Lab #2 (Changing cell voltage *quickly*)

In this lab, we'll use BITSEY to run more simulations of single cells. This time, our goal will be to understand the short-term effects of how changing an ion-channel conductance affects V_{mem}.

Logistical note: like lab #1, you can (and should) work in pairs. However, please choose a different partner this time.

You can use the same BITSEY files as last week. This time, we'll be using and modifying the *setup_lab2*() and *setup_lab2b*() functions. You will notice that *setup_lab2*() is quite similar to *setup_lab1*(). How, then, is lab #2 different from lab #1?

In lab #1, we let the simulations run for many hours of simulated time. You probably noticed that the V_{mem} waveforms started at near 0 volts, then quickly "settled" to a value in less than a second, and then moved slowly (over tens of hours of simulated time) to their final values. In lab #2, we only care about those initial values – i.e., quasi steady state. Thus, all of our simulations will be just one second of simulated time. The wonderful thing about simulating in quasi steady state is that the simulations are all short 😊. This type of computation is how our brains work.

Thus, two very obvious changes for lab #2. First, we will run the simulations for only 1 second rather than the 100K seconds we used for lab #1 – so **python3 main.py lab2 1**. Second, *setup_lab2*() should definitely use the more stable (but much slower) numerical-integration algorithms (p.adaptive_timestep = False). Don't worry – the runs will still be fast.

You will run two simulations for this lab. The first simulation run is to simply run *setup_lab2*() and save the graph of per-cell V_{mem}. It should show that the different values of ion-channel conductivity result in different quasi-steady-state V_{mem}.

After you do that, you have one more simulation run. We would like to understand how resilient this type of computing is to small changes in a cell's initial ion concentrations. To do that we will use a new function *setup_lab2b*(). It's partially written, and you will fill it in.

- Create 4 cells as usual, with no gap junctions, just as in *setup_lab2*().
- All four cells should have the ion-channel diffusion constants from cell #2 of the first simulation – so a potassium diffusion constant of 10.0e-18 (i.e., *Dm_array*[*K*,:]= 10.0e-18), with the other ions remaining at their defaults. In *setup_lab2*(), this should have resulted in quasi-steady-state $V_{\text{mem}} \approx$-57mV.
- Leave cell #0 as a reference. For cells #1-3, perturb their initial [Na] *very slightly*, by an extra .005 moles/m^3 for each successive cell. This should be just enough so that each cell has its time=0 V_{mem} about 15mV higher than the previous cell (unlike lab #1, you will not compensate by changing another ion to restore charge neutrality). If you accidentally change the initial [Na] too much (and thus change the initial V_{mem} by more than about 0.5V), the simulator will not allow the simulation to run!

What happens as a result? The cells now have a different initial charge, and in fact no longer start out charge neutral. Their V_{mem} at time=0 will, therefore, now be nonzero, and in fact different in each cell. But does this affect the results at t=1 second? Turn in the graph of V_{mem} for both *setup_lab2*() and *setup_lab2b*().

Also, please answer the following questions:

1. Just as with lab #1, you can look at the data printed by *dump*() at the end of the simulation. In lab #1, we saw that for each individual ion, its flow through the pumps was equal and opposite to its flow through the ion channels. Is that still true? If not (and in fact it should not be), what claim can we make instead – that shows the system has reached quasi steady state?
2. In class, we discussed a model of quasi-steady-state V_{mem} that worked as a linear system. The debugging function *edb.analyze_equiv_network* () looks at the underlying physics and prints out the equivalent V_N and G for each ion. For run #1, you should see each cell having the same V_{Nernst} (because they all have the same initial ion concentrations), but different conductances (since we changed the ion-channel diffusion constants). Do the numbers it prints match your observed data for run #1 pretty well? I.e., using circuit analysis, do all of the V_N and G values, assembled into an equivalent circuit, correctly result in zero net current into the cell? To minimize busy work, just do this for cell #2.
3. What were your results for the modified run? Hopefully, you found that the system is quite resilient to small changes in initial concentrations. Can you explain why?

You should turn in two files: your main.py and a report with the graphs and the answers to the questions.