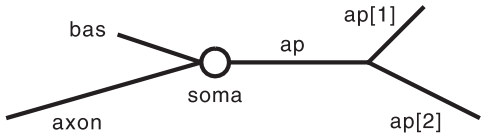
The CellBuilder

The CellBuilder is one of NEURON's latest enhancements. It is a very powerful and convenient graphical tool for constructing and managing models of individual neurons. The CellBuilder is actually a "graphical code generator." In other words, you can use the CellBuilder to enter the specifications for your model cell without having to write any hoc code yourself. When you're satisfied with the specification you have created, the CellBuilder will write the necessary hoc code for you.

Example 1: making a stylized model



This stylized model of a pyramidal cell has a soma, an apical dendritic tree with a primary trunk and two distal branches, a basilar tree that is lumped into a single equivalent cylinder, and an axon. The soma and axon have HH spike currents at full density, and the apical dendrites also have spike currents but only at 10% of the density in the soma and axon. The basilar dendrites are passive.

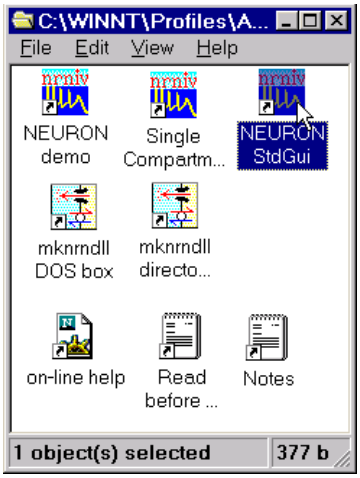
These are the anatomical and biophysical specifications of the model. We'll set the equilibrium potentials for the leak and passive currents so that the resting potential of this cell will be nearly uniform at approximately −65 mV throughout.

	Geometry		
Section	**L**	**diam**	**Biophysics**
`soma`	20 µm	20 µm	`hh`
`ap[0]`	400	2	reduced `hh` *
`ap[1]`	300	1	reduced `hh` *
`ap[2]`	500	1	reduced `hh` *
`bas`	200	3	`pas`
`axon`	800	1	`hh`

*−−`gnabar_hh` and `gkbar_hh` reduced to 10%, `el_hh` = −64 mV
`e_pas` = −65 mV
Throughout the cell `Ra` = 160 Ω cm, `cm` = 1 µf / cm^2

Launch NEURON's standard GUI

We start by running neuron with a hoc file that only brings up the PFWM (Print and File Window Manager) and the NEURON Main Panel. Mac users can just drag and drop the StdGui icon onto the NEURON icon, and MSWin users can double click on the StdGui icon.



UNIX users must enter the command

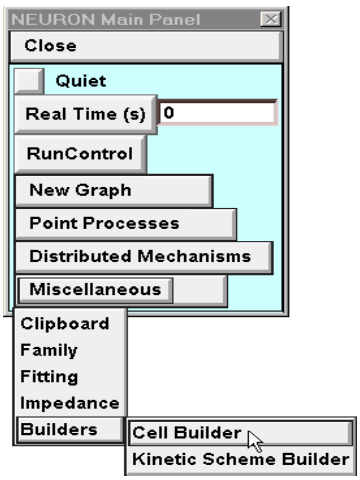
```
nrniv $NEURONHOME/lib/hoc/stdgui.hoc –
```

When the `oc>` prompt appears, enter the command `nrnmainmenu()`

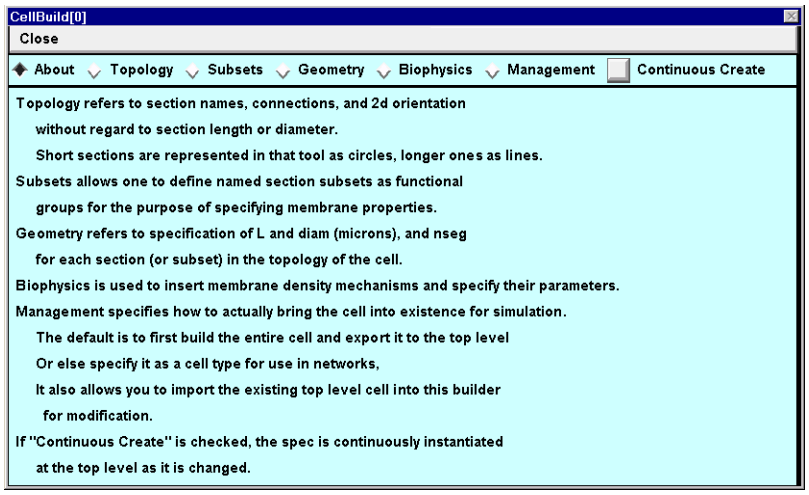
Bring up a CellBuilder

At this point you can use the NEURON Main Panel to bring up a CellBuilder by

`NEURON Main Panel / Builders / Cell Builder`

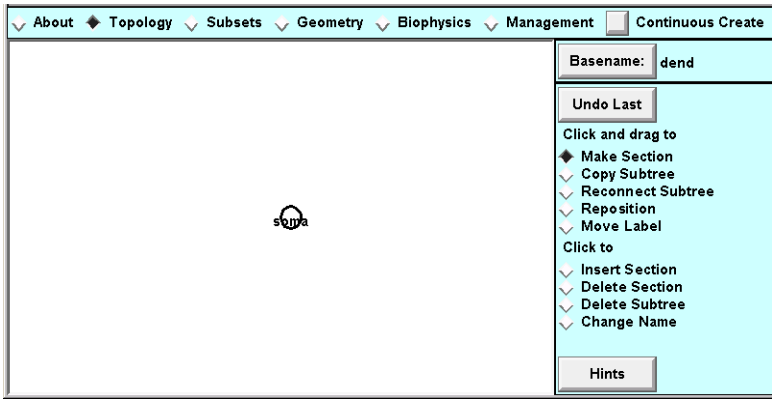


Across the top of the CellBuilder is an array of radio buttons labeled Topology, Subsets, Geometry, Biophysics, and Management. Each of these buttons brings up a different page of the CellBuilder. There is also a checkbox labeled Continuous Create. We're going to see what these controls do.



Topology

The Topology page is for setting up the model's branching pattern. This is where you create new sections and decide how they are connected to each other.



A new CellBuilder always comes up with a section called soma. We want to create new sections.

First we need to change the Basename to ap.

Start by clicking on the Basename button.

This brings up a window with an editable field.
Click inside this field.

Type the new basename and then click on Accept

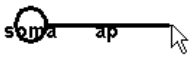
Now click and drag to create the ap branches.

Place cursor near the soma

Click to start new section

Drag to desired length

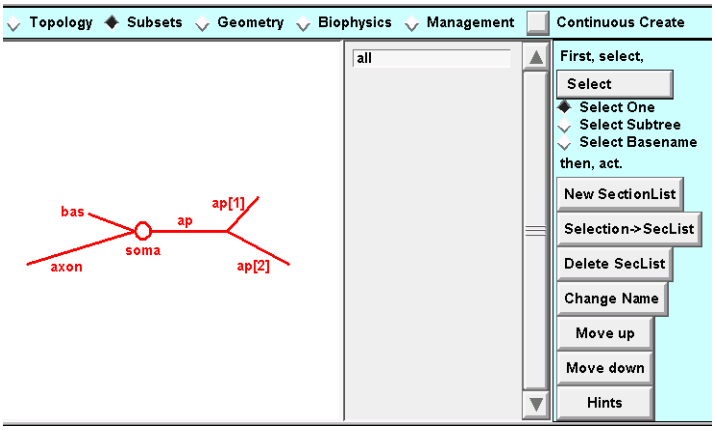
Release mouse button



Next create the basilar branch and the axon in the same way. Use the other buttons as necessary (Undo Last, Delete Section, Change Name etc.). Finally move the labels so they are next to the sections, not on top of them.

Subsets

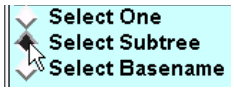
The Subsets page is for grouping sections that share common features into subsets. This will make it easier later when it is time to assign biophysical properties.



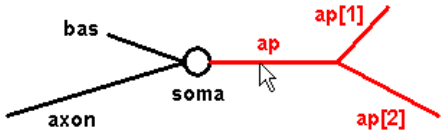
The set "all" contains every section in the model. We'll need it later, so leave it alone.

Each of the apicals has identical active currents, so let's make a subset called apicals.

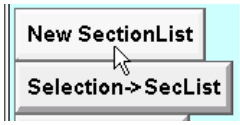
Click Select Subtree



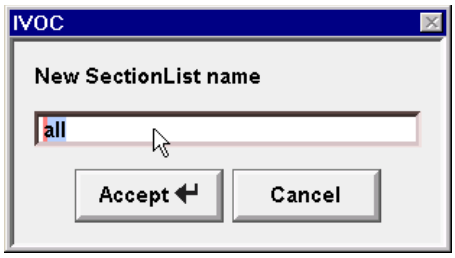
Click on the root of the apical tree . . .



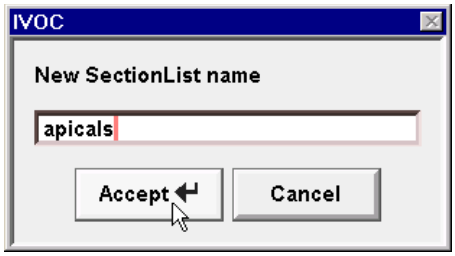
. . . then click on New SectionList



Click in the editing field of the window that pops up . . .



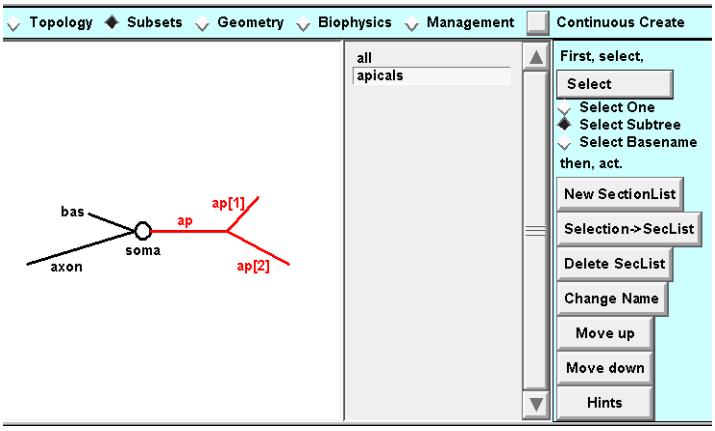
. . . type the name of the new SectionList, and then click on Accept



Before moving on, think: do we need to create any more section lists? The answer is no.

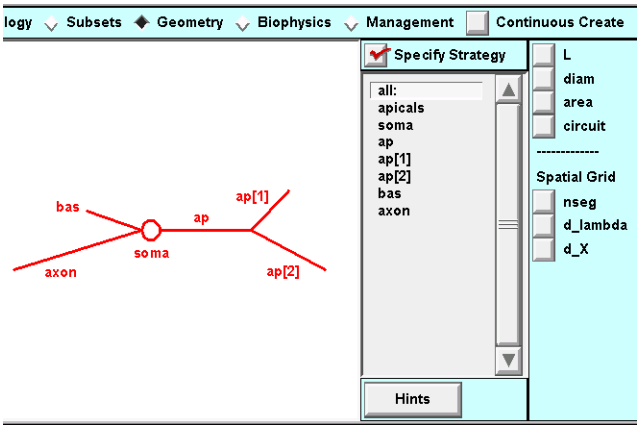
Also ask if we need to change the order of subsets in the center panel of the Subsets page. Sequence is important if a section appears in more than one section list. This is because sequence determines the order in which Geometry and Biophysical properties will be assigned. If a section appears in more than one list, earlier assignments may be overwritten by later assignments. So it's best for the order of section lists to go from general to specific.

The sequence we have here is fine.



Geometry

The Geometry page lets us specify the physical dimensions and segmentation (nseg) of our sections and subsets. The first thing to do here is to set up an efficient strategy for assigning these properties, so make sure the Specify Strategy checkbox is ON (checked). Once we have built our strategy, we toggle the Specify Strategy checkbox OFF and we can then assign specific values.



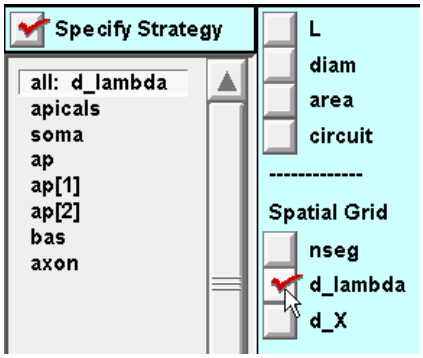
First let's deal with the spatial grid (i.e. discretization or compartmentalization). This is really just a computational issue, not a biological one−−we should instead be focussing on anatomically− and physiologically−relevant subdivisions of the cell, not "how small should we chop to get numerical accuracy and stability." The CellBuilder offers three different ways to make this as painless as possible.

1. The "nseg" button lets you set nseg manually.
2. The "d_X" button lets you specify a physical length, and the CellBuilder will then generate code that automatically sets nseg for each section so that its segments are no longer than d_X.
3. The "d_lambda" button is probably the best all−round choice. This lets you specify a maximum length for each segment, expressed as a fraction of the AC length constant at 100 Hz for a cylindrical cable with the same diameter, Ra, and cm. The resulting grid is fine enough for most purposes; if you need even more accuracy in space, just make d_lambda smaller.

Whatever you choose, the CellBuilder will always set nseg to an odd number. This means that each section is guaranteed to have a node exactly halfway down its length.

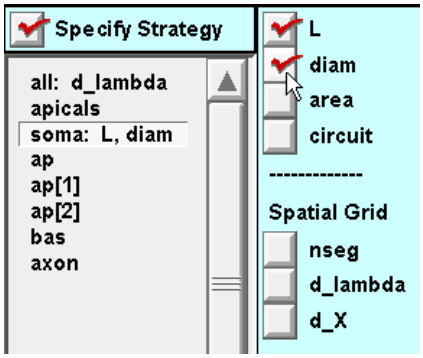
Let's use d_lambda for every section in this model.

Click on the "all" subset, and then click on its d_lambda checkbox.

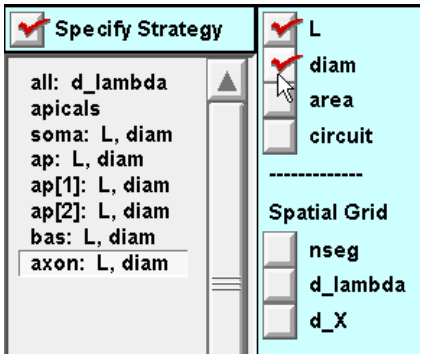


Each section has different dimensions, so we need to specify L and diam individually for each section. In the current version of the CellBuilder, this means you'll have to click on L and diam for each individual section.

Each section needs its own L and diam, from soma . . .



. . . to axon.

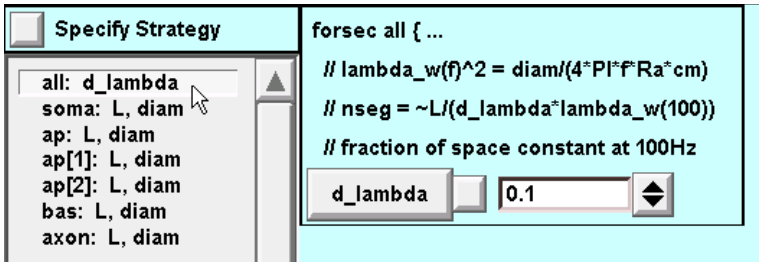


> **An important aside:** when planning your strategy, keep the sequence of subsets and sections in mind. If the order isn't right, or if you need more subsets, then you should go back to the Subsets page and make the necessary changes. We're OK here.

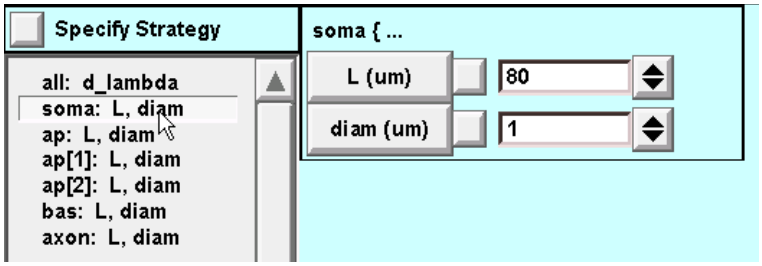
Once the strategy has been specified, click on the Specify Strategy box to turn it OFF. Now we're ready to enter actual values for d_lambda, L, and diam.

First let's take care of the spatial grid.

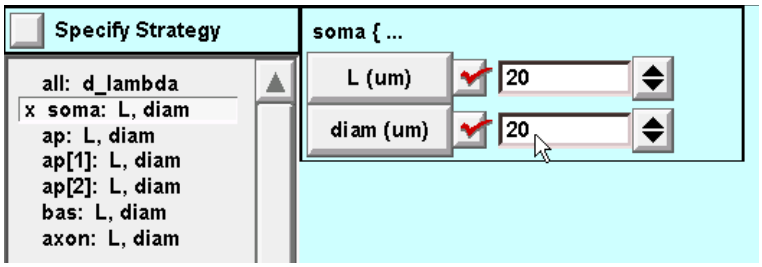
The default value of d_lambda is 0.1, that is, one tenth of a length constant at 100 Hz. That turns out to be short enough for most purposes. We can always come back later and try a different value if we like.



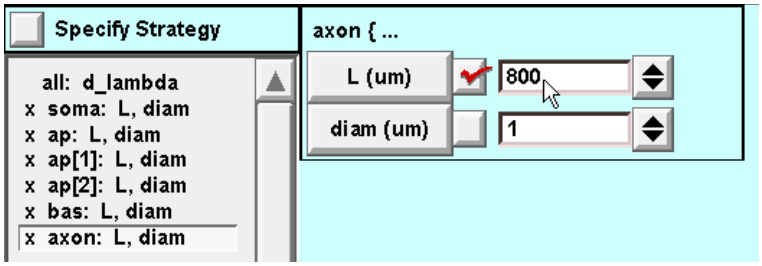
Next we change the soma's L and diam from their defaults . . .



. . . to what we want

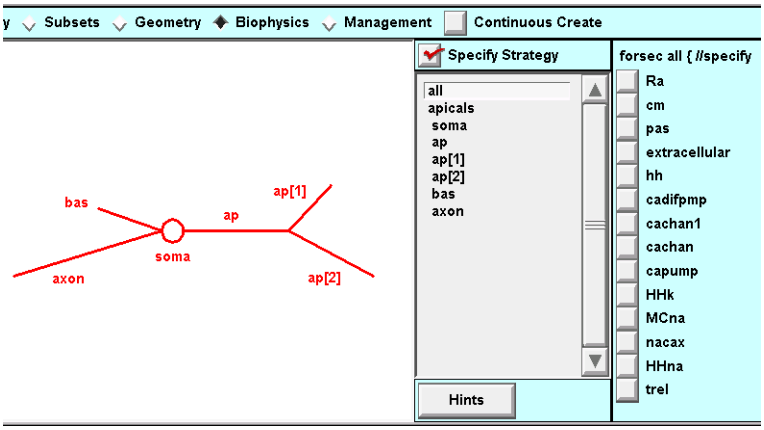


After doing this for all the other sections, we will see



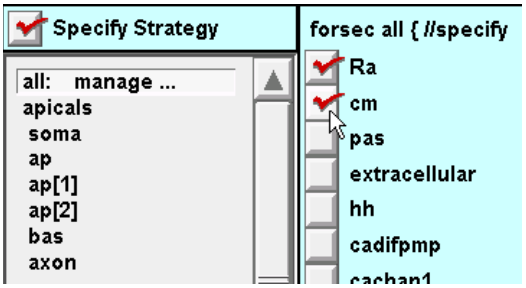
Biophysics

The Biophysics page is where you insert all of those biophysical properties (e.g. Ra, cm, ion channels, buffers, pumps) into subsets and individual sections. Just like you did in the Geometry page, you first set up your strategy, then you review and adjust parameter values. So make sure that you start with Specify Strategy ON.

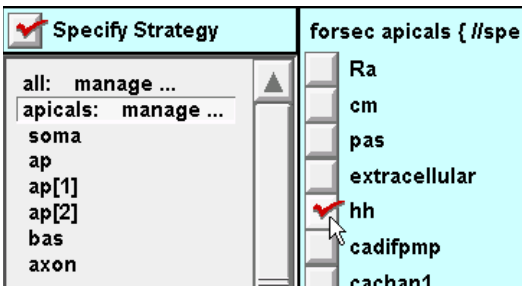


Once again, I must point out that the specification will be executed in the same order as you see in this list of subsets and sections. If the order isn't right, or if you need more subsets, then go back to the Subsets page and make the necessary changes−−and then ***check both the Geometry and Biophysics pages*** to make sure you didn't break anything.

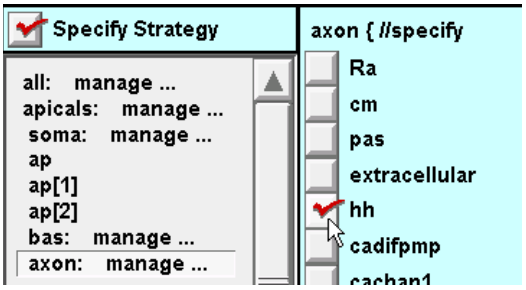
Ra and cm are uniform in this particular model, so click on the all subset and then click on Ra and cm.



The apicals have the hh mechanism, so click on apicals and then click on hh

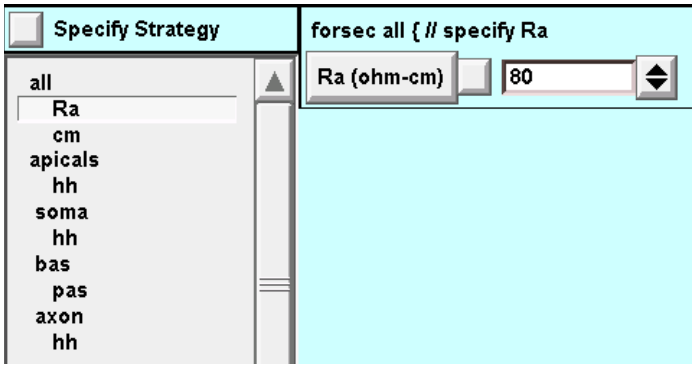


The soma and axon also have hh, while the basilars have pas

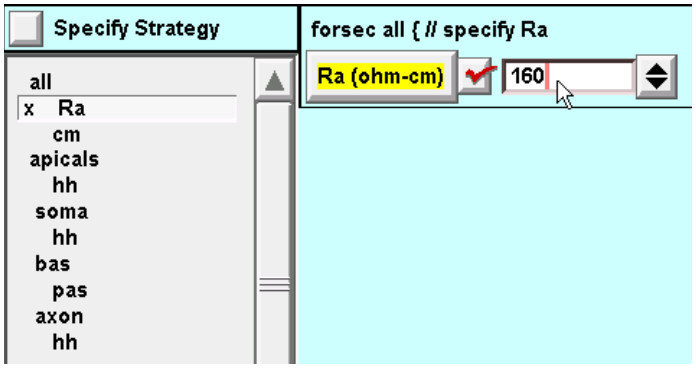


After our strategy is complete, we can enter the numeric values of the parameters, so toggle Specify Strategy OFF.

The *all* subset: change Ra from its default . . .

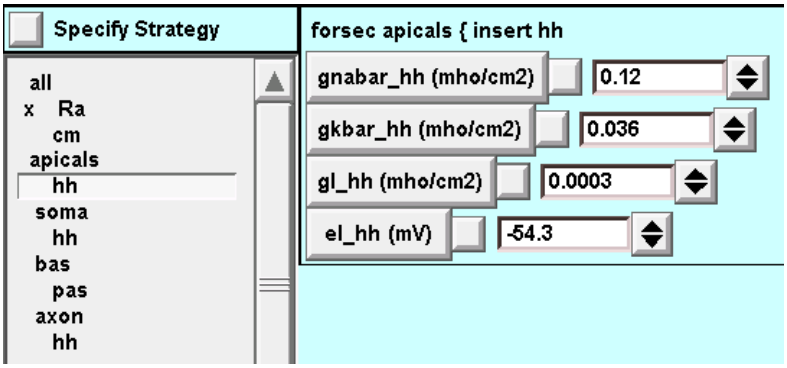


. . . to its desired value

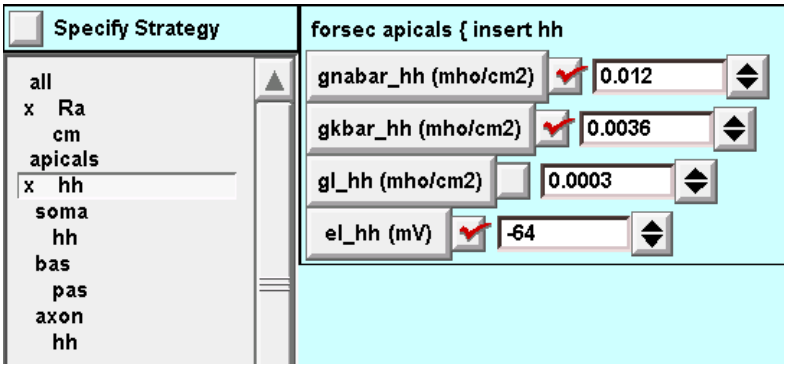


The default value of cm is OK (1 µf / cm^2), so leave it alone.

The *apicals* subset: change gNabar_hh, gKbar_hh, and el_hh from their defaults . . .

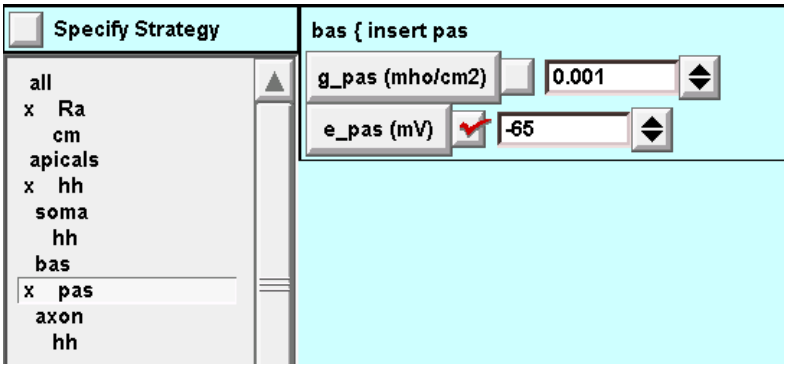


. . . to what we need



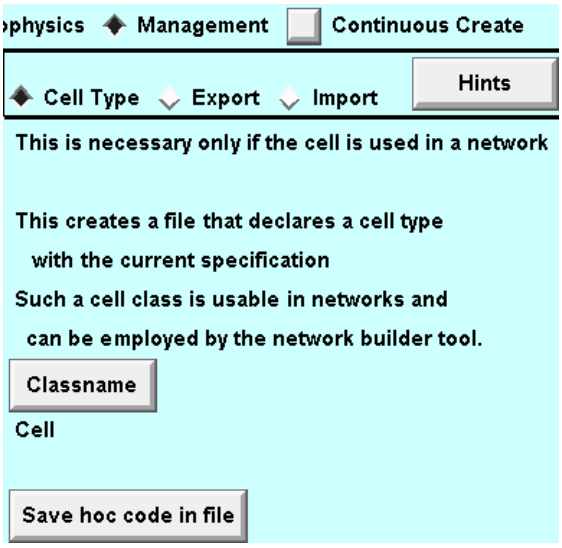
The *soma* and *axon* sections: use the default hh values (no change needed).

The *bas* section: change e_pas from its default of −70 mV to the desired value (shown here).



Management

Finally we get to the Management page. This is for importing and exporting models in a variety of formats. For now let's concentrate on how to get models OUT of the CellBuilder; we'll cover importing in the next example.

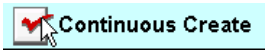


The button labeled "Save hoc code in file" does just what it says−−but what kind of hoc code is written depends on whether you have selected "Cell Type" or "Export". Choose Cell Type if you want to define a new cell type (i.e. a new class of object) that you can use in networks. "Save hoc code in file" will then write a file that contains a template that defines your new cell type. Export is for when all you want is to save the basic cell specification as a model of an individual cell.

Expert tips

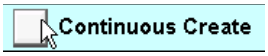
1. Save the CellBuilder to a session file. Do this even if you use Cell Type or Export. Saving the CellBuilder to a session file saves the current state of *all* information in the CellBuilder. You can then build a menagerie of "clones" of model cells through a process of successive revision. If you ever want to modify one of these model cells, just retrieve the session file of its CellBuilder and start editing. If instead you had only saved models with Cell Type or Export, you'll have to start building from scratch because the files written by Cell Type and Export don't contain all of the information needed to recreate the CellBuilder.
2. Here's another reason for saving the CellBuilder to a session file. The hoc code that the CellBuilder generates may not work if a section or subset in your model has a name that conflicts with a keyword or variable that already exists in NEURON's interpreter. If this happens, just change the offending name in the CellBuilder.

Generally the best way to work with the CellBuilder, at least during the development phase of a model, is to use Continuous Create. Turning Continuous Create ON makes the CellBuilder send hoc code to NEURON's interpreter without bothering to write a hoc file.



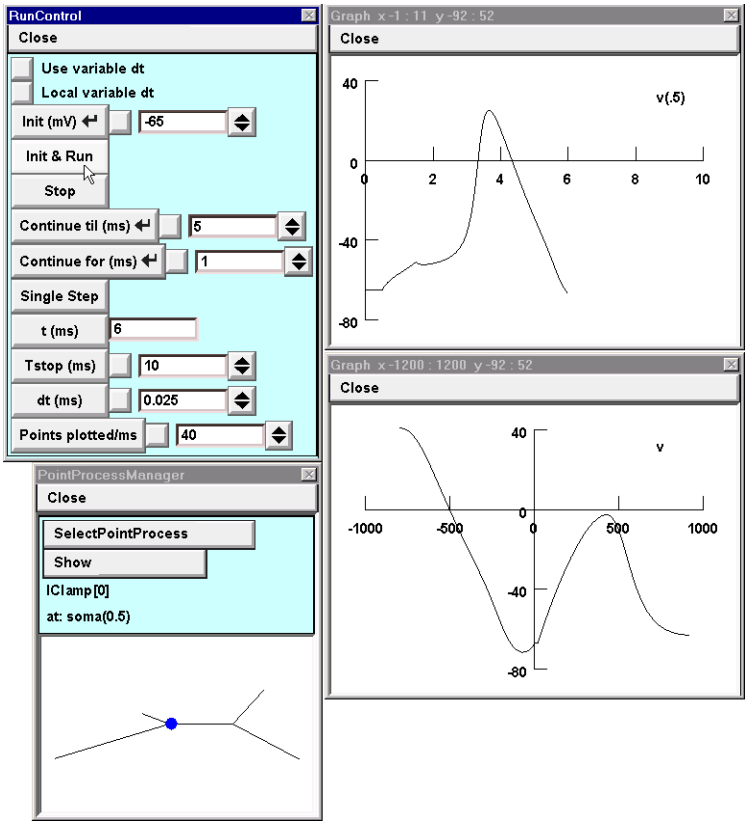
Any changes made to the model while Continuous Create is ON will automatically be echoed to NEURON's interpreter. This lets you immediately test the model you just created or edited.

Automatic updates can bog things down when you are editing a large model on a slow machine. If this happens, just turn Continuous Create OFF.



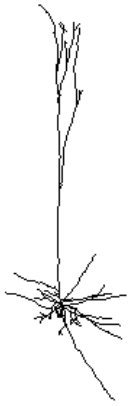
Then make whatever changes you want, and when you're done just toggle it ON and then OFF again.

Now that we've toggled Continuous Create ON and OFF, there is a representation of the model at the top level of the interpreter, ready to be instrumented and exercised. Let's see how it responds to a 0.6 nA x 1.0 ms current pulse applied to the soma.



Example 2: managing a model based on morphometric data

Now that detailed morphometric data is becoming increasingly available, people have to deal with the problem of assigning biophysical properties to models with very complex architectures. To see how the CellBuilder can help solve this problem, we turn to the pyramidal cell model that comes with NEURON's demonstration program.



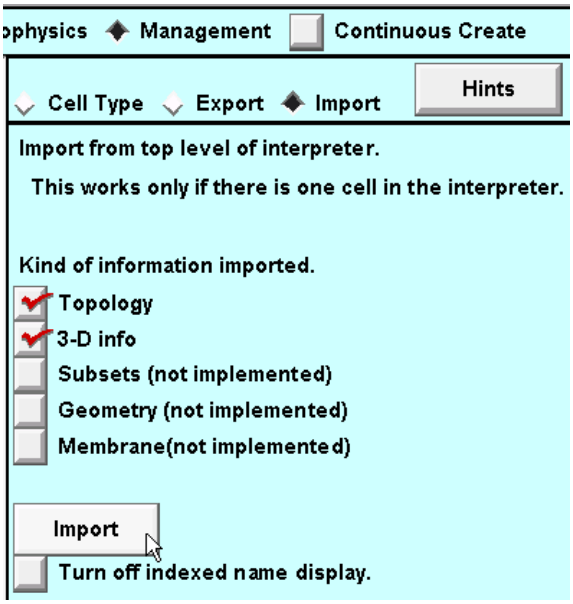
We'll use the CellBuilder to assign these biophysical properties to this model.

Section	Biophysics	Notes
soma	hh	—
axon	hh	—
basilar dendrites	pas	e_pas = −65 mV g_pas = 3.3333 S / cm^2 (Rm = 30,000 Ω cm^2)
apical dendrites	pas	same as basilar dendrites
	hh	gnabar_hh and gkbar_hh reduced to 10% gl_hh = 0 (already has g_pas)

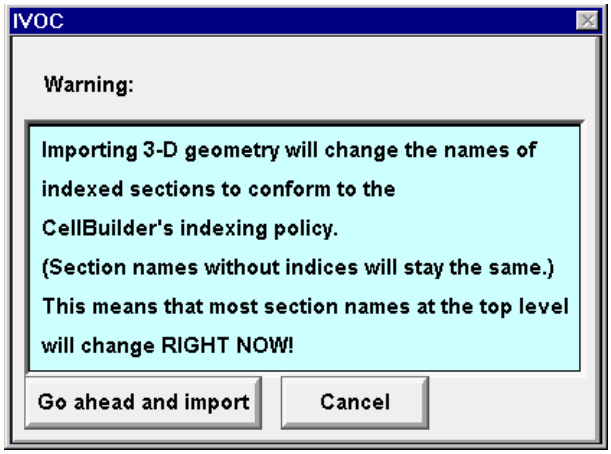
Throughout the cell Ra = 160 Ω cm, cm = 1 µf / cm^2

Get the model into the CellBuilder

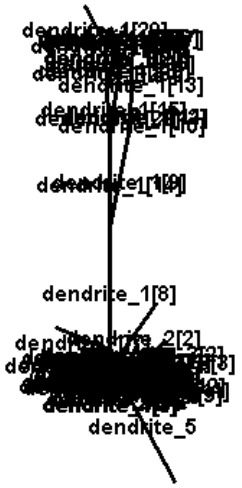
1. Run NEURON's demo program
2. Select Pyramidal from the NEURON demonstrations window
3. Bring up the CellBuilder. This is a new CellBuilder, so it should only show a soma.
4. Select the Management page, then select the Import radio button.
5. Click on the Import button near the bottom of the CellBuilder.



6. You will be warned that Import will discard whatever information is already in the CellBuilder, replacing it with a copy of information about the model cell that already exists in the interpreter. Give it the go ahead.



Now the CellBuilder contains the pyramidal cell's topology and geometry. Looks pretty messy.



Turn off indexed name display to clean things up a bit.

Much clearer.

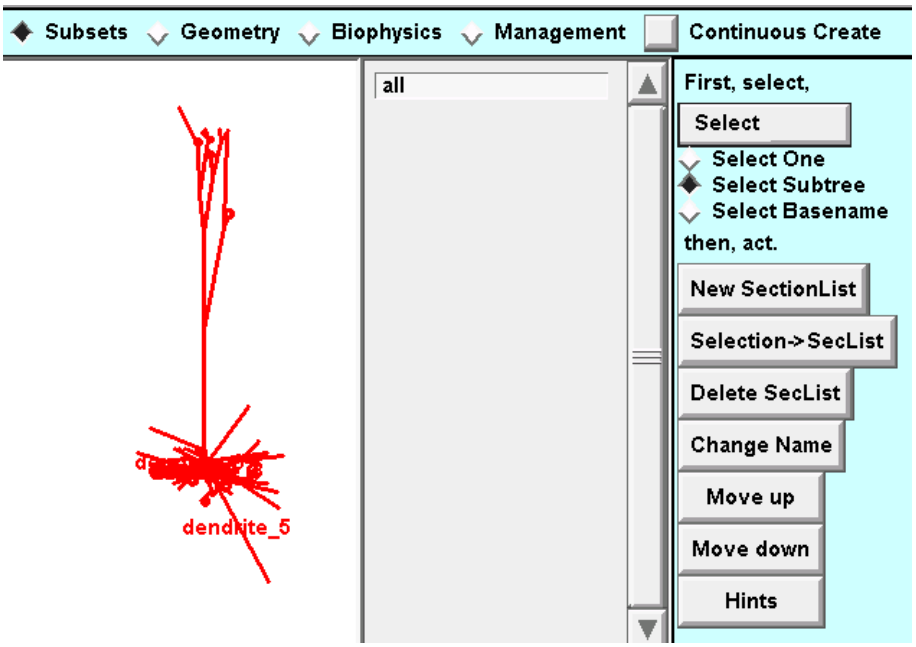
Now save the CellBuilder to a session file (call it rawpyr.ses) and exit the demo program, then restart NEURON using stdgui.hoc and retrieve rawpyr.ses. This reduces the chance of encountering name conflicts.

What makes the CellBuilder particularly convenient for dealing with detailed models like this? In a word: Subsets.

Let's think about our strategy for a moment, and what subsets we'll need (look back at the table of desired biophysical properties). We'll need an "apicals" subset. For the sake of clarity we will also want to have an "axon" subset, because the anatomical axon ended up with the name "dendrite_5" when it was imported into the CellBuilder. Making a subset called "axon" that

contains dendrite_5 will allow us to use an anatomically meaningful name when we need to refer to the axon.

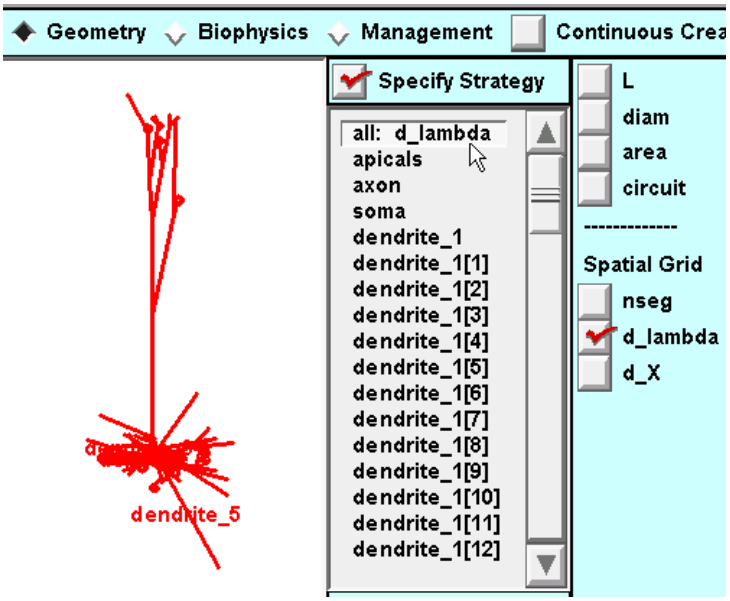
"Creating these subsets is left as an exercise for the reader." Here's what the Subsets page looks like to start. Have at it.



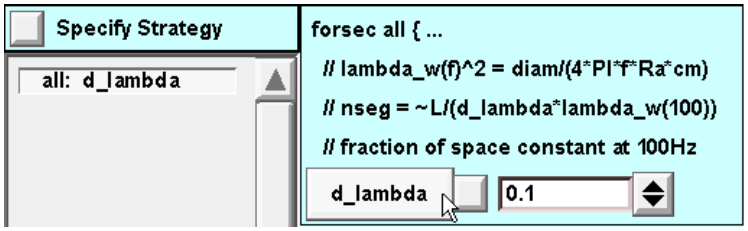
Hints: Select Subtree; if necessary use "shift click" or "click and drag" to select several sections one at a time; it may be helpful to zoom in.

Geometry

First make sure that Specify Strategy is ON. We want to grid the cell using the d_lambda criterion, so choose d_lambda for the all subset.



Then toggle Specify Strategy OFF and verify that d_lambda is to our liking.

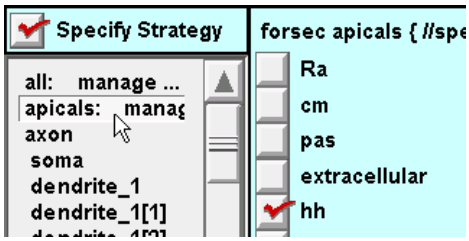


Biophysics

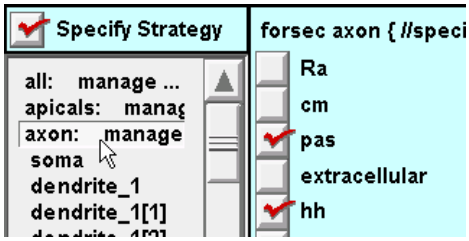
Again make sure that Specify Strategy is ON.

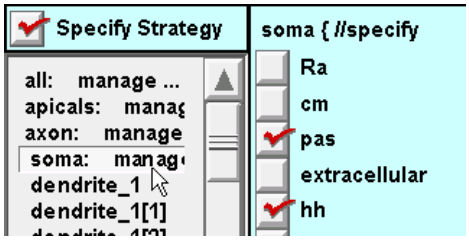
The *all* subset: set Ra and cm, and insert pas

The *apicals* subset: insert hh.
Note: the apicals already have pas because the *all* subset specification will be executed first.



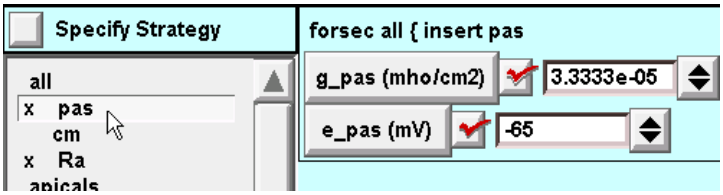
The *axon* subset and soma section: insert hh.
Also insert pas because we will need to set g_pas = 0 in the axon and soma (remember that the *all* subset inserted pas with a nonzero g_pas). This crude hack achieves expedience at the cost of clarity−−a poor trade at best. It would have been preferable to define subsets that would make this unnecessary, i.e. instead of inserting pas into all sections, do this with a subset called "haspas" that contains all sections except the axon and soma. "This is left as an exercise for the reader."

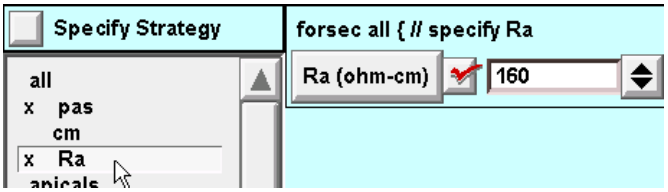




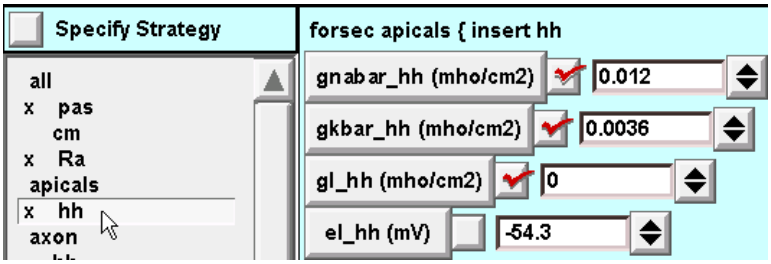
Toggle Specify Strategy OFF, and we are ready to enter the desired parameter values.

The *all* subset: assign values to g_pas, e_pas, and Ra

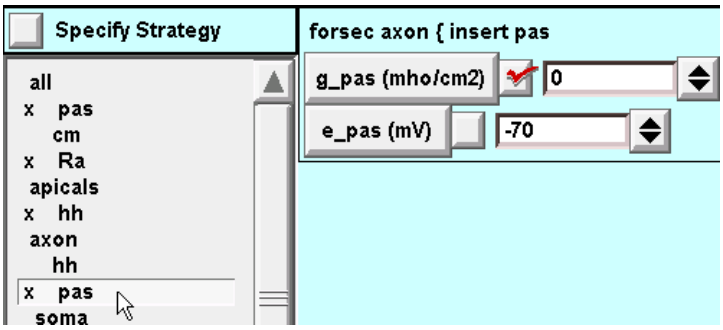




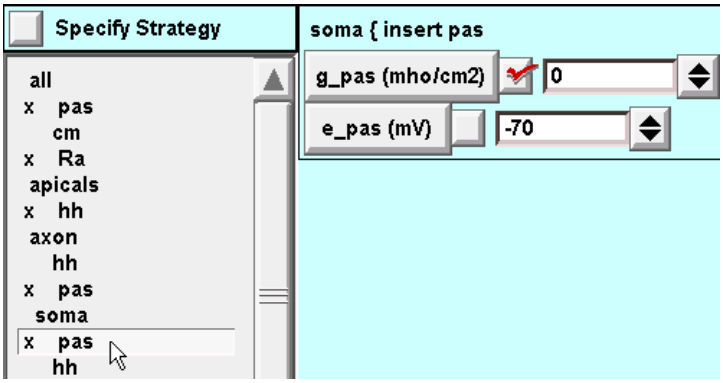
The *apicals* subset: set desired hh values.



The *axon* subset: default hh is fine.
Execution of the *all* subset specification inserted pas, so must set g_pas = 0 for this subset.



The soma section: same as for axon.



When finished, save the configured CellBuilder to pyrfin.ses

To test the model cell, toggle Continuous Create ON and OFF so there will be a representation of the model at the top level of the interpreter. Here is the response of this model to a 2.0 nA x 1.0 ms current pulse applied to the soma.