TOTAL: 83/89

Homework Assignment 2

Part 1 – Binary search algorithm in NEURON

Commented [NP1]: 9/10

A binary search algorithm starts at the middle element of an ordered array and checks whether the middle element is greater or less than the search element. If the elements match, usually the index or position is returned. Otherwise, this process is repeated on the upper or lower half of the ordered array based on whether or not the array is ordered in ascending or descending value.

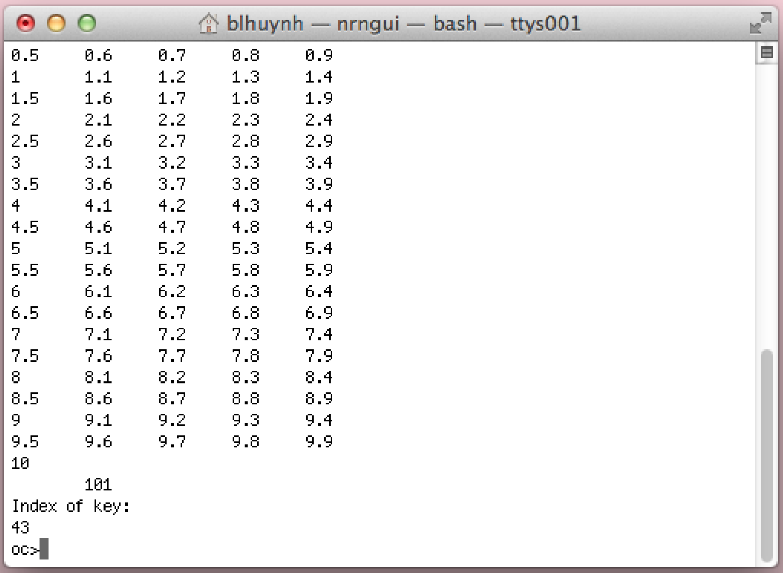
For pseudo code of a binary search algorithm for an ascending array (i.e. `array=[1 2 3]`), the following recursive method can be used to identify the index of the search value given that the search value exists within the array.

```
binarySearch(array, value)
      define midpoint

      if midpoint==value
            return index of midpoint
      if midpoint > value
            return binarySearch(array[0:midpoint],value)
      if midpoint < value
            return binarySearch(array[midpoint:end],value)
```

Unfortunately, it was difficult to implement a function in hoc that allowed for an array as an input so the following iterative method was used to determine the index of a value in an array.

```
// ---- ITERATIVE BINARY SEARCH ALGORITHM ----
// (implemented w/ zero-indexed programming)
vector = [0:0.1:10]
key = 4.3

minimumIndex = 0
maximumIndex = vector.size()-1

while (maximumIndex >= minimumIndex){
      middleIndex = (maximumIndex - minimumIndex) / 2 + minimumIndex
      middleValue = vector[middleIndex]

      if (middleValue == key){ return middleIndex }
      else if (middleValue < key) { minimumIndex = middleIndex + 1 }
      else if (middleValue > key) { maximumIndex = middleIndex - 1 }
}
```

Commented [NP2]: Where's your final answer?



```
0.5     0.6     0.7     0.8     0.9
1       1.1     1.2     1.3     1.4
1.5     1.6     1.7     1.8     1.9
2       2.1     2.2     2.3     2.4
2.5     2.6     2.7     2.8     2.9
3       3.1     3.2     3.3     3.4
3.5     3.6     3.7     3.8     3.9
4       4.1     4.2     4.3     4.4
4.5     4.6     4.7     4.8     4.9
5       5.1     5.2     5.3     5.4
5.5     5.6     5.7     5.8     5.9
6       6.1     6.2     6.3     6.4
6.5     6.6     6.7     6.8     6.9
7       7.1     7.2     7.3     7.4
7.5     7.6     7.7     7.8     7.9
8       8.1     8.2     8.3     8.4
8.5     8.6     8.7     8.8     8.9
9       9.1     9.2     9.3     9.4
9.5     9.6     9.7     9.8     9.9
10
        101
Index of key:
43
oc>
```

Figure 1: Binary Search Algorithm implemented in NEURON

Part 2 – Model axon in NEURON

Commented [NP3]: 14/16

Given – Fiber diameter $D = 8\ \mu m$
Diameter of node: $d = 0.7 * D = 5.6\ \mu m$
Length of myelin: $L = 100 * D = 800\ \mu m = 0.8\ mm$

Based on $V_m(t)$ with $I_{stim} = 0$, V_{rest} or $v_{init} = -65\text{mV}$ by inspection.

The internodal or axoplasmic resistance, R_a, is calculated using the equation

$$R_a = \frac{4\rho_a L}{\pi d^2} = \frac{4\rho_a * 100D}{pi * (0.7D)^2}$$

where ρ_a is the axoplasmic resistivity, L is the internodal length, d is the node diameter, and D is the myelin diameter. Since consecutive nodes of Ranvier are separated by internodal spaces, the internodal resistance, R_a, can represent these spaces if the myelin is assumed to be a perfect insulator and the internode is modeled as a tube of axoplasm. Therefore, the product of the axoplasmic resistivity and internodal length can be divided by the cross-sectional area of the nerve fiber to calculate the internodal resistance. In Neuron, the myelin is constructed by connecting nodes of Ranvier with resistors by modeling the intracellular space that represent not just the length of the node but also the length of the myelin as well.

Shouldn't have "g_hh" in code. HH electrical parameters are part of HH model.
Record stim.i with vector.record function.

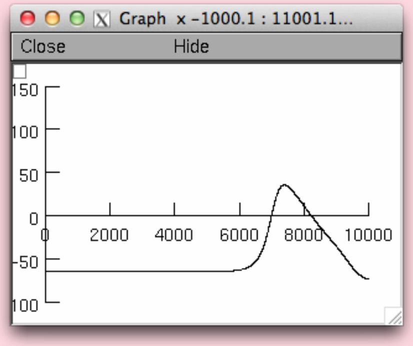


Figure 2: Vm(t) at 45th node

Commented [NP4]: Should increase tmax to see full AP recovery.

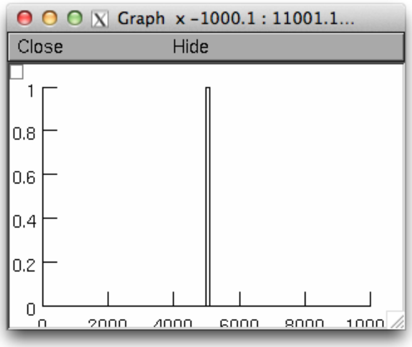
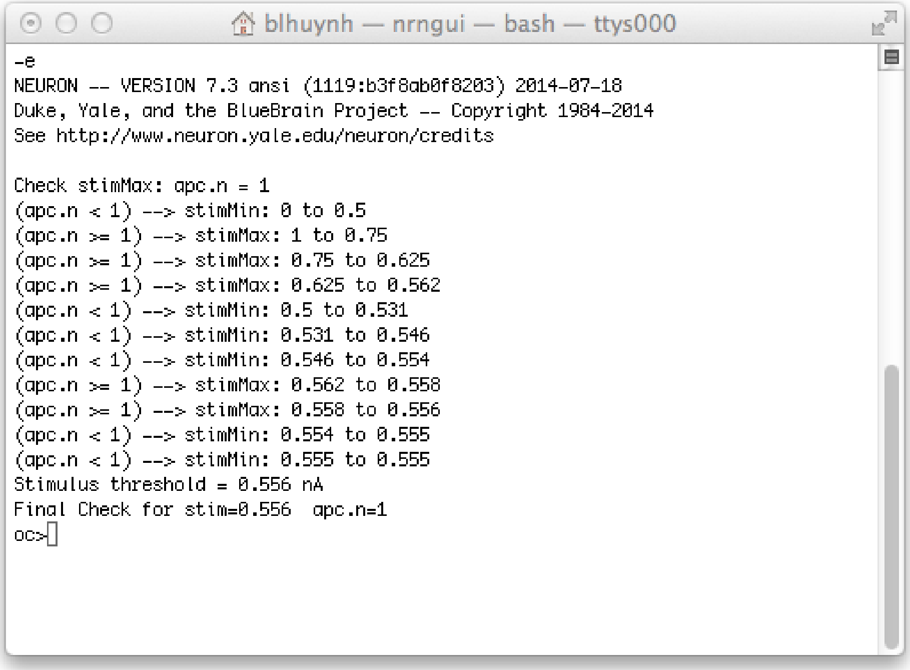


Figure 3: Istim(t)

Part 3 – Intracellular threshold with PW = 0.1ms

Commented [NP5]: 8/8

The binary search algorithm was paired with the modeled axon to determine the minimum threshold to stimulate the axon. Using a resolution of 1 pA, the minimum threshold was calculated to be $I_{\{stim,th\}} = 556\,pA$. The upper bound of 1 nA was determined to be superthreshold and the final value of $I_{stim,th} = 556$ pA was also determined to be superthreshold. A screenshot of NEURON is shown in Figure 4.



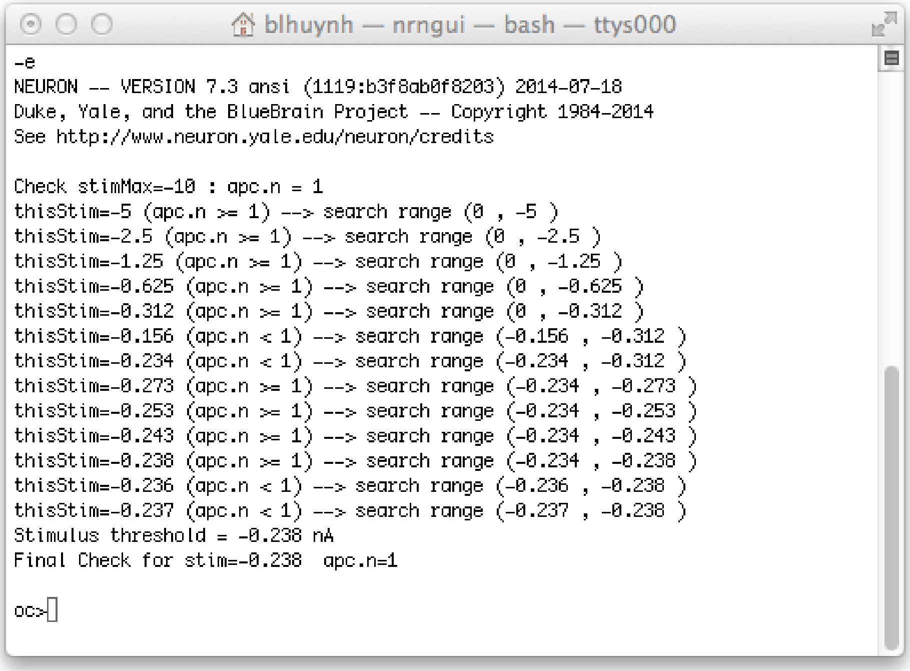
```
-e
NEURON -- VERSION 7.3 ansi (1119:b3f8ab0f8203) 2014-07-18
Duke, Yale, and the BlueBrain Project -- Copyright 1984-2014
See http://www.neuron.yale.edu/neuron/credits

Check stimMax: apc.n = 1
(apc.n < 1) --> stimMin: 0 to 0.5
(apc.n >= 1) --> stimMax: 1 to 0.75
(apc.n >= 1) --> stimMax: 0.75 to 0.625
(apc.n >= 1) --> stimMax: 0.625 to 0.562
(apc.n < 1) --> stimMin: 0.5 to 0.531
(apc.n < 1) --> stimMin: 0.531 to 0.546
(apc.n < 1) --> stimMin: 0.546 to 0.554
(apc.n >= 1) --> stimMax: 0.562 to 0.558
(apc.n >= 1) --> stimMax: 0.558 to 0.556
(apc.n < 1) --> stimMin: 0.554 to 0.555
(apc.n < 1) --> stimMin: 0.555 to 0.555
Stimulus threshold = 0.556 nA
Final Check for stim=0.556  apc.n=1
oc>
```

Figure 4: Threshold Finder via NEURON Binary Search Algorithm

Part 4 – Extracellular threshold with PW = 0.1 ms

A binary search algorithm was applied to the same modeled axon but with extracellular stimulation with extracellular resistivity $\rho_e = 1000\,\Omega * cm$. A cathodic stimulus was also used with a pulse width of 0.1 ms from an electrode 1 mm away in the perpendicular direction from the middle node of Ranvier. A screenshot of NEURON finding the threshold to be $I_{stim,th} = -238$ ~~up~~A is shown in Figure 5.

Commented [NP6]: 9/10
Units are off.



```
-e
NEURON -- VERSION 7.3 ansi (1119:b3f8ab0f8203) 2014-07-18
Duke, Yale, and the BlueBrain Project -- Copyright 1984-2014
See http://www.neuron.yale.edu/neuron/credits

Check stimMax=-10 : apc.n = 1
thisStim=-5 (apc.n >= 1) --> search range (0 , -5 )
thisStim=-2.5 (apc.n >= 1) --> search range (0 , -2.5 )
thisStim=-1.25 (apc.n >= 1) --> search range (0 , -1.25 )
thisStim=-0.625 (apc.n >= 1) --> search range (0 , -0.625 )
thisStim=-0.312 (apc.n >= 1) --> search range (0 , -0.312 )
thisStim=-0.156 (apc.n < 1) --> search range (-0.156 , -0.312 )
thisStim=-0.234 (apc.n < 1) --> search range (-0.234 , -0.312 )
thisStim=-0.273 (apc.n >= 1) --> search range (-0.234 , -0.273 )
thisStim=-0.253 (apc.n >= 1) --> search range (-0.234 , -0.253 )
thisStim=-0.243 (apc.n >= 1) --> search range (-0.234 , -0.243 )
thisStim=-0.238 (apc.n >= 1) --> search range (-0.234 , -0.238 )
thisStim=-0.236 (apc.n < 1) --> search range (-0.236 , -0.238 )
thisStim=-0.237 (apc.n < 1) --> search range (-0.237 , -0.238 )
Stimulus threshold = -0.238 nA
Final Check for stim=-0.238  apc.n=1

oc>
```

Figure 5: Extracellular Threshold with PW = 0.1 ms

Part 5 – Threshold-fiber diameter

Commented [NP7]: 14/15
Units again

Fibers with a diameter of 1-15 μm in 2 μm increments were stimulated extracellularly. With increasing fiber diameter, the spacing between the nodes of Ranvier also increased, resulting in larger transmembrane potential differences and resulting in a lower activation threshold. The threshold-diameter curve can be approximated by

$$I_{th}(D) = I_D + \frac{a}{\sqrt{D}}$$

where the two constants, a and I_D, are determined empirically. The recorded data is listed in Table 1 and plotted in Figure 6.

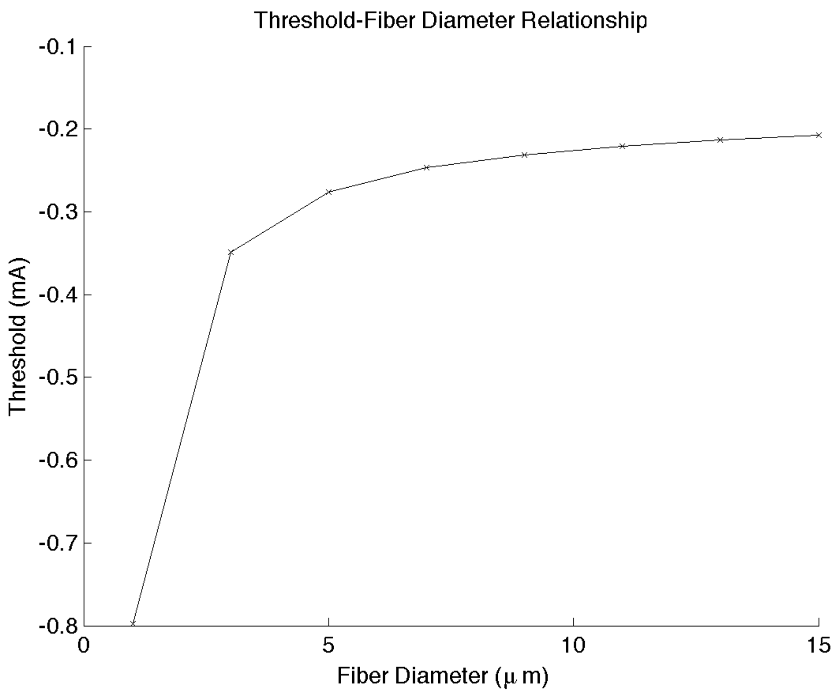


Figure 6: Threshold-Fiber Diameter Relationship

Table 1: Threshold-Fiber Diameter Data

Fiber Diameter (μm)	1	3	5	7	9	11	13	15
Threshold (~~n~~mA)	-0.798	-0.349	-0.276	-0.247	-0.231	-0.221	-0.213	-0.207

Part 6 – Threshold-distance relationship

Commented [NP8]: 14/15

Using an 8 μm diameter axon and PW=0.1 ms, the current thresholds for the 7 following electrode-fiber distances were recorded: {0.1, 0.2, 0.5, 1, 2, 5, 10} mm. Transmembrane potentials generated by extracellular stimulation are larger when the electrode is closer to the neuron. As the distance between the neuron and electrode increases, the stimulation amplitude required to activate the neuron increases by

$$I_{th} = I_R + k * r^2$$

for I_R the offset which determines the absolute threshold, slope k which determines the threshold difference between fibers at different distances, stimulation threshold I_{th}, and electrode-neuron distance r. The recorded data is listed in Table 2 and shown in Figure 7.

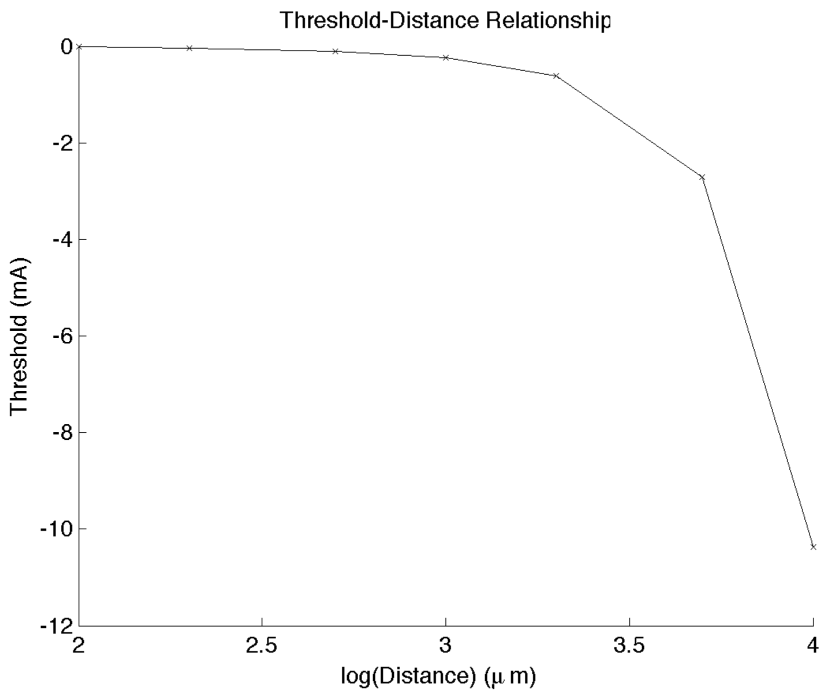


Figure 7: Threshold-Distance Relationship

Table 2: Threshold-Distance Data

Electrode-Fiber Distance (mm)	0.1	0.2	0.5	1	2	5	10
Threshold (m~~n~~A)	-0.016	-0.034	-0.099	-0.236	-0.609	-2.702	-10.0366

Part 7 – Application

Commented [NP9]: 15/15

Shannon and colleagues investigated the threshold-distance relationship for electrical stimulation using an auditory prosthesis to compare with postmortem threshold-distance measurements. This allows for a better estimation of current spread to better understand localization effects of electrical stimulation. Using an auditory brainstem implant (ABI), electrical stimulation is directly applied to the auditory structures in the brainstem of patients who have lost hearing ability due to large, bilateral, eighth nerve tumors. Threshold data were obtained from a single subject with sinusoidal and biphasic pulses and electrode position data were determined postmortem. As stimulation amplitudes are increased, there is a higher likelihood of activating non-relevant structures that are not related to auditory processes. It was concluded that the distances between the electrode to the nonauditory nuclei were sufficiently large enough such that presently used stimulation amplitudes would not activate those regions. The relationship between electrical stimulation threshold and electrode-fiber distance was consistent with the function first postulated by Ranck in 1975. The consistency despite varying models of the brain and stimulation waveforms is encouraging and provides confidence in the ability to predict regions of activation for future methods in electrical stimulation for better selectivity and patient comfort.

Shannon, R., Moore, J., Mccreery, D., & Portillo, F. (1997). Threshold-distance measures from electrical stimulation of human brainstem. *IEEE Transactions on Rehabilitation Engineering, 5*(1), 70-74. Retrieved October 1, 2014, from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=559351

Code & Output
Part 1 – Binary Search Algorithm in NEURON (binarySearchAlgorithm.hoc)
```
load_file("nrngui.hoc")
func round(){
	if ($1>0){
		return int($1+0.5)
	} else {
		return int($1-0.5)
	}
}

objref v1
proc createVector(){
	startVal = $1
	stopVal  = $2
	stepVal  = $3
	numel    = (stopVal-startVal)/stepVal
	v1       = new Vector(numel+1)
	v1.x[0]  = startVal

	for i=1,numel{
		v1.x[i] = v1.x[i-1]+stepVal
	}
}

// Create list of numbers 0:0.1:10 to search from
createVector(0,10,0.1)
findValue = 4.3
v1.printf()

// ---- ITERATIVE binary search ----
```

```
proc iterative(){
      key  = $1
      imin = 0
      imax = v1.size()-1

      while (imax >= imin){
            imid = round((imax-imin)/2 + imin)
            midValue = v1.x[imid]

            if (midValue == key){
                  // print imid
                  break
            } else if (midValue < key) {
                  imin = imid + 1
            } else if (midValue > key) {
                  imax = imid - 1
            }
            // print "¥n"
      }
      if (midValue != key) {
            print "Key not found."
      } else {
            print "Index of key: "
            print imid
      }
}
iterative(findValue)
```

Part 2 – Model axon in NEURON

```
//load_file("nrngui.hoc")
//load_proc("nrnmainmenu")

// ************************* Model specification
******************************
proc params() {
      // Geometrical properties
      D = 8
      num_nodes = 51                  // number of nodes [unitless]
      node_diam = D*0.7         // node diameter [um]
      node_length = 1                 // node length [um]
      myelin_length = 100*D       // internodal length [um]
```

```
	// Electrical properties
	node_cm = 2				// specific membrane capacitance 
[uF/cm^2]
	rhoa = 200				// intracellular resistivity [ohm-
cm]
	node_Rm = 1500				// specific membrane resistance 
[ohm-cm^2]
	ap_thresh = 0				// action potential threshold

	// Stimulus parameters
	mydel = 5				// start at t=5ms [ms]
	myamp = 1.0				// amplitude [nA]
	mydur = 0.1				// duration, aka pulsewidth [ms]

	// Temporal parameters
	dt = 0.001				// [ms]
	tstop = 10				// [ms]
	num_timesteps = int(tstop/dt) + 1

	// Other parameters
	v_init = -65				// [mV]
	celsius = 6.3				// [deg C]
}
params()

// ************************* Model initialization 
****************************
create axon[num_nodes]
proc initialize() {local i
	for i = 0, num_nodes - 1 {
		axon[i] {
			nseg  = 1
			diam  = node_diam
			L		= node_length
			Ra		= rhoa * 
((node_length+myelin_length)/node_length)
			cm		= node_cm

			// Insert passive channel
			insert hh
```

```
                  g_hh = 1/node_Rm // do we need to change this from g_pas ->
g_hh?

            }
      }

      for i = 0, num_nodes - 2 {
            connect axon[i](1), axon[i+1](0)
      }
}
initialize()

// ************************* Instrumentation
*********************************
// Intracellular stimulation
objref stim
proc int_stim() {
      axon[int(num_nodes/2)] { // changed stimulus location to center of the
axon (10 -> )
            stim = new IClamp()
            stim.loc(0.5)
            stim.del = mydel
            stim.amp = myamp
            stim.dur = mydur
      }
}
int_stim()

// Record Vm(t) at all nodes
objref Vm_vec[num_nodes], Istim_vec
Istim_vec = new Vector(tstop/dt)
for i = 0, num_nodes - 1 {
      Vm_vec[i] = new Vector(num_timesteps,0)
      Vm_vec[i].record(&axon[i].v(0.5),dt)
}

// add APCount object to node 20
objref apc
axon[19] apc = new APCount(0.5)
apc.thresh = 20
```

```
// ************************** Simulation control
*******************************
proc stimul() {
      finitialize(v_init)
      while(t<tstop) {
            fadvance()
            Istim_vec.x[t/dt-1] = stim.i
            if (stim.i > 1){
                  print stim.i
            }
      }
}
stimul()

print "apc.n = ",apc.n

// ************************** Data analysis & output
***************************
// Plot Vm(t) at the 45th node
objref g1
proc plot_data() {
      g1 = new Graph()
      g1.size(0, num_timesteps, -100, 150)
      // Vm_vec[int(num_nodes/2)].plot(g1)
      Vm_vec[44].plot(g1)
}
plot_data()

// Plot I_stim(t)
objref g2
proc plot_Istim(){
      g2 = new Graph()
      g2.size(0, num_timesteps,0,1)
      Istim_vec.plot(g2)
}
plot_Istim()

// plot Vm(t) at 20th node
// objref g3
// proc plot_data_node20() {
//    g3 = new Graph()
//	g3.size(0, num_timesteps, -100, 150)
//	// Vm_vec[int(num_nodes/2)].plot(g1)
//	Vm_vec[19].plot(g3)
// }
// plot_data_node20()

// plot_data()
```

Part 3 – Intracellular threshold with PW = 0.1 ms (threshold procedure)

```
proc iterative(){
	count=0
	while (1) {
		thisStim = cutoff((stimMax-stimMin)/2+stimMin)
		myamp = thisStim
		int_stim()
		stimul()

		if (apc.n >= 1){
			print "(apc.n >= 1) --> stimMax: ",stimMax,"to ",thisStim
			stimMax = thisStim
		} else if (apc.n < 1){
			print "(apc.n < 1) --> stimMin: ",stimMin,"to ",thisStim
			if (stimMin == thisStim) { print "Stimulus threshold = 
",thisStim+0.001,"nA" }
			stimMin = thisStim
		}
		count+=1
		if (count>10){break}
	}
}
```