Labs 1 & 2 Report

Introduction

Neuron simulations are becoming an increasingly useful tool in neural engineering to observe the effects of different stimulus and axon parameters on the activation of neurons. Simulations provide a cost effective, time effective, controlled environment to run tests on and are based on real characteristics such as membrane potentials and ion channels. At first in this paper a neuron model based on Hodgkin Huxley dynamics was created. The initial conditions were calculated for Vm and gating variables which then were continually updated to get an estimation of what a neuron model looks like. Also the software NEURON was implemented in Python to more accurately replicate a neuron. The effects of diameter size on action potential propagation speed and factors affecting current density were observed.

Methods

The first step was to calculate all the initial conditions that were going to be used in this model. Initially the Nernst potential for K and Na were calculated using the Nernst equation (Equation 1). Some values that were inputted in this equation were found in (Table 1). Then the GHK equation (Equation 2) was used to find the resting potential of the cell. Next the initial gating variables were calculated using the equations and parameters from the Mainen et al. 1995 paper (Equations 3,4, & 5).

Using these initial conditions a loop was utilized to update the membrane voltage and m,n, and h gating variables. Equations 6,7 8, and 9 were then used to update the membrane voltage and gating variables. With a timestep of 0.01ms these variables were updated for 10ms and graphed with the voltage membrane and the gating variables.

The critical threshold was found by changing the stimulus current and finding the smallest amplitude that gave a spike. The space constant was then calculated using equation 10. Using equation 11, the distance for how far the action potential propagated was determined.

In further experiments NEURON was utilized. NEURON breaks a neuron into cylinders that can be further broken up into segments and given ion channels (usually Hodgkin Huxley).

The first test done in NEURON looked at the effects of axon diameter on action potential speed. Two axons were created both with lengths of 5mm, Hodgkin Huxley ion channels, 1000 segments, but axon 1 has a diameter of 10 um and axon 2 has a diameter of 20 um. Stimulations of 100nA, duration of 0.1ms, and delay of 1ms were given to both axons and the voltages were recorded at positions .1 and .9. These voltage recordings were plotted over 40 ms.

Then a simplified neuron model with sections including a dendrite, soma, axon hillock, non myelinated axon, and myelinated axon were connected in that order. The values of the parameters of that model are given in table 2. A stimulus was given at the dendrite and the voltage in the soma along with the membrane current in the dendrite, soma, myelinated axon, and non-myelinated axon were all plotted. The density of the ion channels in the non-myelinated axon was increased by a factor of 10 and the effects on the negative peak current were observed. Then all of the lengths and diameters were multiplied by factors of 0.5 and 2, the amplitude of the stimulus was adjusted just enough to get an action potential, and the effect on the negative peaks were observed.

Results

In the first step the values of the initial conditions were calculated and are shown in table 3. The equilibrium potential of an ion is the potential required to maintain the concentration gradient so that the concentration force and electric forces are equally balanced and there is no net movement of the ion. K has more positive ions intracellular than extracellular as shown in table 1. The concentration gradient is directed out of the cell, so the Nernst potential is negative to balance the forces. Na has a higher concentration of positive ions outside the cell. The concentration gradient is directed inside the cell, which means that the Nernst potential is positive. Ca has a larger negative extracellular concentration outside the cell than in the cell. Ca has a concentration gradient directed inside the cell, however it also has negative ions which means the Nernst potential is negative. All of the Nernst potentials agree with this.

Membrane voltage is a combination of the K, Na, and Ca Nernst potentials as shown in the GHK equation with K and Ca having the greatest effects. This explains why the membrane potential is a negative value towards the K and Ca Nernst potentials.

The Hodgkin- Huxley channel model describes three types of channels with gating variables m,n and h that model the probability that a channel is open at a given moment in time. The combination of m and h controls Na channels while n controls the K gates.The activation of the K channel is described by n. M describes the activation of the Na channel and h describes the inactivation of the channel. Initially the K channel is closed, given the small value of n. Additionally, the Na channel is closed by the large value for inactivation, h, and the small value for activation, m.

A stimulus of 1250 nA was applied to the model and an action potential is seen in graph 1. The voltage starts at the membrane potential of -72.5 mV, the stimulus is applied and the voltage spikes to about 60 mV, dips a little below the original value and returns up to the initial membrane voltage. Graph 2 demonstrates the effect of the stimulus on the gating variables. The stimulus causes the voltage gated ion channels for Na to open. As the membrane voltage of the model first spikes the activating gating variable for Na, m, rises and the inactivation variable for Na, h, drops. This represents the influx of Na into the cell due to the concentration and voltage gradient which cause the membrane voltage to rise. The high voltage also causes the inactivation of Na channels which is why the inactivation variable h begins to rise and the activation variable drops after the action potential begins. The K channels are more delayed than the Na channels but the high voltage in the cell causes the K channels to open which is why the n gating variable increases after the voltage begins to spike. The high membrane voltage and concentration gradient cause the K ions to go outside of the cell, causing the voltage to repolarize and drop back down a little below the initial value. The membrane voltage hyperpolarizes towards the K nernst potential slightly but then recovers back to the initial membrane potential.

The critical threshold voltage was determined to be 750 nA and gave a peak voltage of 60.23 mV and the action potential started at about -57 mV. Graph 3 demonstrates the critical threshold voltage. This peak voltage shows the model going towards the Na nernst potential which makes sense. Equation 10 was used to calculate a length constant of 0.173. Equation 11 was used to determine that myelin can be extended 0.3712 cm and still enable an action potential. -57 was used for V(x) and 60 was used for Vo in equation 12 but both voltages were subtracted by the membrane voltage of -72.5 to account for the negative values.

For the rest of the tests done NEURON software was implemented in Python. Differences in axon diameter were tested to see the effects on action potential speed. Two axons were created according to the methods. Equation 10 was used to calculate the length constant. Using a Rm of 40,000 Ω *cm^2 and Ra of 200 Ω * cm the length constant was determined to be 0.447 for axon 1 and 0.632 for axon 2. Since axon 2 has a larger length constant it is predicted that the current will travel faster in this axon than axon 2. This is because both axons have the same r_m value,membrane resistance per area, but due to axon 2 having a greater area, this means it has a smaller internal resistance allowing the signal to propagate easier. This was then tested with the NEURON model. A 100 nA stimulus for .1 ms was put at the beginning of both axons and voltages recorded at a section .1 of the length was recorded and voltages were recorded at .9 of the length for both axons. Graphs 4 and 5 show these plots with one for each axon containing both voltages plotted. The plots show that the action potential travels faster in axon 2 than axon 1. It takes longer for axon 1 's voltage at .9 the length to record the peak voltage than it does for axon 2. This can be seen by the distance between the peaks of the graphs. There is a time difference of 2.175ms between the peaks in axon 1 and a time difference of 1.4 ms in axon 2.

A simplified neuron model was then created consisting of a dendrite, soma, axon hillock, non-myelinated axon, and myelinated axon. A stimulus was given and the current densities of each segment and the voltage in the soma was recorded. Graph 6 shows the voltage and graph 7 shows the current densities. The initial spike in current densities are artifacts of the stimulus but there is a downward spike of all the current densities (except the myelinated current with no hodgkin huxley channels and low capacitance) due to the K currents moving inward. The non-myelinated axon has the most negative current. Comparing the soma current density we see that the soma has a minimum value of -0.0149 mA/cm^2 and the non-myelinated axon has a minimum value of -0.13 mA/cm^2 resulting in a difference of -0.115 mA/cm^2. The non-myelinated axon has the largest current density because the soma is a much bigger compartment that is more difficult to charge up while the non-myelinated axon has smaller compartments with a higher density.

The density of ion channels in the non-myelinated axon was increased by a factor of 10 to see the effects. Graph 8 shows that it leads to a larger peak voltage value and Graph 9 showed that it leads to a more negative peak current density for the non-myelinated axon and more positive for the soma. Specifically, the new negative peak for the soma is -0.00672 mA/cm^2 and the new negative peak for the non-myelinated axon is -2.93 mA/cm^2 leading to a difference of -2.93 mA/cm^2. This makes sense because more ion channels in the non-myelinated axon means that more ions can go into the compartments which would lead to a bigger potential and current.

Finally the effect of changing the diameter and length of the neuron on the current density was analyzed. The length and diameter of all compartments were multiplied by a factor .5. Graphs 10 and 11 show the voltage in the soma and the membrane current densities respectively of the neuron with the new lengths. The peak negative current density of the soma was -0.018 mA/cm^2 and for the non-myelinated axon -0.0907 making the distance between the current densities -0.0719 mA/cm^2, smaller than before. The same was done but multiplying the lengths and diameters by a factor of 2. Graph 12 and 13 show the soma voltage and membrane current densities respectively. The current density of the soma is -0.0107mA/ cm^2 and for the non-myelinated axon -0.175 mA/ cm^2 making the difference -0.164 mA/cm^2. Additionally, the currents in mA were calculated and shown in graphs 14, 15 and 16. It shows that as the length and diameter of the neuron grows, the current values also get bigger. This is because there is a larger area which means there can be more current and the current density also is growing with increasing diameter and size. A larger current means that it is easier to detect the action potential. So a larger neuron being recorded at the non-myelinated axon would be the best to detect an action potential because this is where the current is largest.

Discussion

This report focused on neuron models and how they can be used to analyze different characteristics of neurons and how that affects parameters such as action potential and membrane currents. The first model was based off of Hodgkin Huxley dynamics and began with calculating initial values for nernst potentials, membrane voltage and gating variables. These values were updated over time and an action potential was shown with the membrane having a spike and gating variables rising or dropping depending on the part of the action potential. Additionally a minimum stimulation current of 750 nA was found and that the myelin can extend 0.3712 cm and still produce an action potential

Models using NEURON were also created. A simulation validated that a longer length constant correlated with a faster signal propagation. A simple model containing neuron components was created and analyzed the current densities. The unmyelinated axon always contained the most negative current density indicating that it is the best position to record signals. Additionally, the number of ion channels in the non-myelinated axon was increased by a factor of 10 and increased the negative peak of the current density. The diameter and length of the axon were multiplied by a factor of .5 and 2 and it seemed the larger the neuron the larger the current was recorded. This indicates that larger neurons are easier to record signals from.

Neuron models are an extremely helpful tool in doing research. They let scientists perform many experiments on neurons much easier than doing tests with real neurons. Scientists can use these models to analyze resistance effect on action potential, or model a cell they are using for real experiments to see if they get a membrane current large enough to be detectable, and many more experiments modifying variables. ModelDB is a great database with many neuron models. There is a human motor neuron model (Gaine et al 2018). This could be used for an experiment with different electrodes that stimulate neurons on a patient's arm who is no longer able to move it. It can be used to see what type of electrode would stimulate the neurons the best. Overall, neurons are a great resource for investigating neurons.

Appendix

Equations

$E_{ion} = \frac{k_B T}{ze} ln \frac{Co}{Ci} = \frac{25}{z} ln \frac{Co}{Ci}\ (mV)$ (eq) (Equation. 1)

$Vm\ =\ 25\ ln \frac{P_k[K_0]+P_{Na}[Na_o]+P_{Cl}[Cl_i]}{P_k[K_i]+P_{Na}[Na_i]+P_{Cl}[Cl_0]}$ (Equation 2)

$P_K = 1.0$ $P_{Na}\ =\ 0.04$ $P_{Cl}\ =\ 0.45$

$m_0\ =\ m_\infty = \frac{\alpha_m(V_{rest})}{\alpha_m(V_{rest})+\beta_m(V_{rest})}$ (equation 3)

$\alpha(V)\ =\ \frac{A(V_m - V_{1/2})}{1-e^{-(V_m-V_{1/2})/k}}$ (equation 4)

$\beta(V)\ =\ \frac{-A(V_m-V_{1/2})}{1-e^{(V_m-V_{1/2})/k}}$(equation 5)

$\Delta V_m = \Delta t(I_M\ -\ \overline{g}_{Na} m^3(1\ -\ h)(Vm\ -\ ENa)\ +\ \overline{g}_k n^4(Vm - E_k)\ +\ g_{Cl}(V_m - E_{Cl}))$
(Equation 6)

$\Delta m\ =\ \Delta t(\alpha_n(1\ -\ n)\ -\ \beta_n n)$ (Equation 7)

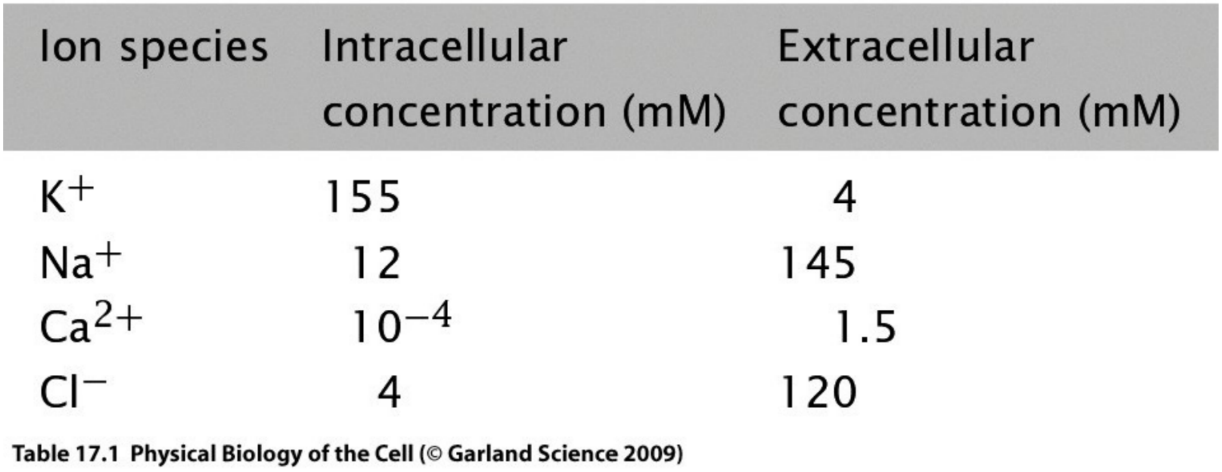
$V_m(t\ +\ 1)\ =\ V_m(t)\ +\ \Delta V_m$(Equation 8)

$m(t\ +\ 1)\ =\ m(t)\ +\ \Delta m$ (Equation 9)

$\lambda\ =\ \sqrt{\frac{aR_m}{2R_i}}$ (Equation 10)

$V(x)\ =\ V_o e^{\frac{-x}{\lambda}}$ (eaution 11)

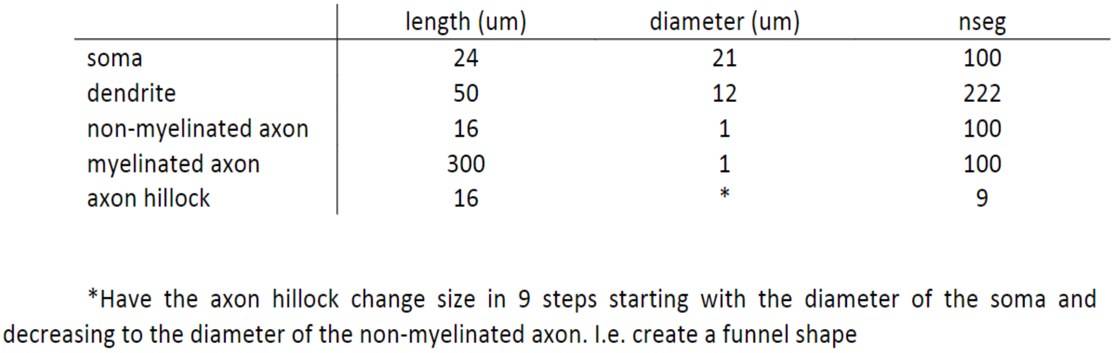
Table
(Table 1)



Ion species	Intracellular concentration (mM)	Extracellular concentration (mM)
K^+	155	4
Na^+	12	145
Ca^{2+}	10^{-4}	1.5
Cl^-	4	120

Table 17.1 Physical Biology of the Cell (© Garland Science 2009)

(Table 2)



	length (um)	diameter (um)	nseg
soma	24	21	100
dendrite	50	12	222
non-myelinated axon	16	1	100
myelinated axon	300	1	100
axon hillock	16	*	9

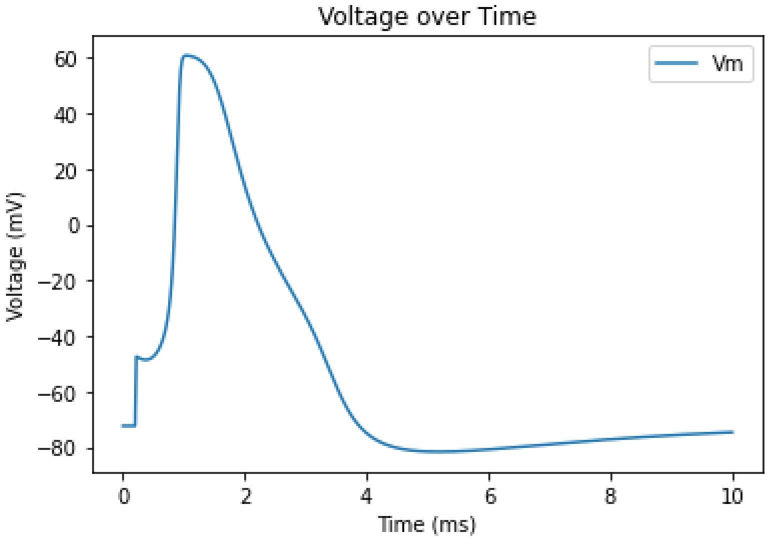
*Have the axon hillock change size in 9 steps starting with the diameter of the soma and decreasing to the diameter of the non-myelinated axon. I.e. create a funnel shape

(Table 3)

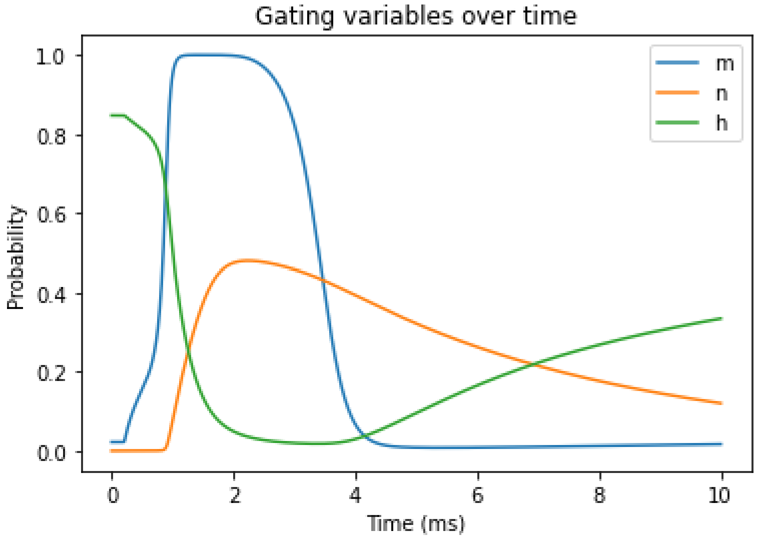
<u>Resting State Variables</u>	<u>Value</u>
K Nernst Potential	-91.43 mV
Na Nernst Potential	62.3 mV
Cl Nernst Potential	-85.03mV
Vm	-72.34 mV
m	0.023

n	0.000350
h	0.847

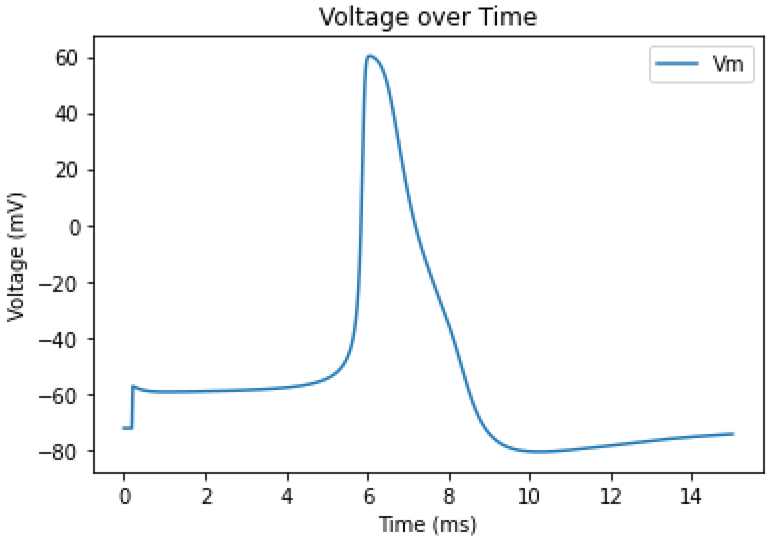
Graphs
(Graph 1)



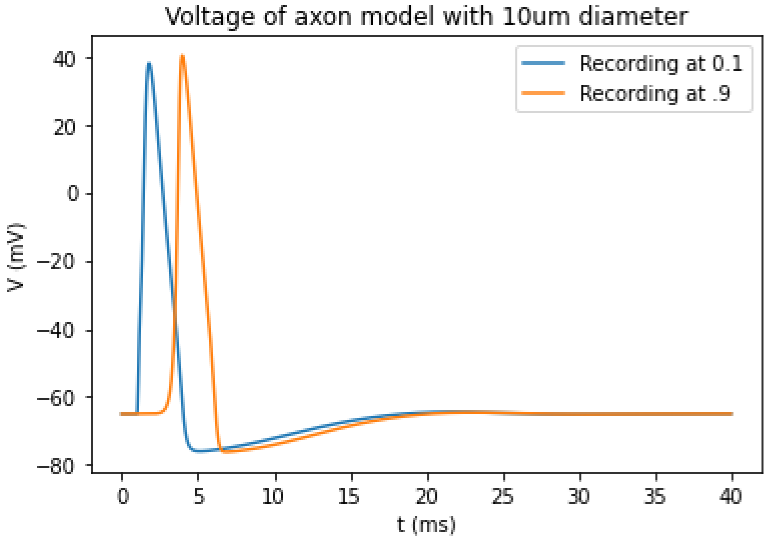
(Graph 2)



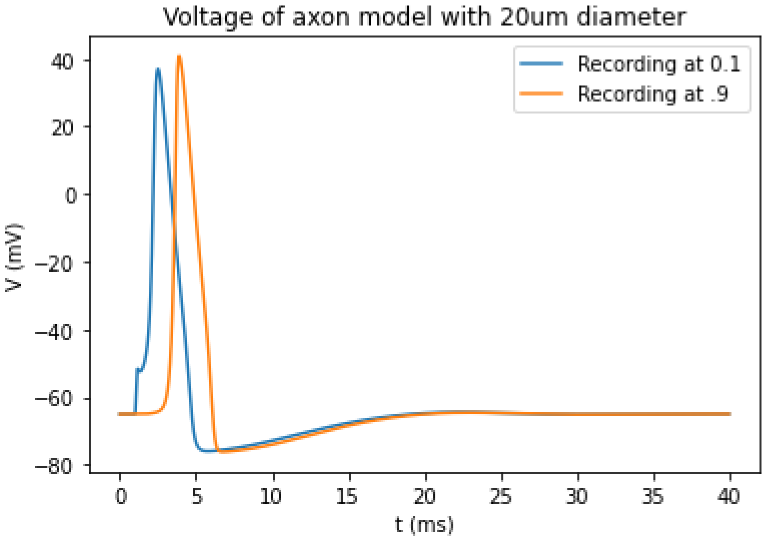
(Graph 3)



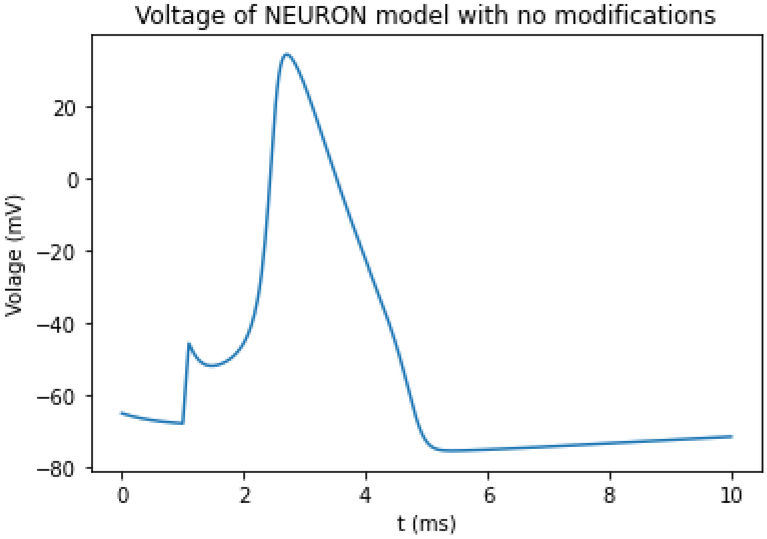
(Graph 4)



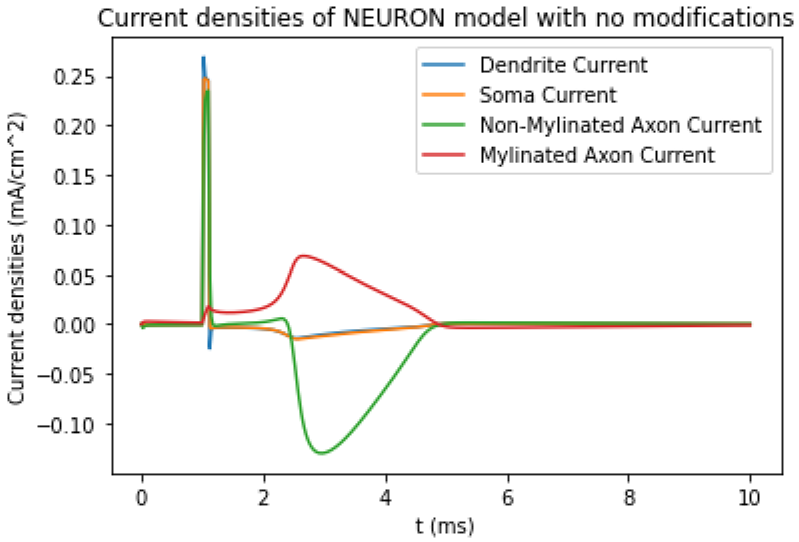
(Graph 5)



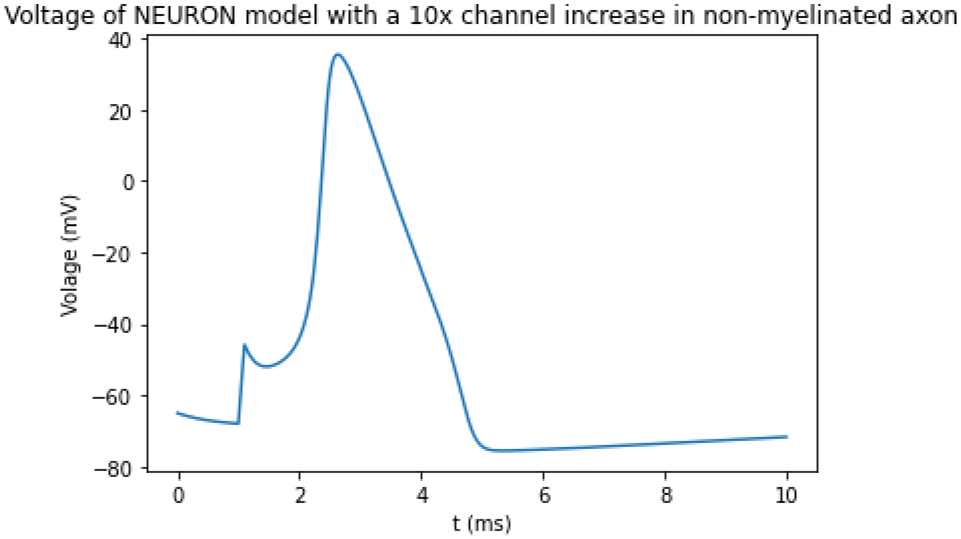
(Graph 6)



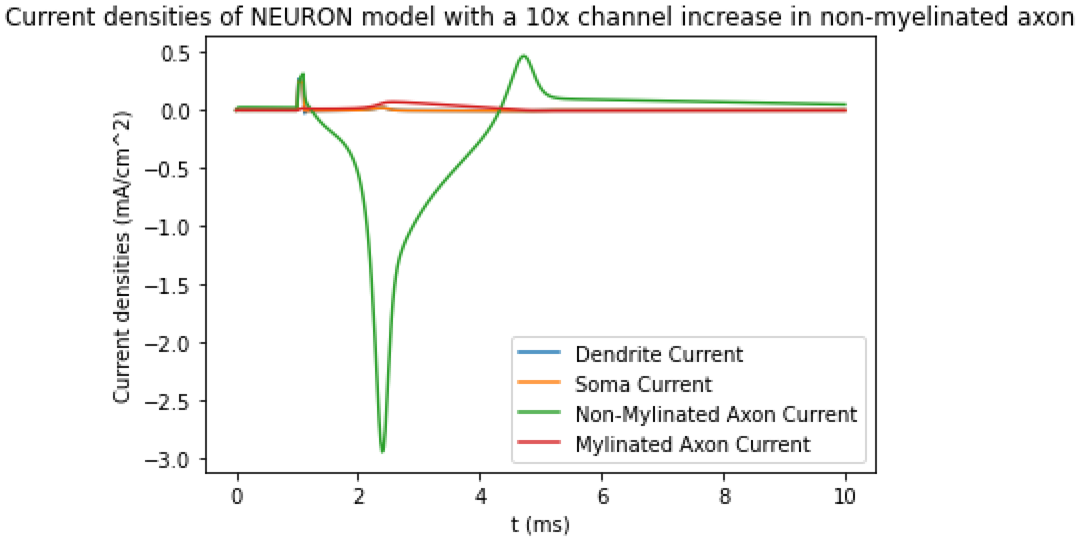
(Graph 7)



(Graph 8)

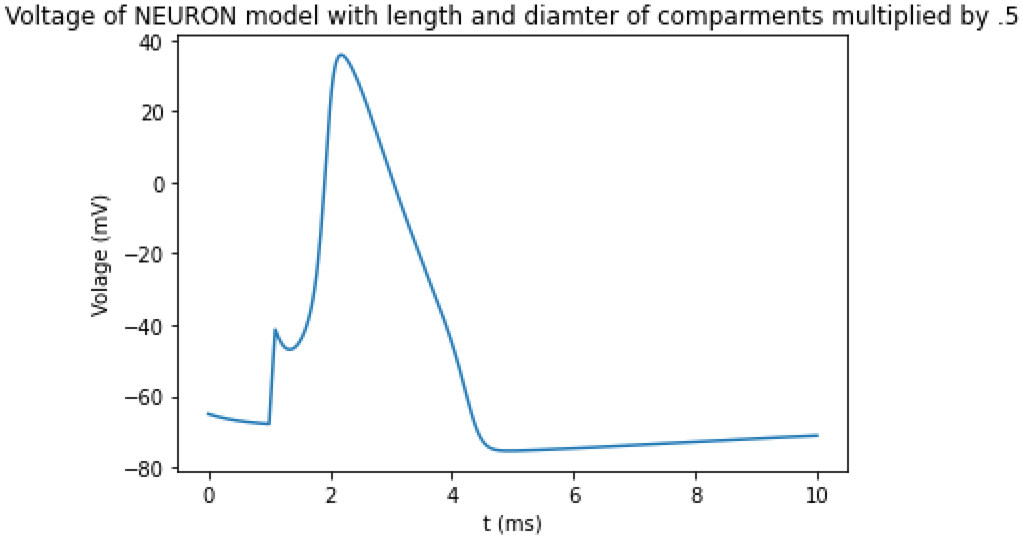


(Graph 9)



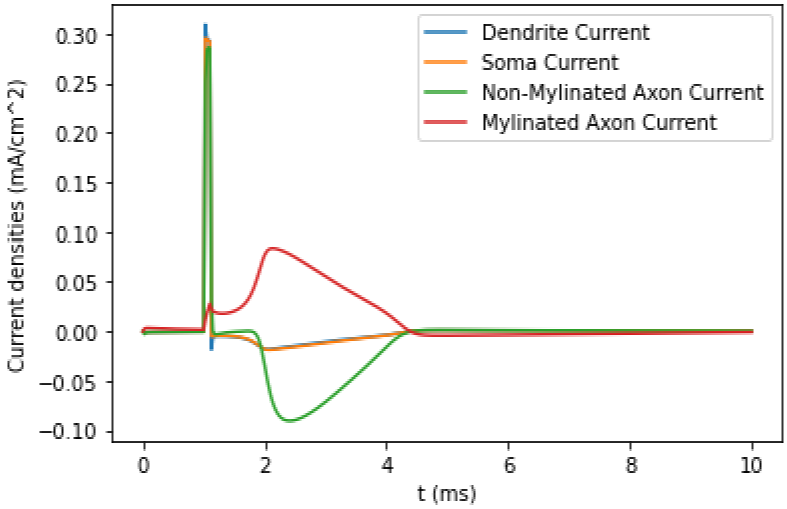
Current densities of NEURON model with a 10x channel increase in non-myelinated axon

(Graph 10)

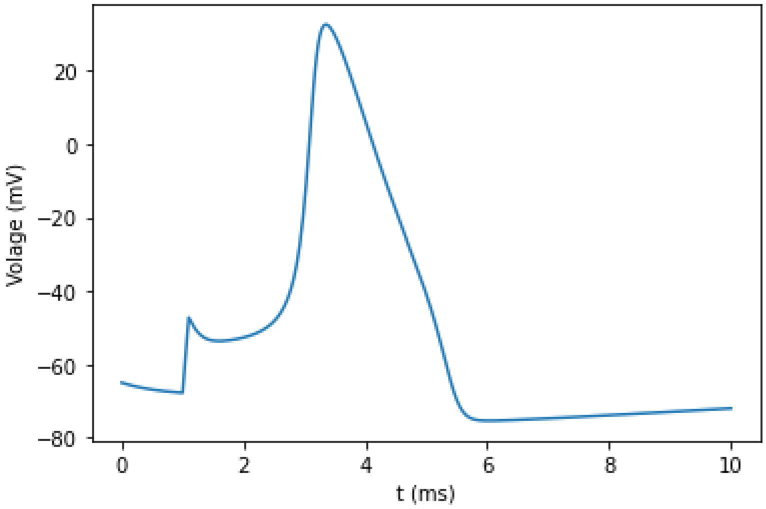


Voltage of NEURON model with length and diamter of comparments multiplied by .5

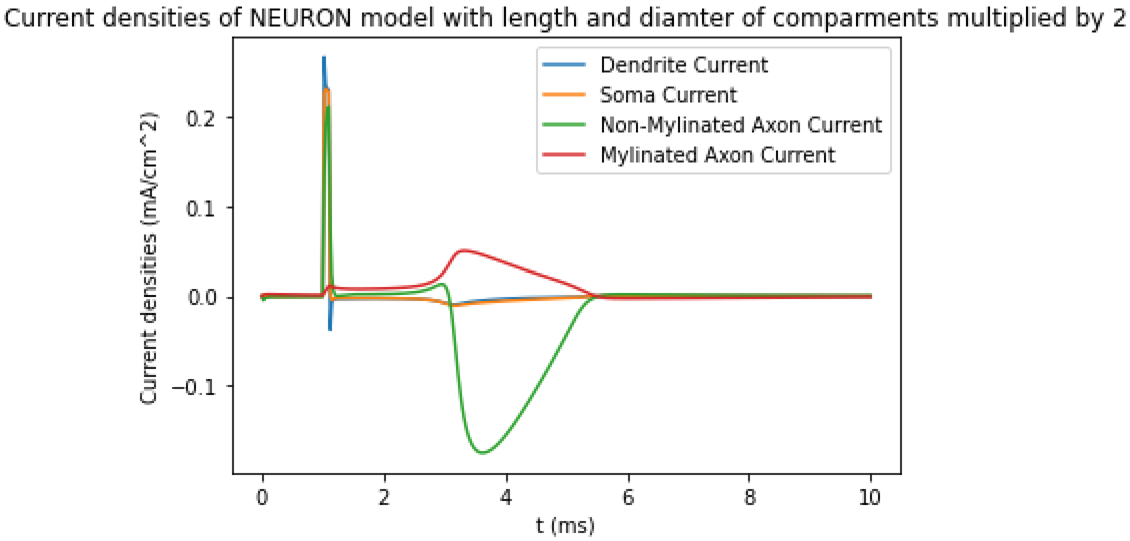
(Graph 11)


Current densities of NEURON model with length and diamter of comparments multiplied by .5

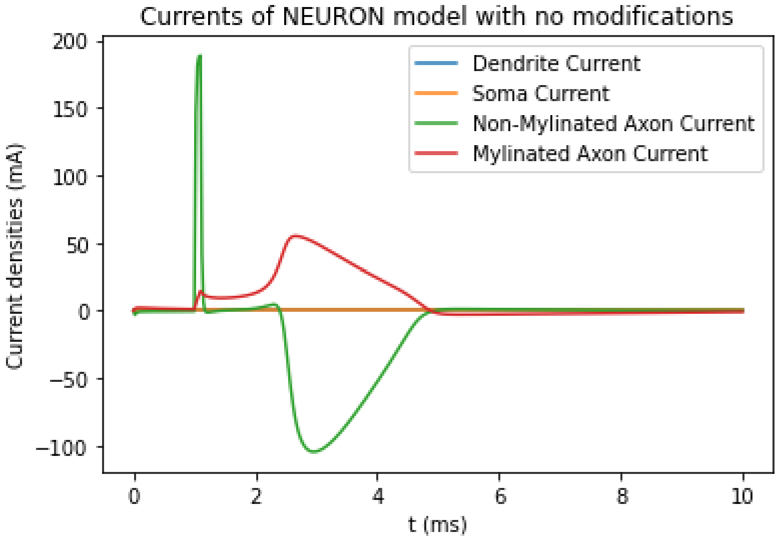
(Graph 12)


Voltage of NEURON model with length and diamter of comparments multiplied by 2

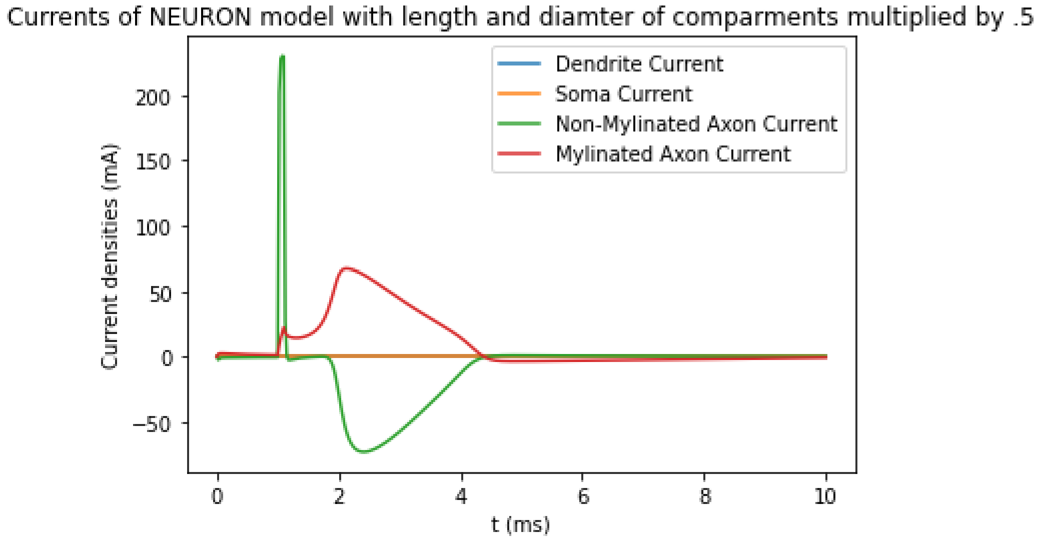
(Graph 13)



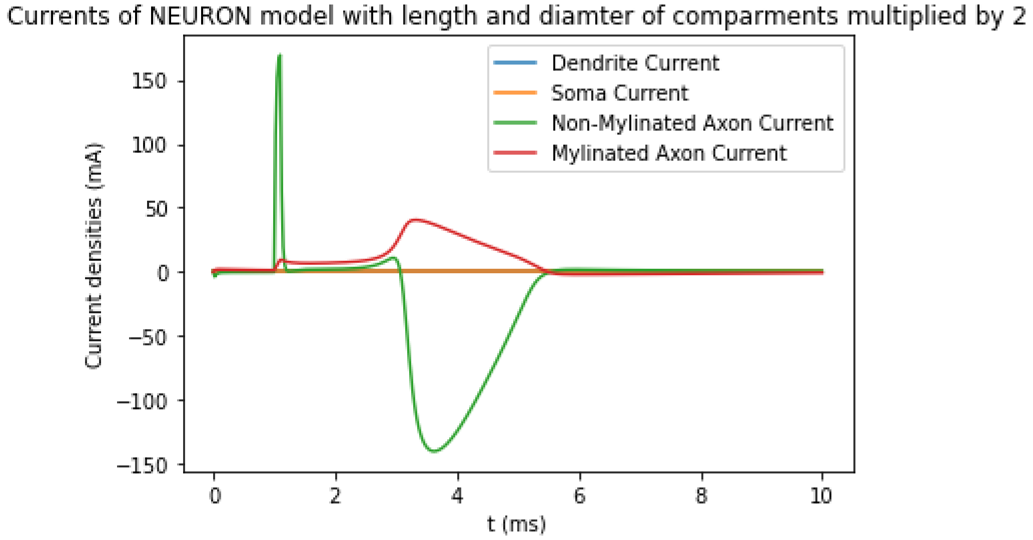
(Graph 14)



(Graph 15)



(Graph 16)



Lab 1 Code

```
import numpy as np
import math
import matplotlib.pyplot as plt
Ek = (25/1) * math.log(4/155)
print("Potassium Nernst Potential:", Ek)
ENa = (25/1) * math.log(145/12)
print("Sodium Nernst Potential:", ENa)
ECa = (25/2) * math.log(1.5/(10**-4))
print("Calcium Nernst Potential:", ECa)
ECl = (25/-1) * math.log(120/4)
print("Chloride Nernst Potential:", ECl)
Cm = 1**10-6           # uF/cm^2
gNa = 100**-3          # mS/cm^2
# Question 2 GHK Equation
Vm = 25 * math.log((1.0*4 + 0.04*145 + 0.45*4)/(1.0*155 + 0.04*12 + 0.45*120))
print(Vm)
alpham = (0.182 * (Vm - (-35)))/ (1 - math.exp((-(Vm - (-35)))/9) )
betam = (-0.124 * (Vm - (-35)))/ (1 - math.exp(((Vm - (-35)))/9) )
m = (alpham)/(alpham + betam)

alphah = (0.024 * (Vm - (-50)))/ (1 - math.exp((-(Vm - (-50)))/5) )
betah = (-0.0091 * (Vm - (-75)))/ (1 - math.exp(((Vm - (-75)))/5) )

h = (alphah)/(alphah + betah)

alphan = (0.02 * (Vm - (20)))/ (1 - math.exp((-(Vm - (20)))/9) )
betan = (-0.002 * (Vm - (20)))/ (1 - math.exp(((Vm - (20)))/9) )

n = (alphan)/(alphan + betan)

print("m value:",m)
print("h value:", (1-h))
print("n value", n)
```

```
dt = 0.01 # ms
VmList = []
VmList.append(Vm)
mList = []
mList.append(m)
nList = []
nList.append(n)
hList = []
hList.append(1-h)

'''
INa = 100 * m**3 * h * (Vm - ENa) # 100 mS/cm^2
Ik =  50 * n**4 * (Vm - Ek)       # 50 mS/cm^2
IL = 0.5*(Vm - (-72.5))
Im = 0
dVm = dt * (Im - INa - Ik - IL)/(1) # uF/cm^2

dm = dt * (alpham * (1 - m) - betam * m)
dn = dt * (alphan * (1 - n) - betan * n)
dh = dt * (alphah * (1 - h) - betah * h)
'''
Im = 0
time = np.arange(0, 10, dt)


for x in time:
  INa = 100 * m**3 * (1-h) * (Vm - ENa) # 100 mS/cm^2
  Ik =  50 * n**4 * (Vm - Ek)       # 50 mS/cm^2
  IL = 0.5*(Vm - (-72.5))
  if x == 0.20:
    Im = 1250 # uF/cm^2 Not right
  elif x == 0.22:
    Im = 0
  dVm = dt * (Im - INa - Ik - IL)/(1) # uF/cm^2 Not right
  Vm = Vm + dVm
  alpham = (0.182 * (Vm - (-35)))/ (1 - math.exp((-(Vm - (-35)))/9) )
  betam = (-0.124 * (Vm - (-35)))/ (1 - math.exp(((Vm - (-35)))/9) )

  alphah = (0.024 * (Vm - (-50)))/ (1 - math.exp((-(Vm - (-50)))/5) )
  betah = (-0.0091 * (Vm - (-75)))/ (1 - math.exp(((Vm - (-75)))/5) )

  alphan = (0.02 * (Vm - (20)))/ (1 - math.exp((-(Vm - (20)))/9) )
  betan = (-0.002 * (Vm - (20)))/ (1 - math.exp(((Vm - (20)))/9) )

  dm = dt * (alpham * (1 - m) - betam * m)
  m = m + dm
  dn = dt * (alphan * (1 - n) - betan * n)
  n = n + dn
  dh = dt * (alphah * (1 - h) - betah * h)
  h = h + dh

  VmList.append(Vm)
  mList.append(m)
  nList.append(n)
  hList.append(1-h)

time = np.arange(0, 10 + dt, dt)
plt.figure(0)
plt.plot(time, VmList)
plt.legend(["Vm"])
plt.title("Voltage over Time")
plt.ylabel("Voltage (mV)")
plt.xlabel("Time (ms)")

plt.figure(1)
plt.plot(time, mList)
plt.plot(time, nList)
plt.plot(time, hList)
plt.legend(['m','n','h'])
plt.title("Gating variables over time")
plt.ylabel("Voltage (mV)")
plt.xlabel("Time (ms)")



print("Peak Voltage:",max(VmList))
print("Voltage: ", -57)
```

2.

```
Rm = 40000 ohm-cm^2
Ra = 200 ohm-cm
d = 3 um
r = 1.5 um

lambda = sqrt((a * Rm)/(2 * Ra))
lambda = sqrt((0.0003 cm * 40000 ohm-cm^2)/(2 * 200 ohm-cm))
lambda = 0.173205

(-57--72.5) = (60--72.5)*e^(-x/0.173).
x = 0.3712 cm
```

Lab 2 Code

```
!pip install neuron

from neuron import h
from neuron.units import ms, mV, um
from scipy.signal import find_peaks
#import matplotlib.pyplot as plt
%matplotlib inline
import matplotlib.pyplot as plt
```

```
#output_notebook()
import textwrap
import csv
import numpy as np
import math
h.load_file('stdrun.hoc')


soma = h.Section(name='soma')
#h.topology()
#soma.psection()
#soma.psection()['morphology']['L']
soma.L = 20
soma.diam = 20
#dir(soma)
#print(textwrap.fill(', '.join(dir(h))))
#help(soma.connect)
#?soma.connect
soma.insert('hh')
#print("type(soma) = {}".format(type(soma)))
#print("type(soma(0.5)) = {}".format(type(soma(0.5))))
mech = soma(0.5).hh
#print(dir(mech))
#print(mech.gkbar)
#print(soma(0.5).hh.gkbar)
iclamp = h.IClamp(soma(0.5))
#print([item for item in dir(iclamp) if not item.startswith('__')])
iclamp.delay = 2
iclamp.dur = 0.1
iclamp.amp = 0.9
#soma.psection()
v = h.Vector().record(soma(0.5)._ref_v)             # Membrane potential vector
t = h.Vector().record(h._ref_t)                     # Time stamp vector
h.finitialize(-65 * mV)
```

```
h.continuerun(40 * ms)

plt.figure()
plt.plot(t, v)
plt.xlabel('t (ms)')
plt.ylabel('v (mV)')
plt.show()

with open('data.csv', 'w') as f:
    csv.writer(f).writerows(zip(t, v))


with open('data.csv') as f:
    reader = csv.reader(f)
    tnew, vnew = zip(*[[float(val) for val in row] for row in reader if row])

plt.figure()
plt.plot(tnew, vnew)
plt.xlabel('t (ms)')
plt.ylabel('v (mV)')
plt.show()

axon1 = h.Section(name = 'axon1')
axon2 = h.Section(name = 'axon2')
axon1.diam = 10 * um
axon2.diam = 20 * um
axon1.L = 5000 * um                 # Hopefully these are in mm
axon2.L = 5000 * um                 # Hopefully these are in mm
axon1.nseg = 1000
axon2.nseg = 1000

axon1.insert("hh")
axon2.insert("hh")
print(axon1.L)
```

```
Rm = 40000   # Ohm * cm^2
Ra = 200     # Ohm * cm
lambda1 = math.sqrt((.001* Rm)/Ra)     # Converted 10 um to .001 cm
lambda2 = math.sqrt((.002* Rm)/Ra)     # Converted 20 um to .002 cm
print("Space constant for axon 1:", lambda1)
print("Space constant for axon 2:", lambda2)




iclamp1 = h.IClamp(axon1(0))
iclamp2 = h.IClamp(axon2(0))

iclamp1.delay = 1
iclamp2.delay = 1
iclamp1.dur = 0.1
iclamp2.dur = 0.1
iclamp1.amp = 100
iclamp2.amp = 100


v1 = h.Vector().record(axon1(.1)._ref_v)
t = h.Vector().record(h._ref_t)
v2 = h.Vector().record(axon1(.9)._ref_v)


v3 = h.Vector().record(axon2(.1)._ref_v)
t2 = h.Vector().record(h._ref_t)
v4 = h.Vector().record(axon2(.9)._ref_v)

h.finitialize(-65 * mV)
h.continuerun(40 * ms)
```

```
#plt.figure()
plt.plot(t, v1)
plt.plot(t, v2)
plt.xlabel('t (ms)')
plt.ylabel('V (mV)')
plt.title("Voltage of axon model with 10um diameter")
plt.legend(['Recording at 0.1', 'Recording at .9'])
plt.show()
print("Max Voltage for V1 recorded at 0.1", max(v1))
print("Max voltage for V2 recorded at 0.9", max(v2))
maxVoltage1 = max(v1)
maxVoltage2 = max(v2)
timeMaxVoltage1 = 0
timeMaxVoltage2 = 0
for x in range(len(v1)):
  if v1[x] == maxVoltage1:
    print("Index of max voltage 1:", x)
    print("Voltage at max voltage 1:", v1[x])
    timeMaxVoltage1 = x
  if v2[x] == maxVoltage2:
    timeMaxVoltage2 = x
    print("Index of max voltage 2:", x)
    print("Voltage at max voltage 2:", v2[x])

print()
print("Time of max voltage 1", t[timeMaxVoltage1])
print("Time of max voltage 2", t[timeMaxVoltage2])
print("Time between max voltages of axon 1:",  t[timeMaxVoltage2] - t[timeMaxVoltage1])
print()




plt.figure()
plt.plot(t2, v3)
plt.plot(t2, v4)
plt.xlabel('t (ms)')
plt.ylabel('V (mV)')
plt.title("Voltage of axon model with 20um diameter")
plt.legend(['Recording at 0.1', 'Recording at .9'])
plt.show()

print("Max Voltage for V3 recorded at 0.1", max(v3))
print("Max voltage for V4 recorded at 0.9", max(v4))
maxVoltage3 = max(v3)
maxVoltage4 = max(v4)
timeMaxVoltage3 = 0
timeMaxVoltage4 = 0
for x in range(len(v3)):
  if v3[x] == maxVoltage3:
    print("Index of max voltage 3:", x)
    print("Voltage at max voltage 3:", v3[x])
    timeMaxVoltage3 = x
  if v4[x] == maxVoltage4:
    timeMaxVoltage4 = x
    print("Index of max voltage 4:", x)
    print("Voltage at max voltage 4:", v4[x])

print()
print("Time of max voltage 3", t[timeMaxVoltage3])
print("Time of max voltage 4", t[timeMaxVoltage4])
print("Time between max voltages of axon 2:",  t[timeMaxVoltage4] - t[timeMaxVoltage3])
print()



soma = h.Section(name = 'soma')
soma.L = 24 * um
soma.diam = 21 * um
#soma.diam = 21 * .5 * um
soma.nseg = 100
soma.insert('hh')
soma.insert('pas')
soma.insert('extracellular')

dendrite = h.Section(name = 'dendrite')
dendrite.L = 50 * um
dendrite.diam = 12 * um
#dendrite.diam = 12 * um *.5
dendrite.nseg = 222
dendrite.insert('hh')
dendrite.insert('pas')
dendrite.insert('extracellular')

nonMylinatedAxon = h.Section(name = 'nonMylinatedAxon')
nonMylinatedAxon.L = 16 * um
nonMylinatedAxon.diam = 1 * um
#nonMylinatedAxon.diam = 1 * um *.5
nonMylinatedAxon.nseg = 100
nonMylinatedAxon.insert('hh')
nonMylinatedAxon.insert('pas')
nonMylinatedAxon.insert('extracellular')


mylinatedAxon = h.Section(name = 'mylinatedAxon')
mylinatedAxon.L = 300 * um
mylinatedAxon.diam = 1 * um
#mylinatedAxon.diam = 1 * um *.5
mylinatedAxon.nseg = 100
mylinatedAxon.insert('pas')
mylinatedAxon.insert('extracellular')
mylinatedAxon.cm = 0.04
```

```
axonHillock = h.Section(name = 'axonHillock')
axonHillock.L = 16 * um
axonHillock.nseg = 9

diams = np.arange(1, 21, 2.22222222)
diams = diams[::-1]
for i in range(0,10):
  axonHillock(0.1 * i).diam = diams[i] * um
  #axonHillock(0.1 * i).diam = diams[i] * um * .5


axonHillock.insert('hh')
axonHillock.insert('pas')
axonHillock.insert('extracellular')

dendrite.connect(soma)
soma.connect(axonHillock)
axonHillock.connect(nonMylinatedAxon)
nonMylinatedAxon.connect(mylinatedAxon)

'''
for seg in nonMylinatedAxon:
  seg.hh.gnabar = seg.hh.gnabar * 10
  seg.hh.gkbar = seg.hh.gkbar * 10
'''

iclamp = h.IClamp(dendrite(0.5))
iclamp.delay = 1
iclamp.dur = 0.1
iclamp.amp = 10
#iclamp.amp = 5
voltage = h.Vector().record(soma(.1)._ref_v)
```

```
curDend = h.Vector().record(dendrite(0.5)._ref_i_membrane)
curSoma = h.Vector().record(soma(0.5)._ref_i_membrane)
curNonMylinated = h.Vector().record(nonMylinatedAxon()._ref_i_membrane)
curMylinated = h.Vector().record(mylinatedAxon(0.5)._ref_i_membrane)

time = h.Vector().record(h._ref_t)
h.finitialize(-65 * mV)
h.continuerun(10 * ms)

plt.plot(time, voltage)
plt.xlabel('t (ms)')
plt.ylabel('Volage (mV)')
#plt.title("Voltage of NEURON model with no modifications")
plt.title("Voltage of NEURON model with no modifications")
plt.show()

plt.figure()
plt.plot(time, curDend)
plt.plot(time, curSoma)
plt.plot(time, curNonMylinated)
plt.plot(time, curMylinated)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon
Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current densities (mA/cm^2)')
#plt.title("Current densities of NEURON model with no modifications")
plt.title("Current densities of NEURON model with no modifications")



plt.figure()
plt.plot(time, curDend * 3.14 * ((21/2 * um)**-4)**2)
plt.plot(time, curSoma * 3.14 * ((12/2 * um)**-4)**2)
plt.plot(time, curNonMylinated * 3.14 * ((1/2 * um)**-4)**2)
plt.plot(time, curMylinated* 3.14 * ((1/2 * um)**-4)**2)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon
Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current (mA)')
#plt.title("Current densities of NEURON model with length and diamter of comparments
multiplied by 2")
plt.title("Currents of NEURON model with no modifications")

print("Minimum non-mylinated value:",min(curNonMylinated))
print("Minimum Soma:",min(curSoma))
print("Difference between soma and non-mylinated value", min(curNonMylinated) -
min(curSoma))

print("Minimum non-mylinated value:",min(curNonMylinated * 3.14 * ((1/2 * um)**-4)**2))
print("Minimum Soma:",min(curSoma * 3.14 * ((12/2 * um)**-4)**2))
print("Difference between soma and non-mylinated value", min(curNonMylinated * 3.14 * ((1/2
* um)**-4)**2) - min(curSoma * 3.14 * ((12/2 * um)**-4)**2))

#Values without changing gnabar and gkbar
#Minimum non-mylinated value: -0.12994448913935763
#Minimum Soma: -0.014862253323731506
#Difference between soma and non-mylinated value -0.11508223581562613

#Values after changing gnabar and gkbar
#Minimum non-mylinated value: -2.9393391529476562
#Minimum Soma: -0.006718267180347251
#Difference between soma and non-mylinated value -2.932620885767309




soma = h.Section(name = 'soma')
soma.L = 24 * um
soma.diam = 21 * um
```

```
#soma.diam = 21 * .5 * um
soma.nseg = 100
soma.insert('hh')
soma.insert('pas')
soma.insert('extracellular')

dendrite = h.Section(name = 'dendrite')
dendrite.L = 50 * um
dendrite.diam = 12 * um
#dendrite.diam = 12 * um *.5
dendrite.nseg = 222
dendrite.insert('hh')
dendrite.insert('pas')
dendrite.insert('extracellular')

nonMylinatedAxon = h.Section(name = 'nonMylinatedAxon')
nonMylinatedAxon.L = 16 * um
nonMylinatedAxon.diam = 1 * um
#nonMylinatedAxon.diam = 1 * um *.5
nonMylinatedAxon.nseg = 100
nonMylinatedAxon.insert('hh')
nonMylinatedAxon.insert('pas')
nonMylinatedAxon.insert('extracellular')


mylinatedAxon = h.Section(name = 'mylinatedAxon')
mylinatedAxon.L = 300 * um
mylinatedAxon.diam = 1 * um
#mylinatedAxon.diam = 1 * um *.5
mylinatedAxon.nseg = 100
mylinatedAxon.insert('pas')
mylinatedAxon.insert('extracellular')
mylinatedAxon.cm = 0.04
```

```
axonHillock = h.Section(name = 'axonHillock')
axonHillock.L = 16 * um
axonHillock.nseg = 9

diams = np.arange(1, 21, 2.22222222)
diams = diams[::-1]
for i in range(0,10):
  axonHillock(0.1 * i).diam = diams[i] * um
  #axonHillock(0.1 * i).diam = diams[i] * um * .5


axonHillock.insert('hh')
axonHillock.insert('pas')
axonHillock.insert('extracellular')

dendrite.connect(soma)
soma.connect(axonHillock)
axonHillock.connect(nonMylinatedAxon)
nonMylinatedAxon.connect(mylinatedAxon)


for seg in nonMylinatedAxon:
  seg.hh.gnabar = seg.hh.gnabar * 10
  seg.hh.gkbar = seg.hh.gkbar * 10


iclamp = h.IClamp(dendrite(0.5))
iclamp.delay = 1
iclamp.dur = 0.1
iclamp.amp = 10
#iclamp.amp = 5
voltage = h.Vector().record(soma(.1)._ref_v)

curDend = h.Vector().record(dendrite(0.5)._ref_i_membrane)
```

```
curSoma = h.Vector().record(soma(0.5)._ref_i_membrane)
curNonMylinated = h.Vector().record(nonMylinatedAxon()._ref_i_membrane)
curMylinated = h.Vector().record(mylinatedAxon(0.5)._ref_i_membrane)

time = h.Vector().record(h._ref_t)
h.finitialize(-65 * mV)
h.continuerun(10 * ms)

plt.plot(time, voltage)
plt.xlabel('t (ms)')
plt.ylabel('Volage (mV)')
#plt.title("Voltage of NEURON model with no modifications")
plt.title("Voltage of NEURON model with a 10x channel increase in non-myelinated axon")
plt.show()

plt.figure()
plt.plot(time, curDend)
plt.plot(time, curSoma)
plt.plot(time, curNonMylinated)
plt.plot(time, curMylinated)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon
Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current densities (mA/cm^2)')
#plt.title("Current densities of NEURON model with no modifications")
plt.title("Current densities of NEURON model with a 10x channel increase in non-myelinated
axon")


print("Minimum non-mylinated value:",min(curNonMylinated))
print("Minimum Soma:",min(curSoma))
print("Difference between soma and non-mylinated value", min(curNonMylinated) -
min(curSoma))
```

```
soma = h.Section(name = 'soma')
soma.L = 24 * um * .5
soma.diam = 21 * um *.5
soma.nseg = 100
soma.insert('hh')
soma.insert('pas')
soma.insert('extracellular')

dendrite = h.Section(name = 'dendrite')
dendrite.L = 50 *  um *.5
dendrite.diam = 12 * um* .5
dendrite.nseg = 222
dendrite.insert('hh')
dendrite.insert('pas')
dendrite.insert('extracellular')

nonMylinatedAxon = h.Section(name = 'nonMylinatedAxon')
nonMylinatedAxon.L = 16 * um *.5
nonMylinatedAxon.diam = 1 * um *.5
nonMylinatedAxon.nseg = 100
nonMylinatedAxon.insert('hh')
nonMylinatedAxon.insert('pas')
nonMylinatedAxon.insert('extracellular')


mylinatedAxon = h.Section(name = 'mylinatedAxon')
mylinatedAxon.L = 300 * um * .5
mylinatedAxon.diam = 1 * um * .5
mylinatedAxon.nseg = 100
mylinatedAxon.insert('pas')
mylinatedAxon.insert('extracellular')
mylinatedAxon.cm = 0.04

axonHillock = h.Section(name = 'axonHillock')
```

```
axonHillock.L = 16 * um * .5
axonHillock.nseg = 9

diams = np.arange(1, 21, 2.22222222)
diams = diams[::-1]
for i in range(0,10):
  axonHillock(0.1 * i).diam = diams[i] * um * .5


axonHillock.insert('hh')
axonHillock.insert('pas')
axonHillock.insert('extracellular')

dendrite.connect(soma)
soma.connect(axonHillock)
axonHillock.connect(nonMylinatedAxon)
nonMylinatedAxon.connect(mylinatedAxon)

'''
for seg in nonMylinatedAxon:
  seg.hh.gnabar = seg.hh.gnabar * 10
  seg.hh.gkbar = seg.hh.gkbar * 10
'''

iclamp = h.IClamp(dendrite(0.5))
iclamp.delay = 1
iclamp.dur = 0.1
#iclamp.amp = 10
iclamp.amp = 3
voltage = h.Vector().record(soma(.1)._ref_v)

curDend = h.Vector().record(dendrite(0.5)._ref_i_membrane)
curSoma = h.Vector().record(soma(0.5)._ref_i_membrane)
curNonMylinated = h.Vector().record(nonMylinatedAxon()._ref_i_membrane)
```

```
curMylinated = h.Vector().record(mylinatedAxon(0.5)._ref_i_membrane)

time = h.Vector().record(h._ref_t)
h.finitialize(-65 * mV)
h.continuerun(10 * ms)

plt.plot(time, voltage)
plt.xlabel('t (ms)')
plt.ylabel('Volage (mV)')
plt.title("Voltage of NEURON model with length and diamter of comparments multiplied by .5")
#plt.title("Voltage of NEURON model with a 10x channel increase in non-myelinated axon")
plt.show()

plt.figure()
plt.plot(time, curDend)
plt.plot(time, curSoma)
plt.plot(time, curNonMylinated)
plt.plot(time, curMylinated)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon
Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current densities (mA/cm^2)')
plt.title("Current densities of NEURON model with length and diamter of comparments
multiplied by .5")
#plt.title("Current densities of NEURON model with a 10x channel increase in non-myelinated
axon")


plt.figure()
plt.plot(time, curDend * 3.14 * ((21/2 * um)**-4)**2)
plt.plot(time, curSoma * 3.14 * ((12/2 * um)**-4)**2)
plt.plot(time, curNonMylinated * 3.14 * ((1/2 * um)**-4)**2)
plt.plot(time, curMylinated* 3.14 * ((1/2 * um)**-4)**2)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current (mA)')
#plt.title("Current densities of NEURON model with length and diamter of comparments multiplied by 2")
plt.title("Currents of NEURON model with length and diamter of comparments multiplied by .5")

print("Minimum non-mylinated value:",min(curNonMylinated))
print("Minimum Soma:",min(curSoma))
print("Difference between soma and non-mylinated value", min(curNonMylinated) - min(curSoma))

print("Minimum non-mylinated value:",min(curNonMylinated * 3.14 * ((1/2 * um)**-4)**2))
print("Minimum Soma:",min(curSoma * 3.14 * ((12/2 * um)**-4)**2))
print("Difference between soma and non-mylinated value", min(curNonMylinated * 3.14 * ((1/2 * um)**-4)**2) - min(curSoma * 3.14 * ((12/2 * um)**-4)**2))

soma = h.Section(name = 'soma')
soma.L = 24 * um * 2
soma.diam = 21 * um * 2
soma.nseg = 100
soma.insert('hh')
soma.insert('pas')
soma.insert('extracellular')

dendrite = h.Section(name = 'dendrite')
dendrite.L = 50 *  um * 2
dendrite.diam = 12 * um* 2
dendrite.nseg = 222
dendrite.insert('hh')
dendrite.insert('pas')
dendrite.insert('extracellular')
```

```
nonMylinatedAxon = h.Section(name = 'nonMylinatedAxon')
nonMylinatedAxon.L = 16 * um * 2
nonMylinatedAxon.diam = 1 * um * 2
nonMylinatedAxon.nseg = 100
nonMylinatedAxon.insert('hh')
nonMylinatedAxon.insert('pas')
nonMylinatedAxon.insert('extracellular')


mylinatedAxon = h.Section(name = 'mylinatedAxon')
mylinatedAxon.L = 300 * um * 2
mylinatedAxon.diam = 1 * um * 2
mylinatedAxon.nseg = 100
mylinatedAxon.insert('pas')
mylinatedAxon.insert('extracellular')
mylinatedAxon.cm = 0.04

axonHillock = h.Section(name = 'axonHillock')
axonHillock.L = 16 * um * 2
axonHillock.nseg = 9

diams = np.arange(1, 21, 2.22222222)
diams = diams[::-1]
for i in range(0,10):
  axonHillock(0.1 * i).diam = diams[i] * um * 2


axonHillock.insert('hh')
axonHillock.insert('pas')
axonHillock.insert('extracellular')

dendrite.connect(soma)
soma.connect(axonHillock)
```

```
axonHillock.connect(nonMylinatedAxon)
nonMylinatedAxon.connect(mylinatedAxon)

'''
for seg in nonMylinatedAxon:
  seg.hh.gnabar = seg.hh.gnabar * 10
  seg.hh.gkbar = seg.hh.gkbar * 10
'''

iclamp = h.IClamp(dendrite(0.5))
iclamp.delay = 1
iclamp.dur = 0.1
#iclamp.amp = 10
iclamp.amp = 37
voltage = h.Vector().record(soma(.1)._ref_v)

curDend = h.Vector().record(dendrite(0.5)._ref_i_membrane)
curSoma = h.Vector().record(soma(0.5)._ref_i_membrane)
curNonMylinated = h.Vector().record(nonMylinatedAxon()._ref_i_membrane)
curMylinated = h.Vector().record(mylinatedAxon(0.5)._ref_i_membrane)

time = h.Vector().record(h._ref_t)
h.finitialize(-65 * mV)
h.continuerun(10 * ms)

plt.plot(time, voltage)
plt.xlabel('t (ms)')
plt.ylabel('Volage (mV)')
#plt.title("Voltage of NEURON model with length and diamter of comparments multiplied by 2")
plt.title("Voltage of NEURON model with length and diamter of comparments multiplied by 2")
plt.show()

plt.figure()
plt.plot(time, curDend)
plt.plot(time, curSoma)
plt.plot(time, curNonMylinated)
plt.plot(time, curMylinated)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current densities (mA/cm^2)')
#plt.title("Current densities of NEURON model with length and diamter of comparments multiplied by 2")
plt.title("Current densities of NEURON model with length and diamter of comparments multiplied by 2")

plt.figure()
plt.plot(time, curDend * 3.14 * ((21/2 * um)**-4)**2)
plt.plot(time, curSoma * 3.14 * ((12/2 * um)**-4)**2)
plt.plot(time, curNonMylinated * 3.14 * ((1/2 * um)**-4)**2)
plt.plot(time, curMylinated* 3.14 * ((1/2 * um)**-4)**2)
plt.legend(['Dendrite Current','Soma Current','Non-Mylinated Axon Current', 'Mylinated Axon Current'])
plt.xlabel('t (ms)')
plt.ylabel('Current (mA)')
#plt.title("Current densities of NEURON model with length and diamter of comparments multiplied by 2")
plt.title("Currents of NEURON model with length and diamter of comparments multiplied by 2")

print("Minimum non-mylinated value:",min(curNonMylinated))
print("Minimum Soma:",min(curSoma))
print("Difference between soma and non-mylinated value", min(curNonMylinated) - min(curSoma))

print("Minimum non-mylinated value:",min(curNonMylinated * 3.14 * ((1/2 * um)**-4)**2))
print("Minimum Soma:",min(curSoma * 3.14 * ((12/2 * um)**-4)**2))
print("Difference between soma and non-mylinated value", min(curNonMylinated * 3.14 * ((1/2 * um)**-4)**2) - min(curSoma * 3.14 * ((12/2 * um)**-4)**2))
```