6; % scale to 10V p2p noise

% Plot the data for electrode voltage without deadzone
figure(3)
subplot(1,2,1);
hold on;
time_axis_part2 = linspace(0, 1, 40000); % Increment in 0.025 ms
plot(time_axis_part2,electrode_voltage);
% Add the pink noise to plot
plot(time_axis_part2,recordedNoise);
axis([0 1 min(electrode_voltage) max(electrode_voltage)]);

% Plot the data for electrode voltage with deadzone
subplot(1,2,2);
hold on;
plot(time_axis_part2,electrode_voltage_with_dead_zone);
% Add the pink noise in to the plot
plot(time_axis_part2,recordedNoise);
axis([0 1 min(electrode_voltage) max(electrode_voltage)]); % consistent axes


% Part 3
L3P2D = importdata('lab3part2export.txt');
% Place a cell with hillock 50 um from origin using currents mat object
p3_neuron_offset = [0 50 0]; % Y direction
p3_interpolant = scatteredInterpolant(L3P2D(:,1), L3P2D(:,2), L3P2D(:,3), L3P2D(:,4));
% Grab the voltage at the coordinates of the compartments (K matrix)
K_matrix = p3_interpolant(XYZ + p3_neuron_offset);
% V_ext(x,y,z) = I_0(x,y,z) * K(x,y,z)
p3_Vext = currents .* K_matrix;
% Get the superposition of voltage by summing all the voltages
for i= 1:758,
    superposition(1,i) = sum(p3_Vext(:,i));
end

figure(4)
plot(linspace(1, 20, 758), superposition);

% Part 4 - Model Neuron Stimulation

% 1 - The current needed to fire the NEURON model is 4 nA
threshold_current = 4 * 10e-3; % Get 4 nA in milliamps

% 2 - Calculate the external voltage seen by every compartment given an
% electrode that is 1 mm away from middle of axon injecting 1 mA
% Create an axon of 5 um with 1000 compartments and scale 2.5 mm to um
axon_x_coords = [transpose(linspace(-2.5 * 10e3, 2.5 * 10e3, 100)), zeros(100, 1), zeros(100,1)];
% Assume the electrode tip is 1mm away from the middle of the axon
electrode_dist = 1 * 10e3; % Convert 1mm to um
electrodeXYZ = [0 electrode_dist 0];
% Give the electrode a current of 1 mA
elec_current = 1; % mA
% Get a new K-Matrix
K_matrix_2 = p3_interpolant(axon_x_coords + electrodeXYZ) ./ 1000; % Scale from 1 A to 1 mA
% Use the K-Matrix to calculate Vext(x,y,z) for each compartment
compartments_ext_v = threshold_current .* K_matrix_2; % This will be along electrode axis
% compartments_ext_v = calcVext(threshold_current, axon_x_coords, electrodeXYZ);

figure(6)
plot(compartments_ext_v);

% Calculate equivalent intracellular current that arises from injected
% extracellular current using Warman equation
% I_int(n) = g_a * (V_e(n-1) + 2V_e(n) + V_e(n+1))
for i=1:length(compartments_ext_v)
  if i == 1
     I_int(i) = compartments_ext_v(i) + 2*compartments_ext_v(i+1) + compartments_ext_v(i+2);
  elseif i == length(compartments_ext_v)
     I_int(i) = compartments_ext_v(i-2) + 2*compartments_ext_v(i-1) + compartments_ext_v(i);
  else
     I_int(i) = compartments_ext_v(i-1) + 2*compartments_ext_v(i) + compartments_ext_v(i+1);
  end
end

minimal_current = max(compartments_ext_v) / threshold_current

g_a = 3*10^-5; %[S]
injection_current = g_a * I_int;

function voltageTrace = calcVext(currentTracesOverTime, currentXYZ, electrodeXYZ)
   % Cindy's Favorite Equation
   % V = I_0 / (4*pi*r*small_sigma)
   % Scale the input distances to meters
   currentXYZ = currentXYZ * 1e-6;
   electrodeXYZ = electrodeXYZ * 1e-6;

   r = sqrt((currentXYZ(:,1)-electrodeXYZ(1)).^2 + (currentXYZ(:,2)-electrodeXYZ(2)).^2 + (currentXYZ(:,3)-electrodeXYZ(3)).^2);
   small_sigma = 0.3333; % S/m => Conductivity

   size(r)
   size(currentTracesOverTime)

   % Perform Cindy's Equation for all of the currents for all compartments
   voltageTraces(:,:) = currentTracesOverTime(:,:) ./ (4 .* pi .* r(:,:) .* small_sigma);
   % Results in voltage traces at a distance r away from the compartment

   voltageTrace(1,:) = sum(voltageTraces(:,:));
end
```