## Computational Neuroscience 4030/6030 – Project 2 (25 pts)

The goal of this project is to investigate basic properties of *stochastic resonance* in neural signal processing. This is a complex area of neuroscience where there is much contention and uncertainty. However, we will consider certain ideas that indicate a possible way that neurons could use "background noise" to improve their ability to respond to weak inputs.

You will reproduce the situation shown in the textbook's Figs. 9.7-9.8 using the templates for 2D models of spiking cells and other code that I distributed for class. You will also add some additional information than what's given in the book. Carefully and methodically work through these small steps in order, which guide you to the end. There is a special task for graduate students at the end, and some optional bonus credit. 8 points is reserved for appropriate functions and plotted output in steps 2-4 . The rest of the points are for your written responses to the questions posed to you. You will lose 2 points for any written answer that you skip or do not justify adequately. Answers only need to be a couple of sentences each.

*Don't forget to put titles and axes on your graphs* so that you can keep track of which is which.

**1)** Read Section 9.4 carefully. Eq. (9.7) is implemented via the script `Project2.py`. You may need to use VCL so that you will have access to the Dopri integrator. Otherwise, your script will run very, very slowly with Vode. You should rename it 'Project2_NAME.py' where NAME= your initials.

**2)** Write a function named `ISI` that accepts a trajectory from the neuron model as an argument, and returns an array of inter-spike intervals (abbreviated ISI) in that function. The ISIs are simply the times between successive spikes. Your function should return an empty array if the neuron does not spike or if it only spikes once. (You will not get points for this part unless you actually write this as a Python *function*, which you will use multiple times later in your script.)

**3)** Write a function `avfreq` that accepts a list or array of spike times and returns the mean frequency of the spike train in Hz. Test it on a spike train from a previous week's script to make sure it's correctly working.

**4)** Write a function named `testrun` that accepts three parameters: a noise amplitude, a sinusoidal amplitude, and the final integration time. This function will set the corresponding parameters in the model, run the integration and process the output, plotting the voltage trace in one figure and an ISI histogram in another (use a 1ms bin width and a largest ISI of 1500ms – *this is important later!*). It should also *return* the average frequency value.

You can use the `plt.clf()` function to clear the current figure. The current figure can first be selected using `plt.figure(i)` where *i* is the figure number.

**5)** Test your function 'testrun' by applying sinusoidal stimuli at 55 Hz of amplitudes 0.05 (units are pA) **with no noise**, for a length of 500 ms. Explain what happens to the neuron in detail – account for all the behavior that is plotted, including the contents of the histogram and the average frequency. Save the voltage trace only for the report.

**6)** At the ipython prompt, call the test function repeatedly, increasing the sinusoidal amplitude until the neuron regularly spikes.

- What is the smallest amplitude that causes this in the absence of noise? Report your value to the nearest 0.01pA. Save the figure showing the voltage trace graph before it regularly spikes.
- Explain why the neuron spikes and why it spikes at the average frequency that your function reports.
- Save the figures of the first spiking voltage trace and its ISI histogram.

**7)** For a sinusoid with a fixed amplitude of 0.06 pA, add increasing amounts of Gaussian noise (by increasing its standard deviation 'amplitude') to the neuron, starting at 0.01 *and integrating for 5000 ms.* Record the average frequency of the responses and look at the ISI histograms (use zoom if necessary). You will only need to add very small amounts of noise (a standard deviation of 0.2 will be way too much!).

- At a standard deviation noise of 0.01, zoom in close on the voltage trace (e.g., a window of 200ms) and explain what you are seeing.
- What is the smallest amount of noise that causes any spiking? Report your value to the nearest 0.001. Explain why there are spikes now, and why there are only a few.
- What, in your judgment, is the smallest amount of noise that makes the histogram reliably indicate the sinusoidal frequency? Let us call this value $\nu_0$. Consider the number of spikes recorded in the 5000ms window (and indicated on the y-axis of your histogram) when making your assessment. *Hint: Your histogram should now resemble the one in the textbook.*
- What is the average frequency recorded for your chosen value of $\nu_0$? Explain why it has that value compared to the driving frequency.
- Why should we integrate for this long now? Explore this by comparing your histogram for only 500ms integration time with your chosen $\nu_0$.
- Explain in your own words the shape of the histogram for your $\nu_0$. Explain the presence of any multiple peaks in your histogram, and why they are centered around the specific ISI values that you observe.

**8) GRAD STUDENTS ONLY (or 5 bonus points for undergraduates):** ***Population coding.***
Why is the average frequency a poor measure of the quality of the timing performance of the neuron at noise levels around $\nu_0$? Consider a population of these neurons all being driven by a common noisy input of strength $\nu_0$. Suppose you were measuring the population output of all these neurons simply by overlaying all their spike trains on top of each other. Explain how would you expect the ISI histogram to be different? What consequence does this have for information processing?

**<u>!! TEN !!</u> bonus points (optional for everyone)**: Adapt your code in this script to demonstrate this network summation effect in a network of *n* neurons that are *not* coupled together and only receive the common noise input of strength $\nu_0$. Show the histograms for $n = 2$ and $n = 10$, and comment on your observations. Hint: you might like to make use of the numpy concatenate and sort functions, which you can read about online. If you get stuck using them please ask me for help.

**9)** Put together all your responses in a short report combining the figures, descriptions, and explanations. Paste your script's code at the end of the report (it does *not* have to show all the commands you entered in ipython to reproduce your graphs). Format it using a mono-space font such as Courier, slightly reducing its size if necessary to fit 80 columns across. Submit your report by email, before **Sunday 11th November at midnight**.

---

**Start this project as soon as possible, so that you can visit me in office hours and/or email me to ask questions about completing this in time. You may ask your colleagues for general advice about understanding the meaning of the questions, and general programming advice *only*. Don't submit work that is *at all* identically written or coded to that of your colleagues, and do not let others use any parameter values that you have found by yourself! (See the syllabus for the rules and consequences of plagiarism.)**