Fall 2017 - Mini project on alternating spinal cord oscillator
Due noon, Thursday, Nov. 9

Write a simulation in Matlab to reproduce the behavior of the coupled alternating oscillator discussed in the Wang and Rinzel (1992) paper (see below for hints on interpreting their notation). The simulation should be done in Matlab. The simulation will be very similar to a HH simulation, but without even a K+ current (just a sodium-like current). There are 4 differential equations total, 2 for each of two identical interconnected neurons. Each neuron has one equation for voltage (determined by a leak current, an m^3h current, and an inhibitory synaptic current coming from the other neuron, that is sigmoidal function s of the other neuron's voltage), and a second equation representing the inactivation gate of the Na-like current. The reason you don't have a differential equation for the m variable is that it is assumed to update so quickly compared to the other variables that it updates instantaneously, so you can just represent it as an instantaneous function of voltage (which is given in the paper).

Regarding initial conditions, you can look at Figure 1B to see the phase plane (V vs. h). A recommended approach is to pick an initial condition somewhere in the middle of the little enclosed (limit cycle) region. But be careful not to set the two cells in exactly the same initial condition because they can get locked in a tie!! Another approach: choose voltage values for the two cells from by picking a time in Figure 1A, and then picking the h's for each cell accordingly (i.e. tied to the two voltage values).

To turn in:

1. A PDF report with all required figures. Include captions, and detail the parameters used.

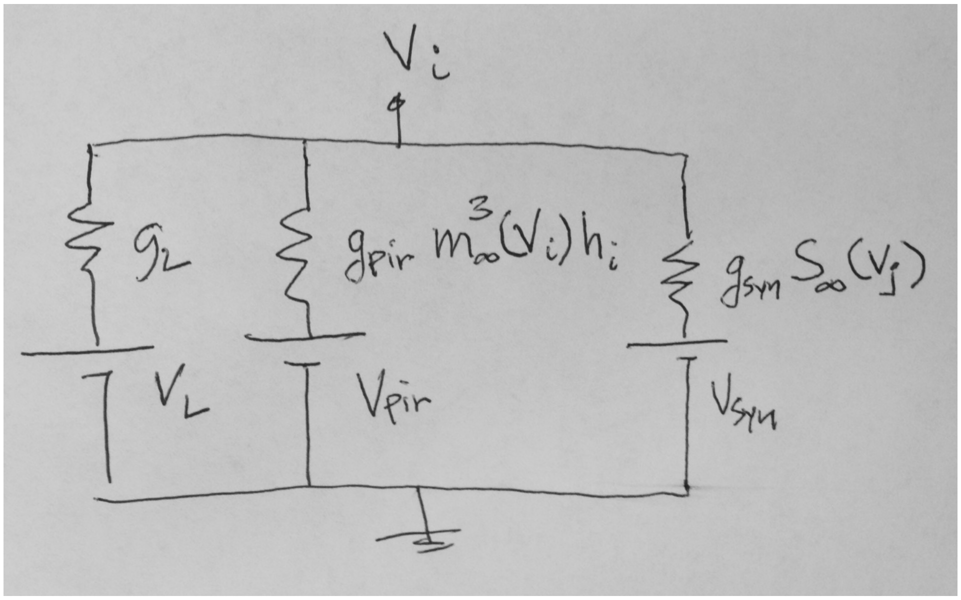
2. Reproduce the voltage traces shown in Figure 1A. Show the voltages of the two neurons on separate plots, or if you use the same plot, make sure they appear in different colors. Include a caption with your figure stating what the figure shows.

3. Create a plot like the one in Figure 1B showing V vs. h for one of the neurons. Plot from the time of your initial condition through two full cycles of the oscillation. You do not need to plot the nullclines.

4. Turn in your matlab code. Include the code in the PDF report (copy and paste). No need to turn in the .m files

5. Extra credit: Modify your simulation to oscillate synchronously. Turn in a figure like Figure 4, and the matlab code that produced it.

Hints on interpreting the Wang and Rinzel notation:



The above circuit is their model for cell i (except the capacitor is not shown). Cell j's circuit looks identical, but for swapping i's and j's.

In cell i, there are two state variables governed by differential equations, Vi, and hi. Same for cell j.

The way the two cells are coupled is that the synaptic conductance in cell i (shown above as g_syn*S_inf(V)) is a function of the the voltage of cell j (and vice versa), and the synapse is inhibitory because Vsyn is -80mV.

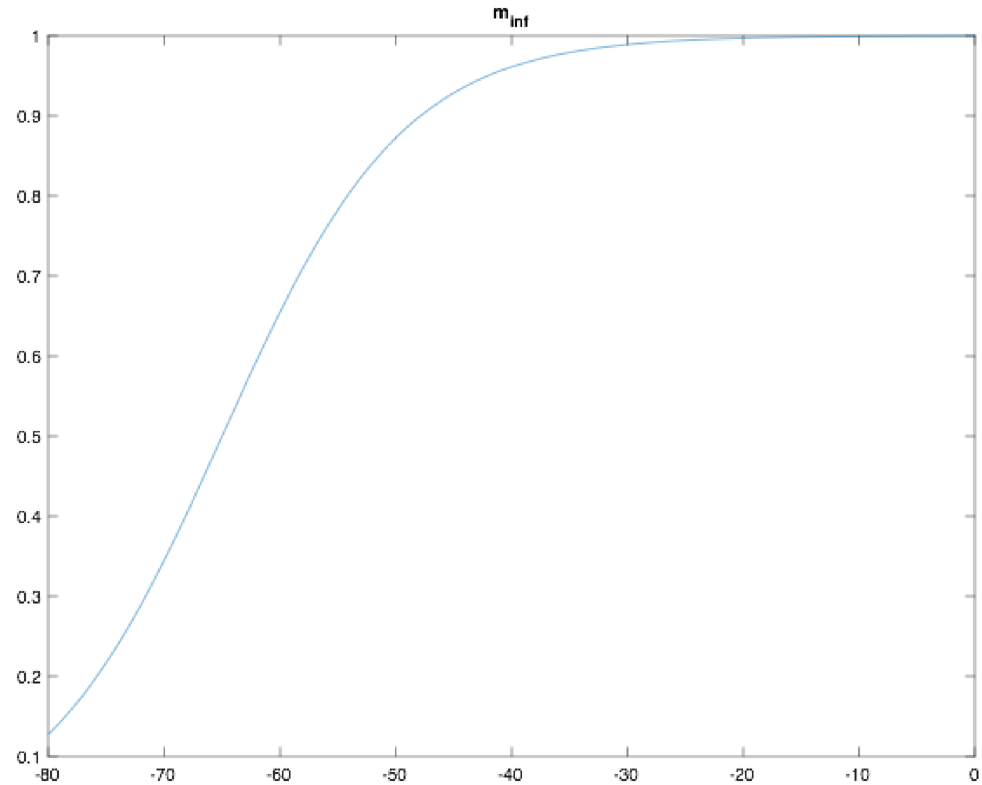
To relate Wang and Rinzel's notation to the notation we have been using, I would have labeled the 3 batteries EL, Epir and Esyn to make it clear those are constant battery values; I would have called gpir gpirbar and gsyn gsynbar to make clear those are the constant full-conductance values rather than time varying values. I would also drop (and did drop in the picture above) the notation sij because it's confusing -- we just need a generic function S_inf(V).   Also, phi (in the differential equation for h) is really just a scaling factor on tau_h, the bigger the phi, the shorter the tau.

The functions m_inf(V), S_inf(V), h_inf(V), and tau_h(V) are generic, and can be used for both cells (by plugging in the appropriate V).

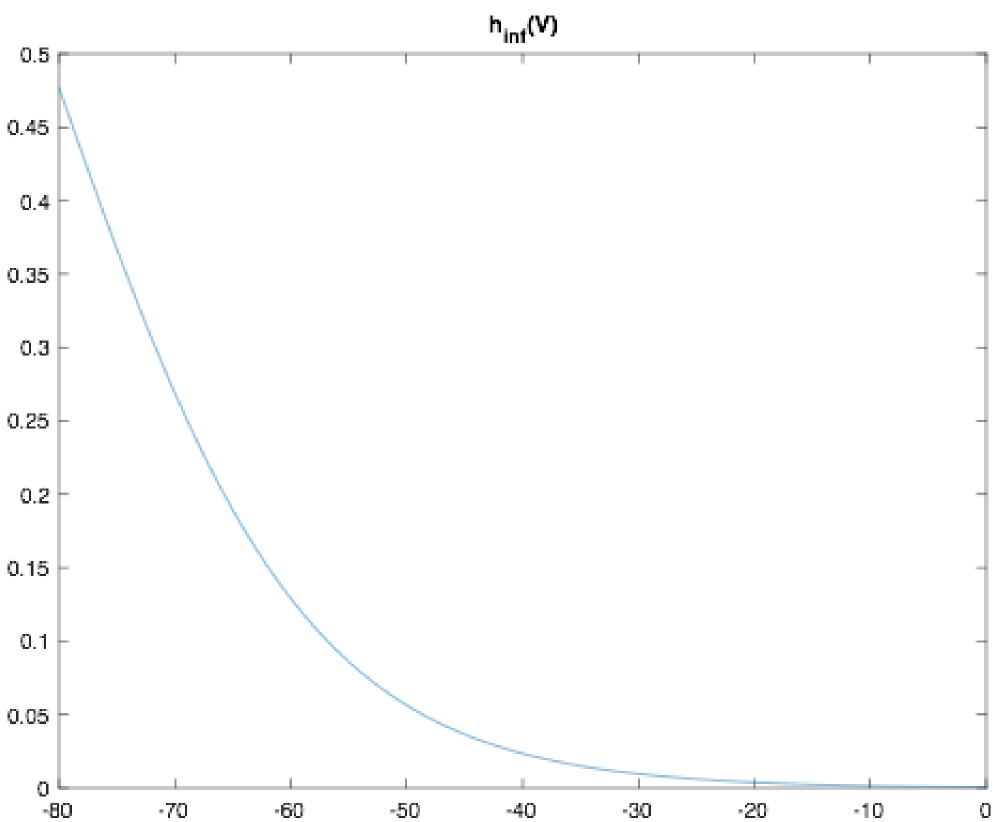
Another thing to consider: you can map the voltage differential equation into the form we have used in class,

dV/dt = (Vinf - V)/tau

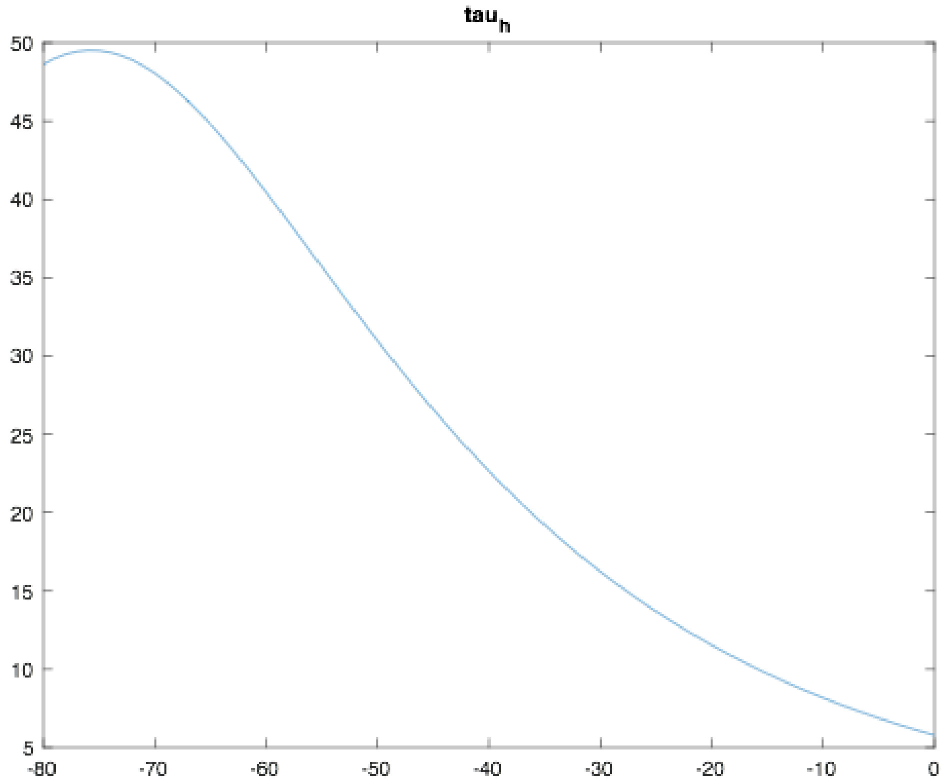
by computing Vinf and tau in the usual way (weighted average of batteries and C/gtotal). I like that formulation because then you have those two intuitive quantities explicitly available in your program, which could make your program easier to debug.

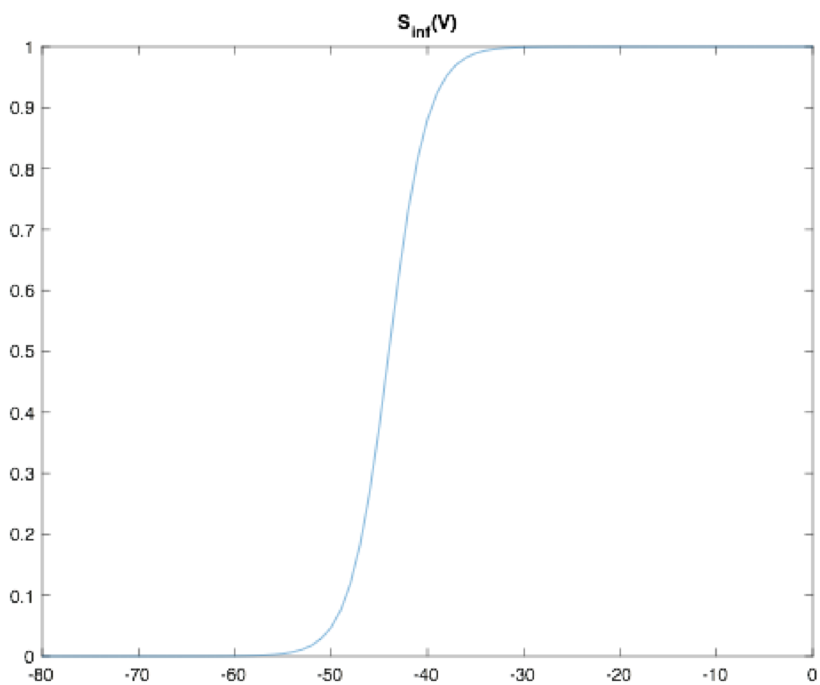


m_inf
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
-80
-70
-60
-50
-40
-30
-20
-10
0



h_inf(V)
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-80
-70
-60
-50
-40
-30
-20
-10
0






```
function [ val ] = S_inf( V )
theta = -44;
k = 2;
val = 1./(1+exp(-(V-theta)./k));
end
```

```
>> figure; plot(-80:0, S_inf(-80:0))
```