Practical 1: NEURON simulation environment and passive properties of neurons

David Sterratt
2010 version by Mark van Rossum and James A. Bednar
2019 version by Matthias Hennig and Theoklitos Amvrosiadis

October 1, 2019

1 Aims

This practical has two aims:

- To get you acquainted with the NEURON simulator
- To gain an intuition of the equations for the passive properties of neurons and apply them in practice.

2 Starting and stopping NEURON

How you run NEURON depends on the operating system you are using.

2.1 DICE network (LINUX)

NEURON is normally installed on DICE in */usr/bin/*. To start NEURON, type

nrngui

at the UNIX command line. On other systems it may be in another location, such as */usr/i686/bin/* or */opt/nrn/i686/bin/*, in which case you can add that directory to your PATH or type the full path when running the command:

/usr/i686/bin/nrngui.

Some text about the version number of NEURON etc. should appear. At the end of the text you should see a prompt like this:

oc>

This prompt tells you that anything you type at the prompt will be interpreted by NEURON. NEURON understands commands in a programming language called HOC (pronounced hoak). To get out of NEURON, type *quit()* or Ctrl-D. You should see the operating system command prompt again.

A new window entitled **Main Menu** also appears. It provides access to the Graphical User Interface (GUI) and looks like this:

2.2 Other operating systems

NEURON also runs on Windows and MacOS operating systems. Go to `https://www.neuron.yale.edu/neuron/download` for free download.

2.3 Help and more info

Go to `https://www.neuron.yale.edu/neuron/docs` for full documentation.

3 Creating a single-compartment model

We will first create the simplest possible model of a patch of membrane: a single passive compartment. Figure 1 below shows the equivalent circuit. It comprises a membrane capacitance C_m, a membrane resistance R_m, a leakage reversal potential E_L and a current source I_e. We will imagine our circuit represents a soma, though it misses some crucial features of a real soma, namely active properties.

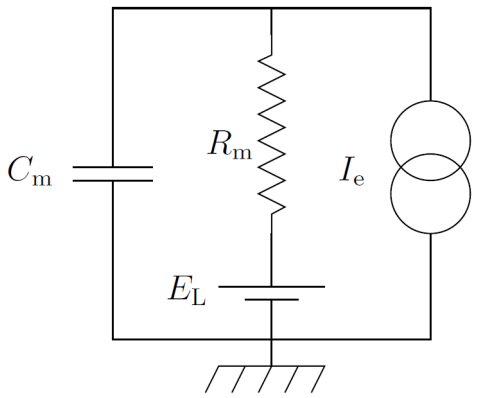


Figure 1: The single passive compartment

Start NEURON as explained in the previous section. Next, type:

create soma

This creates a *section* called **soma.** In NEURON a section is a geometrical unit rather than an electrical one. Each section is divided into one or more *segments.* Segments represent small patches of membrane which we assume to have the same membrane potential - they are another name for the *compartments* of a compartmental model.

Next, type:

access soma

This means that whenever we refer to properties of a section, NEURON will by default assign them to our soma section. In practice, it's better to refer to sections explicitly all the time rather than relying on the default section. However, we need at least one access statement for the GUI to work.

Now, type:

soma psection()

This prints out the properties of the section we have just created (though in a somewhat strange format; this is because the output is designed so that HOC as well as humans can read it). When NEURON creates a section, it gives it default values, as shown in Table 1.

Variable name	Meaning	Value	Units
nseg	Number of segments in a section	1	
L	Length of section l	100	μm
Ra	Specific axial resistance r_a	35.4	$\Omega * cm$
diam	Diameter d	500	μm
cm	Specific membrane capacitance c_m	1	$\mu F/cm^2$

Table 1: Default properties of a section

So far, we have introduced two passive parameters, the specific membrane capacitance c_m and the specific axial resistance r_a. We have still to introduce the specific leakage conductance $\overline{g}_L$ and its associated leakage reversal potential E_L. To do this we must insert channels into the membrane. NEURON has a special channel type called *pas* to simulate the membrane resistance and reversal potential. To insert it into the membrane, type:

soma insert pas

If you now type *psection()*, you should notice that there are two extra properties, as shown in Table 2.

Variable name	Meaning	Value	Units
g_pas	Specific leakage conductance $\overline{g}_L$	0.001	S/cm^2
E_pas	Leakage reversal potential E_L	-70	mV

Table 2: Default properties of the *pas* channel

The specific membrane conductance is often given in terms of the specific membrane resistance r_m. Since conductance is the inverse of resistance ($g = 1/R$), what is the value of the membrane resistance? (try to calculate)[1]. A more typical value for the specific membrane resistance is $r_m = 10k\Omega * cm^2$. What would be the value of the specific membrane conductance in this case?[2].

To change this, type

soma g_pas = (new value of conductance)

To confirm that you have changed the properties, type *soma psection()* again.

4 Injecting current into a single compartment

We have now built a single compartment with passive leakage. To test its properties we will inject some current. To do this, type:

objref stim
soma stim = new IClamp(0.5)

The first line creates a new reference to an *object* called **stim**. The second line makes **stim** refer to an **IClamp** object placed *halfway (0.5)* down the **soma**. The job of the **IClamp** (current clamp) object is to inject current into the section. In order to do this, you need to set some properties of the IClamp as follows:

stim.amp = 1
stim.del = 100
stim.dur = 100

This will inject a pulse of current with an amplitude of 1 nA (**stim.amp**), starting with a delay of 100ms (**stim.del**) after the simulation start, and lasting for 100ms (**stim.dur**). Note that we cannot refer to the **IClamp** **stim** in the same way that we can a section. Therefore, we have to refer explicitly to **stim** whenever we want to set its properties using the *dot* notation.

[1]The answer is $r_m = 1/0.001\Omega * cm^2 = 1000\Omega * cm^2 = 1k\Omega * cm^2$
[2]The answer is $\bar{g}_L = 1/10000 S/cm^2 = 0.0001 S/cm^2$

5 Running the simulation from the HOC command line

Now, all that remains to be done is to run the simulation.[3]

Before we run the simulation, look at the membrane potential of our compartment by typing:

soma print v

Note down the value, and now type:

run()

This runs the simulation for 5ms of simulated time. Now look at the membrane potential again; you should find that it has changed. You can set the duration of the simulation by changing the value of the global variable **tstop**, as follows:

tstop = 300

The next time you type *run()*, the simulation will run for 300ms of simulated time.

6 Running the simulation from the GUI

Of course, it would be tedious to run a simulation in the way described in the previous section. To see our results plotted in graphs we need to use the graphical **Main Menu** that popped up when NEURON started.

This window contains menus **File**, **Edit**, **Build**, **Tools**, **Graph**, **Vector** and **Window**. For our purposes here, we will explore the GUI methods for running a simple simulation. Under the **Tools** menu select **RunControl**. This generates a window (see below) that allows you to control primary parameters for running the simulation.

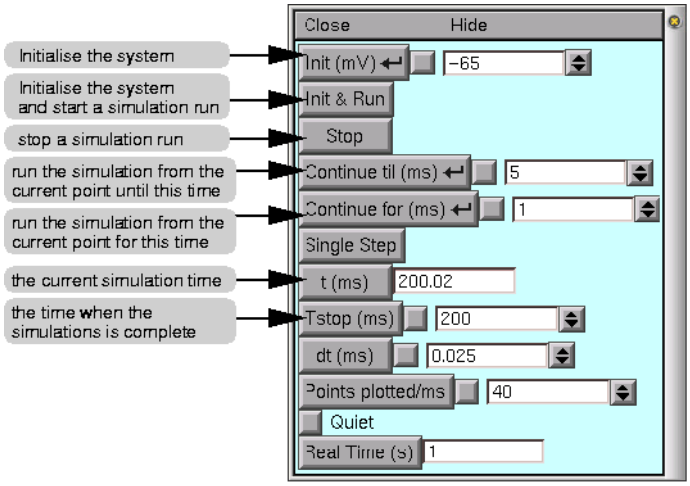


Figure 2: The **RunControl** window

The **RunControl** window has a common format, as do many other windows you will encounter. There are buttons, such as **Init & Run** which will run the simulation, and *field editors* next to some buttons (e.g. next to our **tstop** variable button). Whenever you see this type of box, you can enter a new value by simply clicking into the field editor box and start typing. A red vertical bar will appear in the box and the field name will be highlighted with yellow to signify that you are editing the value of that parameter. After you have entered the new number

[3]We need to load some standard libraries of functions into NEURON by typing *load file("nrngui.hoc")* In modern versions of NEURON this library is normally loaded on startup, so the line is redundant. However it is needed for portability (Windows users who click on HOC files from Windows Explorer don't get this unless the file contains the statement).

(or expression), you need to tell NEURON to accept the number (or expression) you just entered. You can do this by either pressing the **Enter** key, by clicking on the button associated with the field editor or, if there is another button (called a *check box*) between the main button and the field editor box, you can click on that button. The check box is used to toggle between the default value and your changed value. Thus, if you change the value of **tstop**, a check mark will appear in the check box signifying that you have changed the value of this parameter. By pressing the check box, you can toggle between the default value and the changed value.

7 Viewing the results graphically

NEURON can produce plots versus time on a number of different types of axes: the **Voltage axis**, the **Current axis** and the **State axis**. In this section we will look at the voltage and current axes. [4]

7.1 Voltage axis

In addition to controlling the simulation, we will want to observe the voltage changes in the soma over the period of simulation (300ms in our current example). Under the **Graph** menu of the main window, select **Voltage axis**. This generates another window that will display voltage. When there is more than a single compartment (e.g. when we have added dendrites) then we have to specify which section's voltage to plot. In the current example, as we have set the soma as the default section in our commands above, the soma voltage will be plotted by default. You can open as many voltage or other graph windows as you like.

Now run the simulation (using the **Init & Run** button in the **RunControl** window), observing the voltage being graphed.

To look at the graph in more detail, click the right mouse button over the *Voltage Axis*. Select **View. . .** and then **Zoom in/out** from the menu that pops up. (Throughout the rest of this document I will abbreviate selecting from submenus with an arrow. For example **"View. . .→Zoom in/out"** means "select **View. . .** and then **Zoom in/out**".)

- Now clicking on the plot with the left mouse button and dragging the pointer *upwards* **increases** the *vertical* scale; dragging *downwards* **decreases** the *vertical* scale. Similarly dragging *right* **increases** the *horizontal* scale and dragging *left* **decreases** it.
- Clicking on the plot with the middle mouse button (scroll wheel) and dragging moves the trace relative to the axes.

This is easier to do than to describe, so take some time to practise moving the plot around and zooming in and out. Once you have zoomed in, you should be able to see a plot like the one in Figure 3.

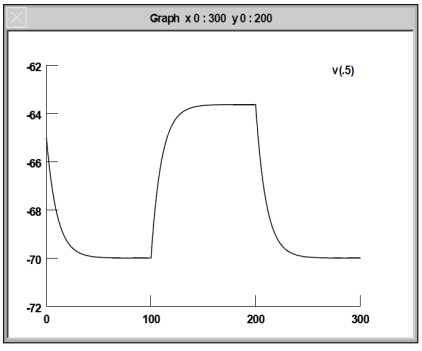


Figure 3: The voltage axis. The trace shows the membrane potential of the section plotted versus time.

[4] Each of these requires its own graph, since the method used to solve the cable equations, developed by Mike Hines, calculates each of these variables at different time points. For example, voltages are calculated at each time step, but currents are calculated at the half time step before voltage and the state variables are calculated at the half time step after the voltage. For more information on the numerical method used to solve these equations see Hines and Carnevale (1997).

7.2 Current axis

It is also useful to plot the different currents flowing through the membrane. Let's first plot the capacitive current, I_C. To do this, we must select **Graph⟶Current axis**. This brings up an empty plot. If you try to **Init & Run** no plot appears as it does in the case of the **Voltage axis**. This is because no quantities are selected by default. To select some currents to plot, right click on the current axis and chose **Plot what?** from the menu. This should bring up a window from which we can choose a variable to plot. Double-click on **soma.** in the first column. This should bring up a list of quantities in the second column. Click on **i_cap(0.5)** ("the capacitive current measured 0.5 of the way down the section"). In the text box at the top of the window[5] you should now see **soma.i_cap(0.5)**. Click on the **Accept** button. You should see the legend **i_cap(0.5)** appear on the current axis. Now run the simulation using **Init & Run**. Initially, it will look as though nothing is appearing on the axes. However, this is because the scale is too small. You could try zooming in as in the previous section. However, a convenient way of setting the scale is to right-click and select **View⟶View = plot**. You should end up with something like Figure 4.

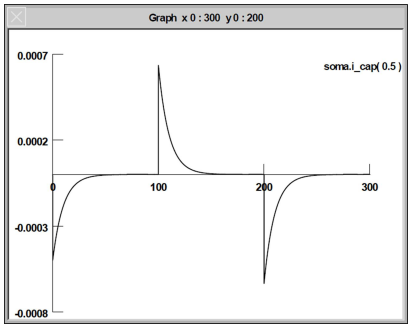


Figure 4: The current axis. The trace shows the capacitive current versus time.

7.3 Plotting more than one quantity on the same axes

Suppose we would now like to plot the leakage current through the cell membrane (the current flowing across the resistor R_m), in addition to the current onto the capacitor. To do this, select **soma.i_pas(0.5)** using the same procedure as in the last section.

Now when you run the simulation, you will see two traces. However, it would be nice if we could easily tell them apart. To change the colour and thickness of the lines you can right-click on the graph and select **Colour/brush**. This brings up a palette of colours and lines.

We select colours and/or line types by clicking the button next to the colour or line type and then clicking on the relevant legend. After we have run the simulation again, we should see something like Figure 5 below.

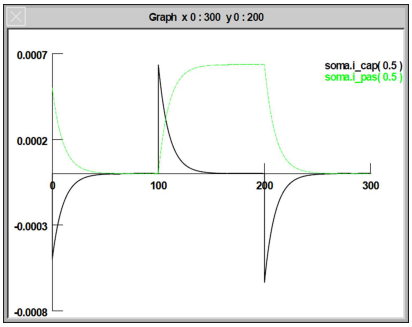


Figure 5: The capacitive current (black solid line) and the leakage current (green dotted line) are shown.

[5]The variables can instead be directly typed in if we know the exact name of the variable we want to plot.

8 Significance of the results

Let's try now to understand the behaviour of our simple model system and connect it with the equations you learned in lecture 2.

First, let's make a note of our model's parameters. On a piece of paper (or a digital form, if you prefer) you should make a table that looks like the following (you can omit the "Meaning" column):

Variable name	Meaning	Value	Units
c_m	Specific membrane capacitance		
C_m	Membrane capacitance		
r_m	Specific membrane resistivity		
R_{in} (or R_m)	Input resistance		
r_a	Specific axial resistance		
d	Section diameter		
$L(h)$	Section length (sometimes h for height)		
I_{inj} (or I_{ext})	Injected current amplitude		
$E_L(E_r)$	Leakage reversal potential or resting potential		
V_{ss}	Steady state membrane potential		
τ	Membrane time constant		

Table 3: Model parameters

These are fundamental properties of any computational model and it will be useful to work them out. Some of them you already know, either because they were set by default, or because you set them explicitly. You can refer to the previous sections of this document, or remember that you can simply type *psection()* in the command window and find out in that way. Be sure to get the units correct for the calculations that will follow.

8.1 Membrane potential variables

For the values of E_L and V_{ss} you can take a look at the voltage plot that you made previously. The leakage reversal potential is the point to which the membrane potential will settle when there are no currents injected into the cell (hence also *resting* potential). When did you first start injecting the current? (You can look at the current plot for a hint) What was the membrane potential right *before* that point?

On the other hand, as its name would suggest, the *steady state* potential is the point where the membrane potential has *stopped changing* during current injection. You can think of it as the boundary potential given the model parameters. What did the value of the membrane potential settle to *after* we started injecting current? Remember that you can zoom in/out of the plot if needed.[6]

8.2 Passive membrane properties

By now, you should be left with 3 unknown variables, C_m, R_{in}, and τ. These you will have to calculate. It would be good to open the slides of lecture 2 now, if you haven't already done so.

8.2.1 Membrane capacitance

First, remember that the specific capacitance c_m is the capacitance *per unit area* of the membrane.[7] Thus, initially, you need to calculate the area of the section you have created. Considering this is a cylindrical section, how would you calculate that?[8] Next, use the area you found to calculate the capacitance of our cell membrane. Take care to use matching units!

[6] The answers are $E_L = -70mV$ and $V_{ss} \simeq -63.65mV$

[7] So, $c_m = C_m/A$, where A is the area of the membrane.

[8] The area of a cylinder (not measuring the top and bottom circles) is given by $A = \pi * d * l$, where d and l are the diameter and length of the cylinder, respectively.

The specific membrane capacitance is relatively constant across different neuronal types and is not influenced greatly by differences in the protein content of the membrane.[9] Keeping that constant, what would happen to the cell's capacitance if we were to increase its area? Can you give a simple and intuitive explanation why?

8.2.2 Input resistance

You can calculate the input resistance in two ways and check whether the results agree.

One way is to use the voltage graph and the steady state equation for the membrane potential. The change from the resting potential of the cell to the steady state potential is solely caused by our current injection across the membrane resistance, R_{in} or R_m.[10] Based on the membrane potential values you found in section 8.1, what is the input resistance of the cell?

Alternatively, we can use our knowledge of the specific membrane resistivity, $r_m = A * R_m$. Given the area you calculated in section 8.2.1, what is the input resistance of the cell? Does this value match the one above? (If it doesn't, check the units!) With everything else constant, what would happen to the input resistance if the cell was larger? Why do you think that is? From Ohm's law, what does a high resistance mean for a neuron's excitability?

8.2.3 Membrane time constant

The membrane time constant, τ, reflects the speed with which the membrane potential changes. Alternatively, one can think of it as the temporal filtering property of the cell, as inputs that arrive with frequencies higher than the time constant are filtered out, since the membrane cannot integrate them in time (a typical low-pass filter).

Again, there are two ways of calculating this variable. First, from the time-dependent equations of membrane voltage, we have defined $\tau = R_m * C_m$, so it's a straightforward substitution of the values you calculated in the previous two sections.[11]

The second way is a graphical one, again taking into account the time-dependent solutions for the membrane voltage. From these, one can see that τ represents the time it takes for the membrane potential to reach 2/3 of the final steady state potential after current injection.[12] Equivalently, it is the time it takes for the membrane potential to fall from the steady state 2/3 of the way back to the resting potential.[13] Based on this information, try to identify the membrane time constant from the voltage graph (remember you can zoom as needed). Does its value match the one you found arithmetically above? What would it mean for a cell to have a longer time constant in terms of its "memory"?

9 Creating a cable

So far we have looked at the *temporal* passive properties of a single compartment. Now we wish to look at the *spatial* passive properties of a long stretch of axon modelled by a number of compartments connected together as shown in Figure 6. This chain of compartments is sometimes called a *cable*, because of its similarity to an electrical cable. To do this, we will first create a new section called **axon**, set it as the default section and insert passive properties. Type:

create axon
access axon
axon insert pas

To change the specific membrane conductance, diameter, length and number of segments, we will use a new syntax. It means "change the properties of **axon** in the curly brackets without typing **axon** on every line":

axon { diam=1 L=10000 nseg=51 g_pas=0.0001 }

[9] Gentet, Luc J. et al. Biophysical Journal , Volume 79 , Issue 1 , 314 - 320

[10] Such that $V_{ss} = E_r + R_{in} * I_{inj}$

[11] Would we get the same result with $\tau = r_m * c_m$? Why/why not?

[12] $V(t) = E_L + R_{in} * I_{inj}(1 - e^{-t/\tau})$, assuming the cell is at the reversal potential when the current injection starts.

[13] $V(t) = E_L + V_{ss} * e^{-t/\tau}$, assuming the cell is at the steady state voltage when the current injection stops.

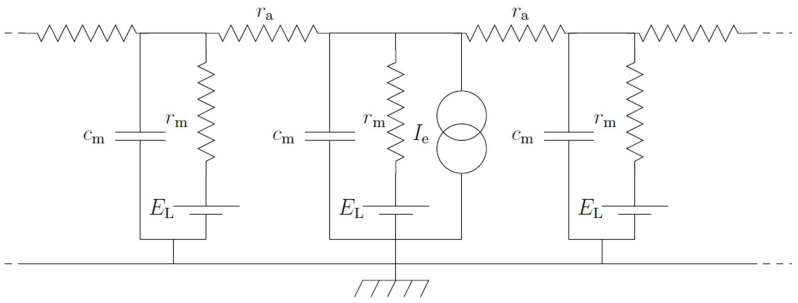


Figure 6: A stretch of axon modelled by single compartments linked together

This code changes the specific membrane conductance to $0.0001 S/cm^2$, the diameter to $1\mu m$ and the length to $10000\mu m (10mm)$. These values are typical for an axon in the mammalian CNS that projects to a different brain area. It also divides the axon into 51 *segments*. Each segment is the *compartment* of a compartmental model.

To inject current halfway along the axon, use the following code:

```
objref stim
axon stim = new IClamp(0.5)
stim.amp = 0.1
stim.del = 10
stim.dur = 200
```

10 Viewing the results

In section 7 we looked at a plot of the membrane potential as a function of *time*. In this part of the practical, we would like to look at how the membrane potential varies as function of *space*. To see what happens at all points in a section (or a group of sections), you can use a space plot. This dynamic graph plots a variable (for example, membrane potential) against space for each time step in the simulation.

To create a space plot, select **Shape plot** from the **Graph** menu in the **Main Menu**. This will open a shape plot window in which a schematic representation of the axon will appear. You can rotate the axes by selecting **3D Rotate** from the **Graph Properties** menu. The middle mouse button is used to move the whole representation. (This is more useful for neurons with complex morphology.)

From this window, you can select from the menu (with the right mouse button) a **Space Plot**. To create the space plot graph, you need to select a section of the neuron to include in the space plot by clicking the left mouse button at the beginning of the section (in the schematic picture), dragging the mouse across the section you want to plot (while holding the mouse button down), and releasing the mouse button when you have covered the sections you want to plot. A line will appear from where you first clicked the mouse button to the current location of the pointer. When you release the mouse button, the section you selected will be highlighted in colour, and a new window with the space plot will be opened. You should see that the x-axis of this graph stretches from 0 to 10000µm, the length of the section we created and that the membrane potential is uniformly $-65mV$ along the section.

In the RunControl window, set **tstop** to $50ms$ and then press the **Init & Run** button in the window to see the space plot in action. You should see that the membrane potential first of all falls gradually towards $-70mV$ and then, $10ms$ into the simulation, develops a peak of about $-51mV$ (Figure 7, right). The peak is halfway down the axon, at $5000\mu m$ and the membrane potential falls away exponentially to either side as shown in Figure 7 (left).

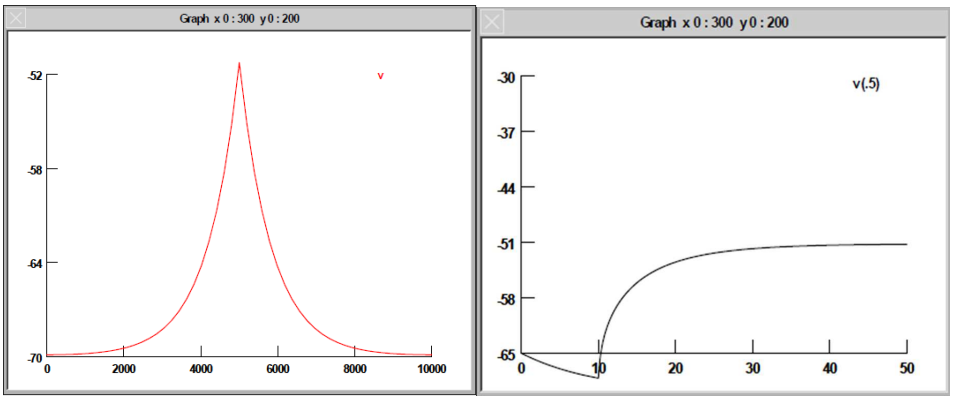


Figure 7: (left) The membrane potential as a function of distance and (right) time

11 Significance of the results

Let's calculate one last fundamental passive property of our cell, the *electrotonic length,* λ, also known as *length constant.* This represents the distance which potentials can passively "travel" before they decay. It depends on the properties of the compartment in which the potential is travelling and is given by $\lambda = \sqrt{\frac{r_m * d}{4 * r_a}}$. Since all of those properties are known for our axon, you can easily find out its value (keep in mind that they might be different from the soma values). Another way to find the length constant is graphically. Again, using the cable theory equations[14], one can see that λ is the distance from the current injection site where the voltage has dropped from the peak value $2/3$ of the way to the resting value. Check that the two ways produce similar results.

12 Dendritic filtering

So far we have looked at the temporal properties of a single compartment and the spatial properties of a cable in the steady state. Now we are going to look at the *spatiotemporal* properties of a cable. This is particularly relevant to dendrites, which don't tend to have as many active conductances as axons.

Most of the parameters of our dendritic cable will be the same as the axon we have just investigated. Therefore, we can change the parameters of the axon and imagine that it is a dendrite:

axon { L=1000 nseg=40 g_pas=0.0001 }

What we would like to so is measure the membrane potential from the end of the dendrite and inject current at different points along the dendrite.

To measure the membrane potential from the end, bring up a **Voltage axis** and delete any existing plots by right-clicking, selecting **Delete** and clicking on the legend of the trace (i.e. the text **v(0.5)**). Then record from **axon.v(0)** by selecting a new plot from the **Plot what?** dialog box and editing the text in the text box before clicking **Accept**.

We could create the **IClamps** as we did earlier. However, NEURON provides a convenient way of moving an electrode around. First of all we will delete our old electrode by redeclaring it:

objref stim

Now, select **Tools→Point Processes→Managers→Electrode** from the main menu. Click on the **IClamp** button and change the amplitude to $1nA$, $10ms$ delay and $1ms$ duration. Now change the initial membrane potential to $-70mV$ in the **RunControl** window and simulate for about $50ms$.

[14] $V(x) = E_L + V_0 * e^{-|x-x_0|/\lambda}$, where V_0 is the peak value of the membrane voltage halfway down the axon

Change the location of the electrode, click on **Location** in the electrode window and then click on the relevant part of the schematic diagram of the dendrite. You should see the text just above the window change as you click.

The peak of the voltage curve is smeared more as the distance of the injection site increases. This effect is called *dendritic filtering* as the dendrite changes inputs depending where they are on the dendritic tree. The equations for this behaviour are more complex. They are solutions of the full cable equation and require knowledge of boundary conditions to solve.

12.1 Voltage clamp

A common experimental configuration is the voltage clamp. In a voltage clamp experiment the voltage stays at a fixed value. This is accomplished by measuring continuously the voltage and injecting compensating current if the voltage starts to deviate.

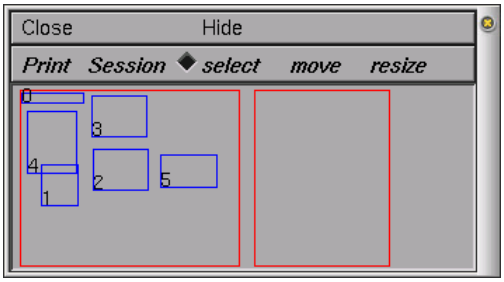
Add an extra electrode to do the voltage clamp (**VClamp**). Locate it at position 0.0. Clamp continuously at $-70mV$ and plot the current of the voltage clamp (**VClamp.i button**). This is the current required to maintain the neuron at $-70mV$.

Move the current injection electrode around and inspect "VClamp.i".

What is the difference in filtering (time to peak and amplitude) compared to the unclamped cable?

13 The Print & File Window Manager

To open the **Print & File Window Manager**, select **Print & File Window Manager** from the **Window** menu on the **Main menu**. The Print & File Window Manager window contains several different buttons, menus and two rectangles.



The leftmost of the two red rectangles in the window represents the entire NEURON display — we will call this the **Manager** rectangle. Each smaller blue rectangle (with a number in it) represents a NEURON window. The second red rectangle represents a sheet of paper — we will call this the **Selection** rectangle. It is used to print selected windows to a file or printer and to save selected windows in a session file. To select one of the windows to print or save, click with right mouse button in one of the smaller blue rectangles in the **Manager** rectangle, and you will see it appear in the **Selection** rectangle. To remove the window from the **Selection** rectangle, simply click with the right mouse button in the small numbered rectangle that you wish to remove. You may also want to move (left mouse button) or resize (middle mouse button) the windows in the **Selection** rectangle.

The **Print** menu gives various options for printing the currently selected windows (i.e., the ones in the **Selection** rectangle) including a **PostScript** printer or a file. If you would like to change the printer that you will print to, you can select the **Select Printer** option in the **Print** menu. This will pop up a window in which you can enter the command to print your file to the new printer. You can also change the layout of the paper to Landscape from Portrait mode (or vice versa) by selecting the **Land/Port** option in the **Print** menu. If you would like to print the PostScript output to a file, then you can select the **PostScript** option under the Print menu[15].

Unfortunately there is no direct way of producing encapsulated PostScript files that are suitable for inclusion in LATEX and other documents. To convert the **PostScript** files produced by NEURON to encapsulated **PostScript**,

[15]If you don't like the border around the figure, you can get rid of it (under UNIX) by creating a file called **~/.nrn.defaults** that contains the single line **scene print border: 0*

a graphics package or command-line utility is required. For example the CONVERT program can be used at the UNIX command line:

convert neuron-picture.ps neuron-picture.eps

14 Saving and retrieving sessions

After creating several graphs, you may want to save the windows you have created (i.e., graphs and panels) to a file so that you can recall them at a later time. NEURON allows you to save either all or selected windows to a *session* by selecting the **Save selected** or **Save all** option of the **Session** menu in the **Print & File Window Manager**. **Save all** will save the position and contents of all NEURON's windows. **Save selected** will save only those windows that are currently selected in the **Selection** rectangle in **Print & File Window Manager**. Either of these options will pop up a window in which you can enter the filename of your save session.

Try to save all the windows into a session file called *practical1.ses*. (You need to type in the *.ses* suffix explicitly - unlike many modern programs, NEURON doesn't add suffixes automatically.)

If we save our session to a file, we can load the session by selecting the **Retrieve** option of the **Session** menu in the **Print & File Window Manager.**

15 Checking your conceptual understanding

Please try to answer the following questions (some involve general neurobiology knowledge and some are more specific to the lectures you have had up to now). If you need to refer to the lecture slides or notes, feel free to do so, but try to think you way through them before that. Discussing the answers among your classmates is encouraged.

- Where do most of a neuron's inputs arrive?
    1. Soma
    2. Dendrites/Spines
    3. Axon terminal
    4. Axon Initial Segment (AIS)
- What is the principal neurotransmitter released by excitatory neurons in the central nervous system (CNS)?
    1. GABA
    2. Glycine
    3. Glutamate
    4. Noradrenaline
    5. Serotonin
    6. Dopamine
- What is the principal neurotransmitter released by inhibitory neurons in the central nervous system (CNS)?
    1. GABA
    2. Glycine
    3. Glutamate
    4. Noradrenaline
    5. Serotonin
    6. Dopamine
- Which part of the neuron releases the neurotransmitter in most cases?

1. Soma
2. Dendrites/Spines
3. Axon terminal
4. AIS

- Which ion causes the release of neurotransmitter at the presynaptic terminal?

1. Na^+
2. Ca^{2+}
3. K^+
4. Cl^-

- Which molecules are responsible for the postsynaptic effects of neurotransmitters?

1. Receptors
2. Ion Pumps
3. Voltage-gated channels
4. Synaptic vesicles

- An AP is generated at the AIS of an excitatory neuron. Put the following events in order of occurrence:

1. Voltage-gated Ca2+ channels open
2. The AP reaches the axon terminal
3. Glutamate binds to postsynaptic AMPA receptors
4. Synaptic vesicles are released
5. Voltage-gated Na+ channels open
6. An Excitatory Post-Synaptic Potential (EPSP) is generated

- The resting membrane potential of a cell is around:

1. -120mV
2. -70mV
3. -55mV
4. 0mV
5. +30mV

- Which force(s) is(are) responsible for passive ion flow across the membrane?

1. Chemical concentration gradient
2. Temperature difference
3. Electrical potential difference
4. All of the above
5. 1 and 2
6. 1 and 3
7. 2 and 3

- What part of the cell does the resistance in the equivalent circuit represent?

1. Lipid membrane

2. Ion pumps
3. Ion channels
4. All of the above

- What part of the cell does the capacitance in the equivalent circuit represent?

1. Cell membrane
2. Ion pumps
3. Ion channels
4. All of the above

- Which ion is primarily responsible for the rapid phase of membrane voltage increase during an action potential (AP)?

1. Na^+
2. Ca^{2+}
3. K^+
4. Cl^-
5. All of the above

- What is responsible for the absolute refractory period of a cell?

1. Opening of K^+ channels
2. Closure of Na^+ channels
3. Inactivation of Na^+ channels
4. Opening of Ca^{2+} channels
5. Opening of Cl^- channels

- What is responsible for maintaining the electrochemical gradients of the ions at physiological levels?

1. Ligand-gated channels
2. Voltage-gated channels
3. Ion pumps
4. Synaptic vesicle endocytosis

- What is true for an ion's reversal potential:

1. On either side of it the flow of the ion has opposite directions
2. At the reversal potential, the net flow of the ion is zero
3. It depends on the ion's intracellular concentration
4. It depends on the ion's extracellular concentration
5. It depends on the temperature
6. All of the above