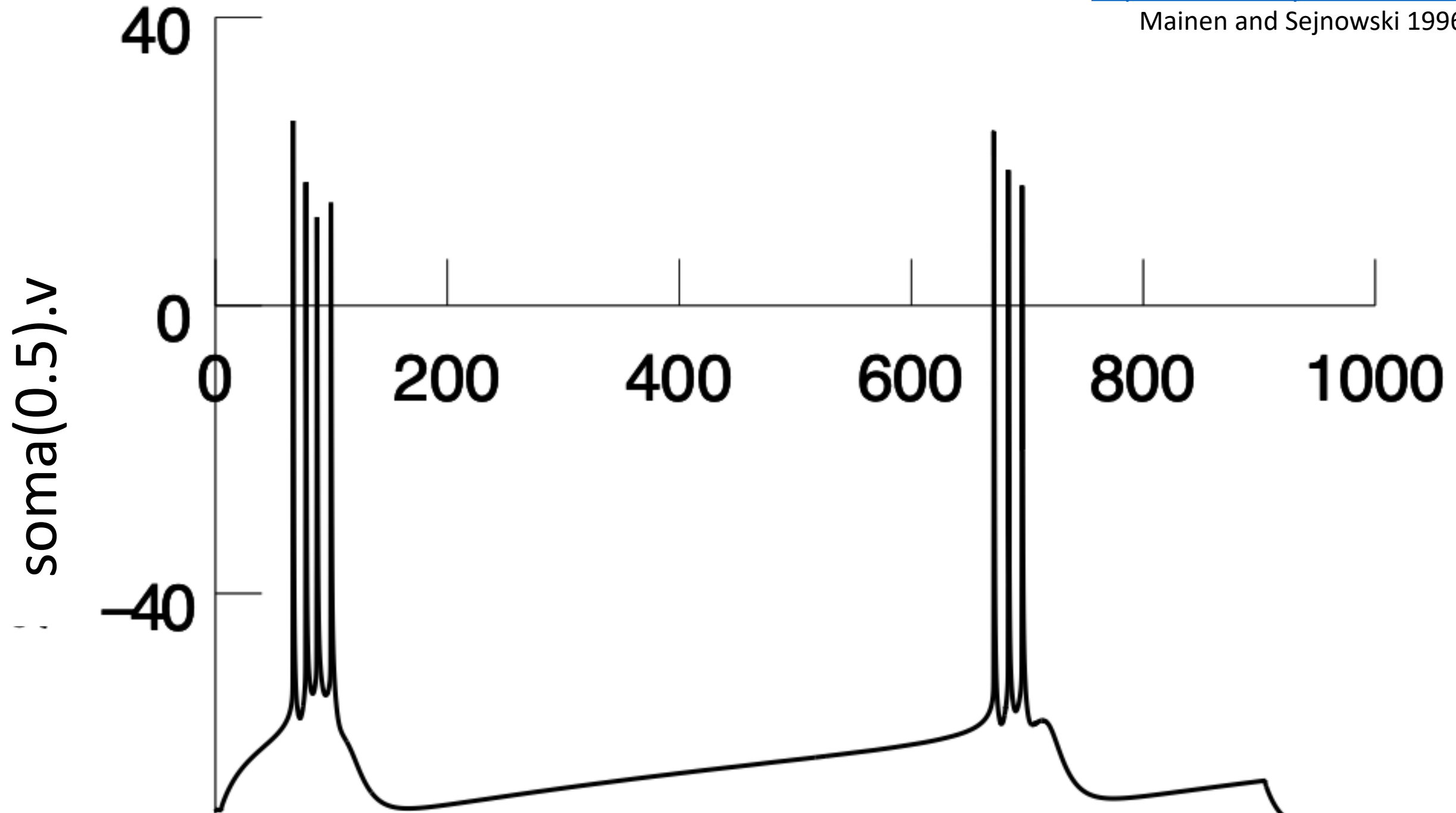
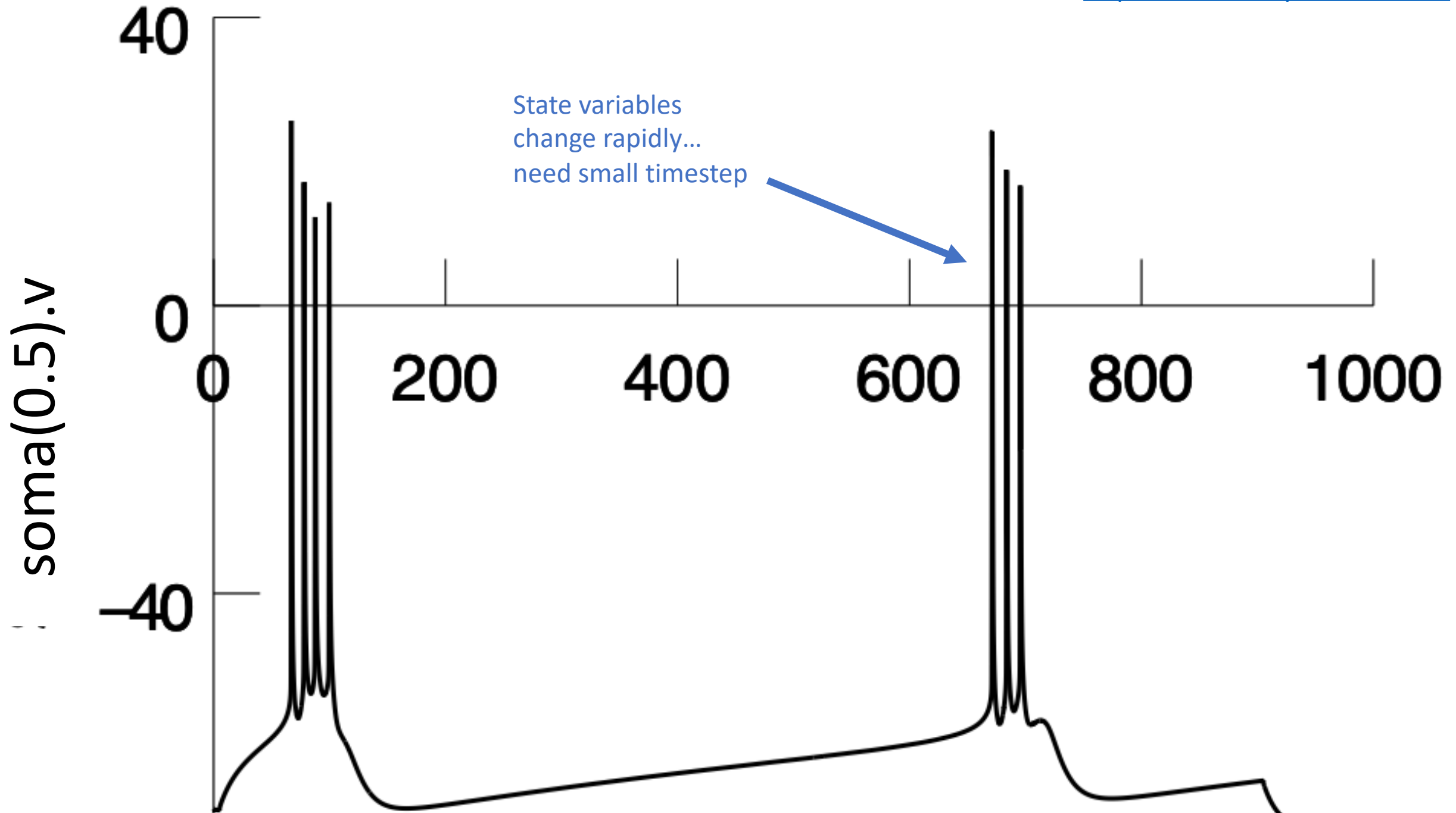
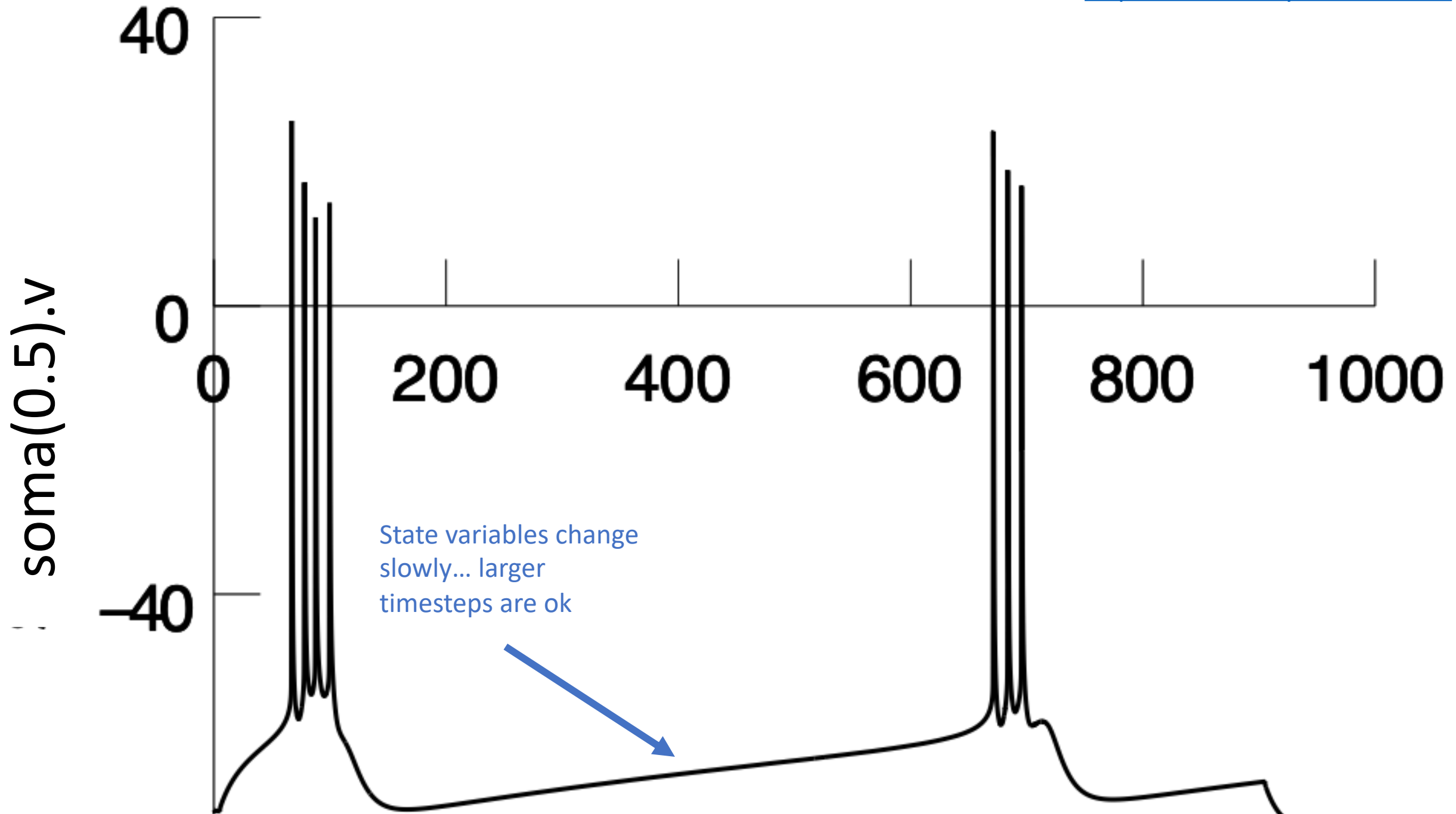


Numerical Methods:

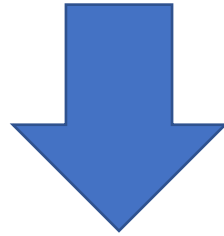
Adaptive Integration



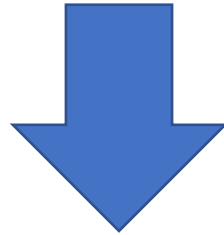




We'd like to use big timesteps when things are changing slowly, and small timesteps when things are changing quickly.

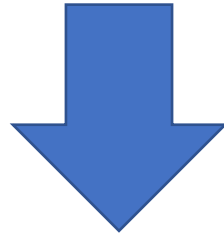


But we don't know in advance when either will happen, so what do we do?

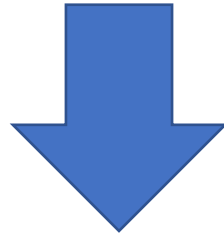


Math!

We'd like to use big timesteps when things are changing slowly, and small timesteps when things are changing quickly.



But we don't know in advance when either will happen, so what do we do?



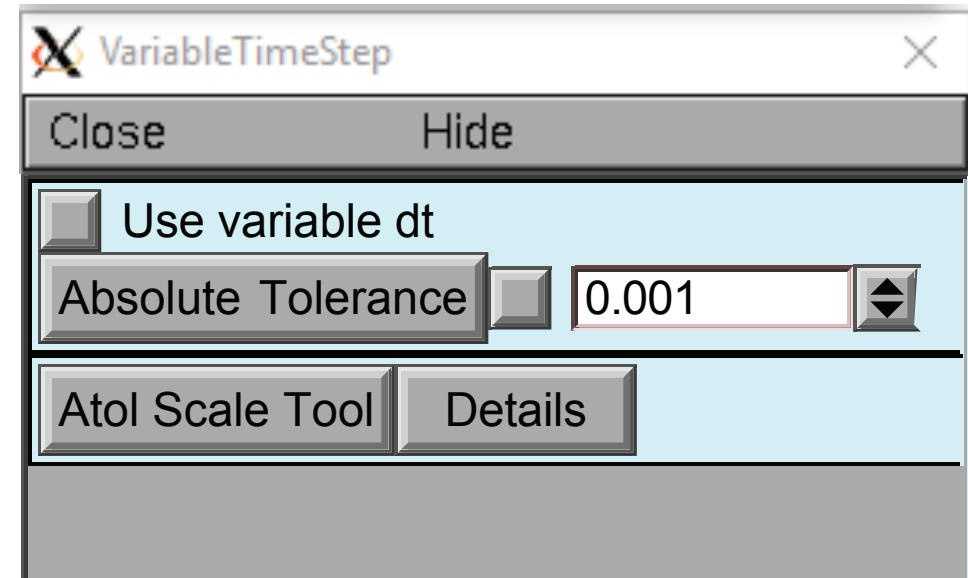
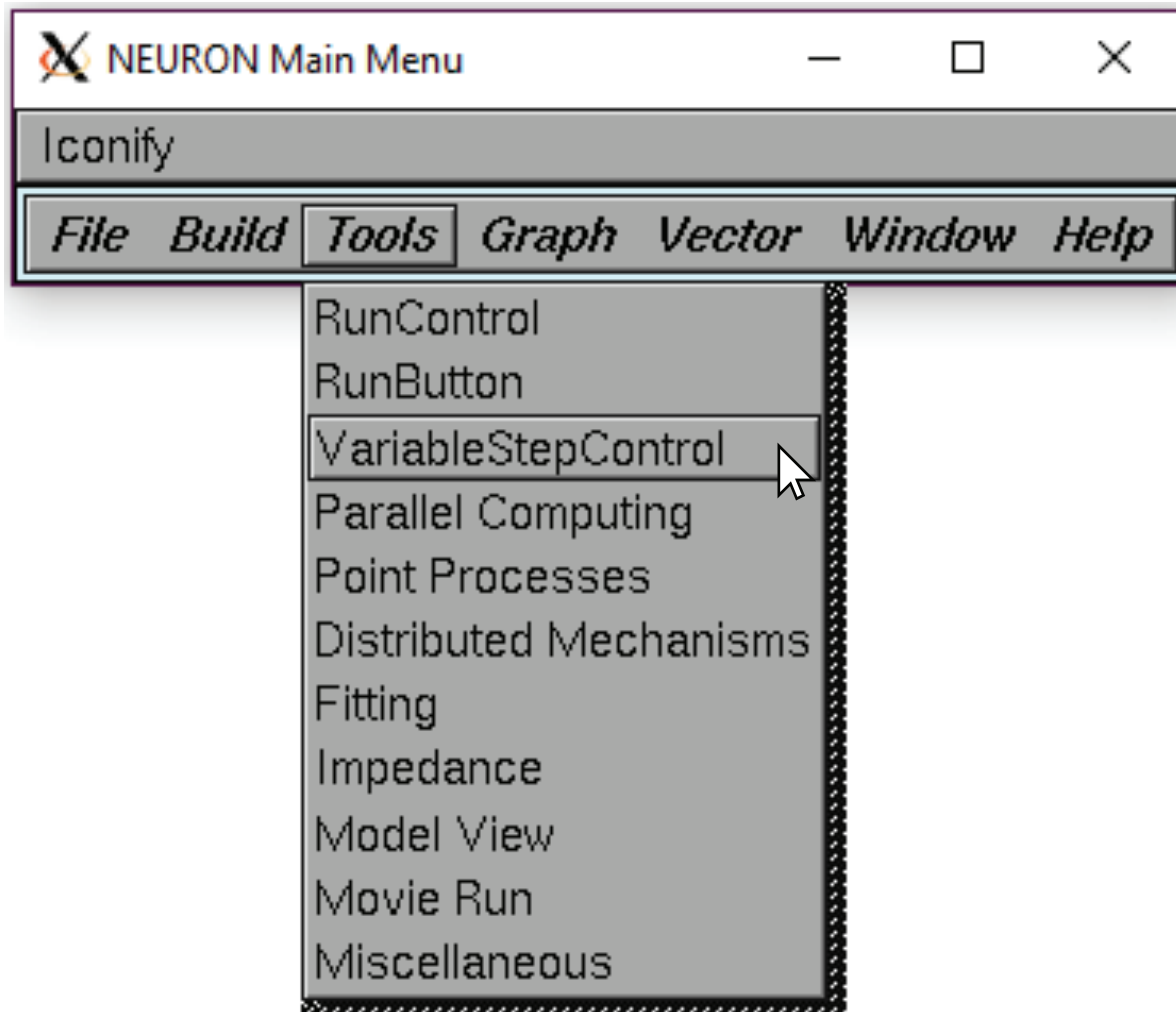
Variable step integration

Enabling adaptive integration

```
h.cvode_active(True)
```

or

```
h.CVode().active(True)
```

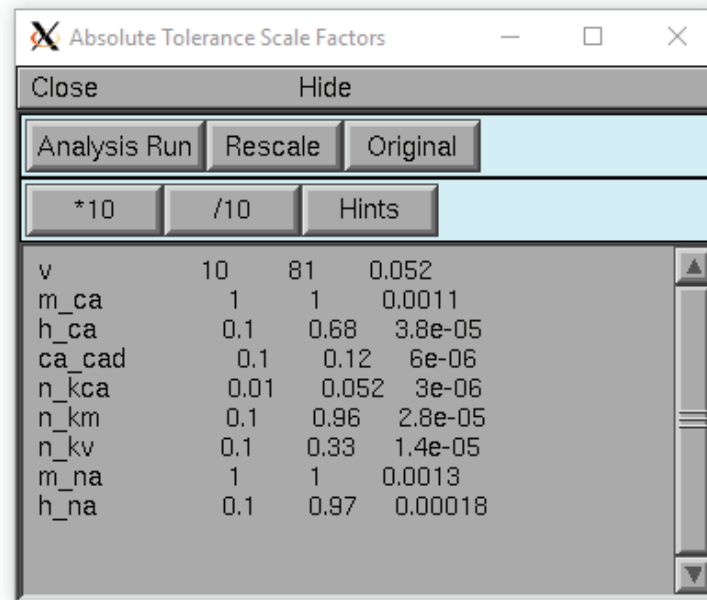
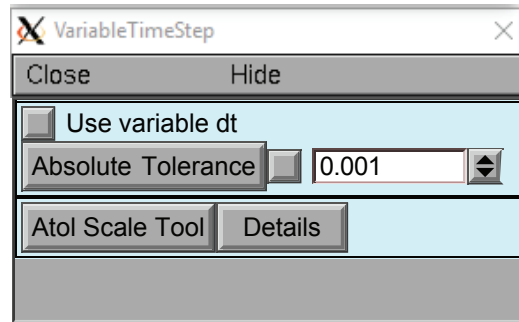


h.cvode active is defined in stdrun.hoc which is loaded automatically whenever the gui is imported.

This is a composite image, not a screenshot to allow combining the window decoration and vector-based window contents.

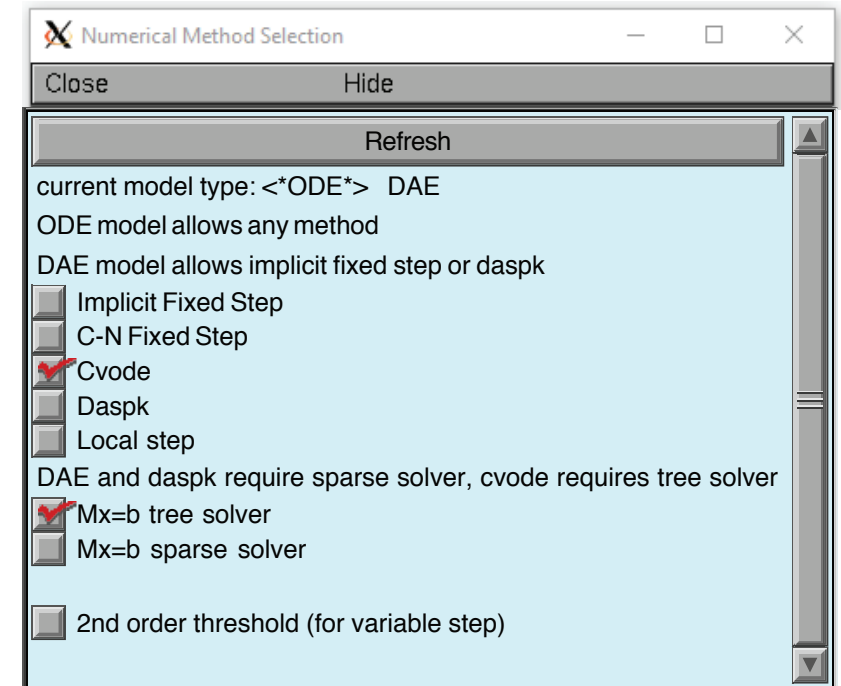
Options:
per state
variable
tolerance,
integration
methods

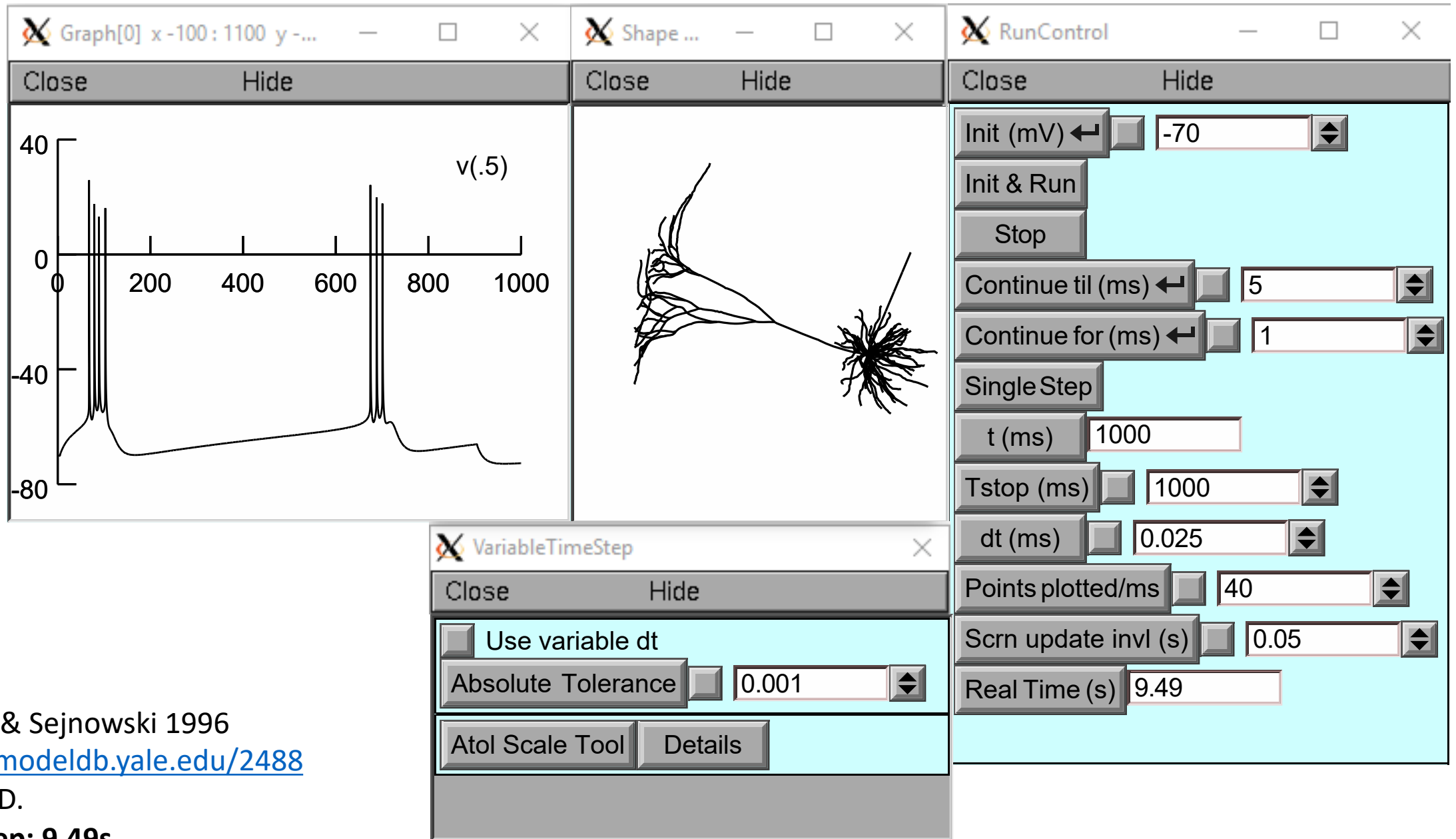
`h.CVode().atol(0.001)`



The Absolute Tolerance Scale Factors dialog box has a title bar with close, maximize, and minimize buttons. Below the title bar are 'Close' and 'Hide' buttons. There are three buttons: 'Analysis Run', 'Rescale', and 'Original'. Below these are three buttons: '*10', '/10', and 'Hints'. The main area is a table with 4 columns and 9 rows.

v	10	81	0.052
m_ca	1	1	0.0011
h_ca	0.1	0.68	3.8e-05
ca_cad	0.1	0.12	6e-06
n_kca	0.01	0.052	3e-06
n_km	0.1	0.96	2.8e-05
n_kv	0.1	0.33	1.4e-05
m_na	1	1	0.0013
h_na	0.1	0.97	0.00018

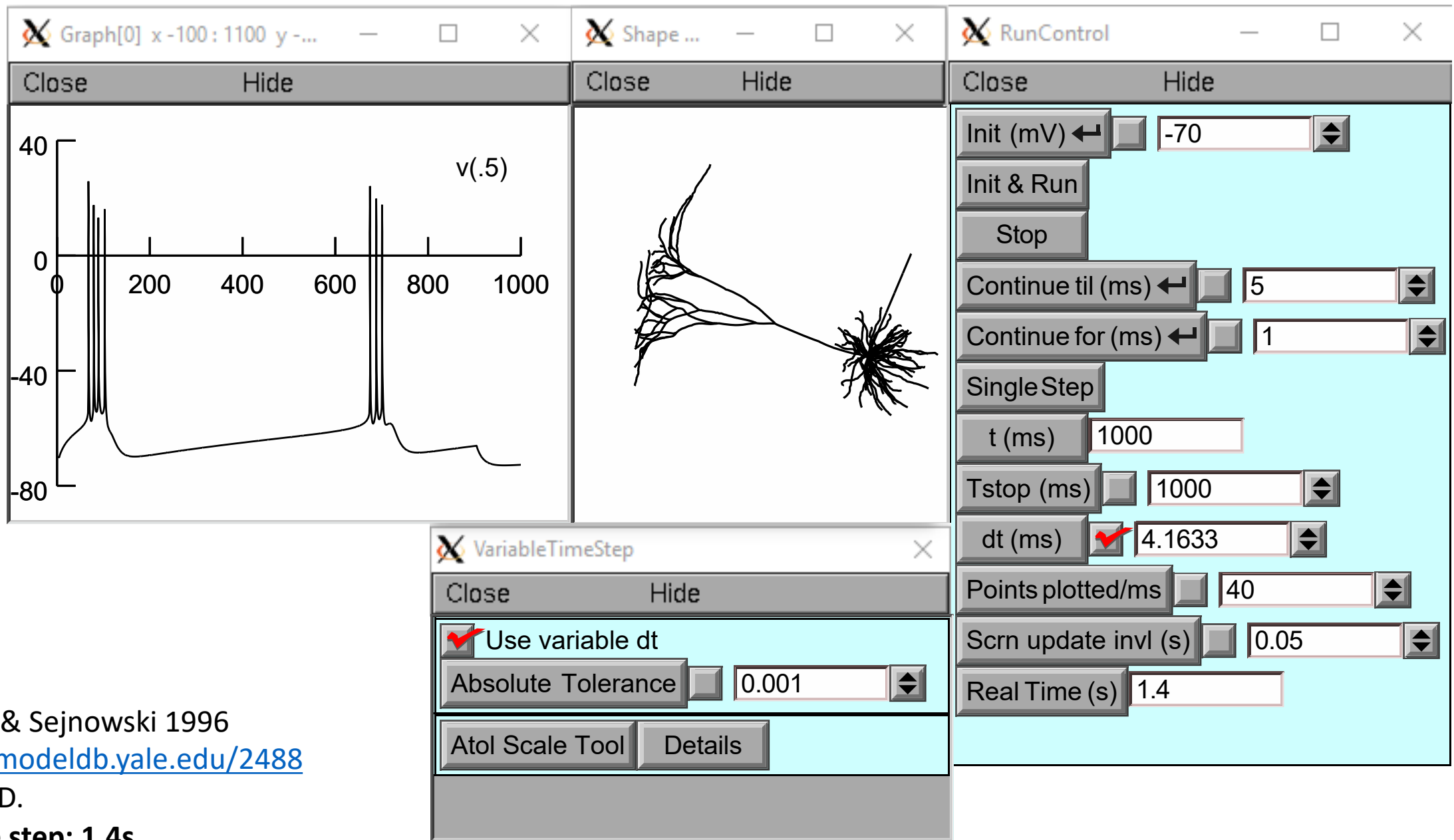




Mainen & Sejnowski 1996
<https://modeldb.yale.edu/2488>

Figure 1D.

Fixed step: 9.49s



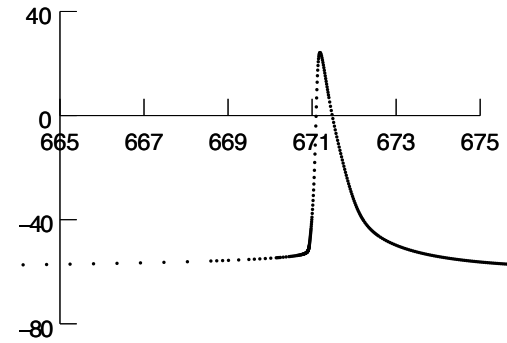
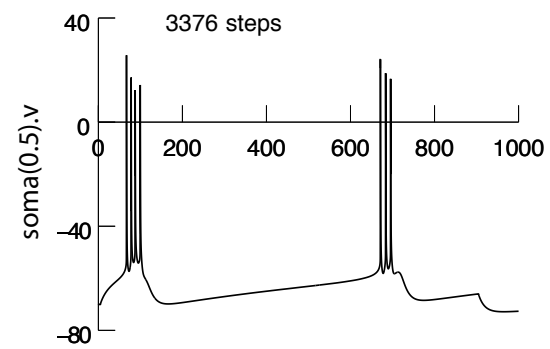
Mainen & Sejnowski 1996
<https://modeldb.yale.edu/2488>

Figure 1D.

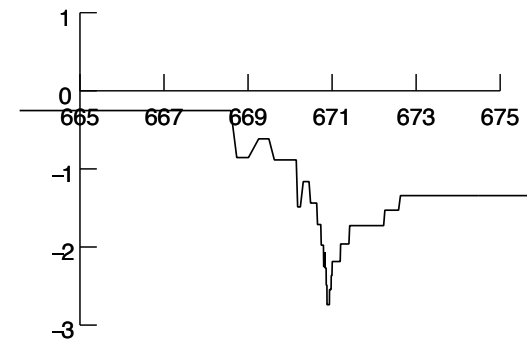
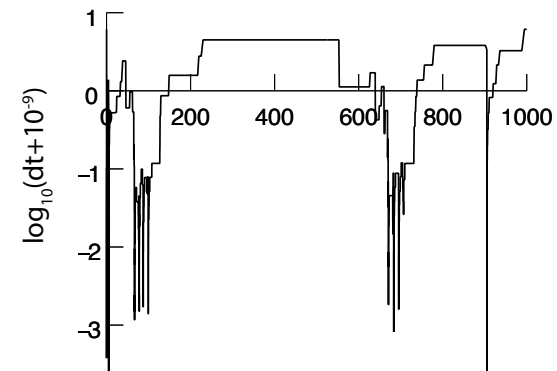
Variable step: 1.4s

NEURON 7.5 on a 3.4 GHz i7-4770 with 24 GB RAM in WSL • composite image

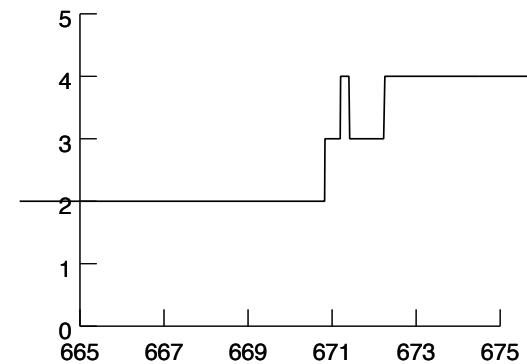
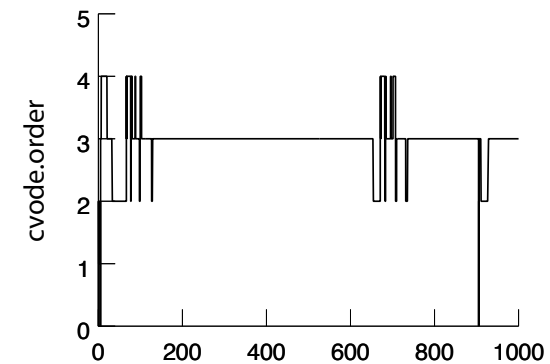
Membrane
Potential



Timestep
Size



Order of
Integration Method



Running simulations: improving accuracy

Increase time resolution (by reducing time steps) via, e.g.

```
h.dt = 0.01 * ms
```

Enable variable step (allows error control):

```
h.CNode().active(True)
```

Set the absolute tolerance to e.g. 10^{-5} :

```
h.CNode().atol(1e-5)
```

Increase spatial resolution by e.g. a factor of 3 everywhere:

```
for sec in h.allsec(): sec.nseg *= 3
```