EECE 5136/6036: Intelligent Systems

Homework 2: Given 9/14/23; Due 9/24/23 (11:59 PM)

One of the most widely used current models for neurons is the *Izhikevich model*, proposed by Eugene Izhikevich (Izhikevich, 2003, 2004). It models the membrane potential of a neuron by two coupled differential equations:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I(t) \tag{1}$$

$$\frac{du}{dt} = a(bv - u) \tag{2}$$

with the condition that:

$$if \;\; v \geq 30, \quad v = c, \;\; u = u + d \tag{3}$$

Thus, the model is defined completely by four parameters: a, b, c and d. By setting the parameters to different values, neurons with many different natural behaviors can be obtained, as described in Izhikevich (2003, 2004). Whenever v crosses 30, a spike is said to have been generated, but in fact, the model generates the full shape of the action potential. Thus, it is more than an integrate-and-fire model but far less complicated than a full biophysical model.

In this homework, you will implement several types of neurons using the Izhikevich Model. A MATLAB program to simulate and plot the behavior of one type (regular spiking (RS) neuron) is given in the file *neuronRS.m* in the attached materials (Izhikevich, 2003), but other types can be generated using the same code with different values of a, b, c, and d. You can modify this or write your own code in any language of your choice.

Problem 1. (100 points) Implement the program to simulate a single regular spiking (RS) neuron and drive it with different levels of external input I. The parameter settings for the RS neuron can be read from Figure 2 (upper right) in Izhikevich (2003) In the file *neuronRS.m*, this is implemented in lines 24-28, which produces a step-function I input set by default to $I = 1.0$. The step occurs at time $T1 = 50$, but you should set $T1 = 0$ (constant input from the beginning of each trial). Note that the membrane potential is represented by V in the code, while the array VV is used to save the time-series of V for plotting. The array *spike_ts* is used to mark spikes (0 = no spike, 1 = spike).

You will run several simulations with different I values, starting from $I = 0$ (which should produce no response) and increase in each trial in steps of 1.0 to a level of $I = 40$. Thus, you will run a total of 41 trials for $I = 0,\ 1.0,\ 2.0,\ ...,\ 40.0$. In each case, you should simulate the neuron for $1,000$ steps, discard the output time-series VV for the first 200 steps, and for the last 800 steps, calculate the *mean spike rate*

$$R = \frac{number\ of\ spikes\ in\ the\ last\ 800\ steps}{800} \tag{4}$$

Discuss your results briefly (15 lines max). What kind of pattern do you see, and what does it say about the behavior of the neuron? In what sense is the neuron performing any useful function here?

Required Plots:

1. The full 1,001-step time-series of the membrane potential v for the $I = 1$; 10; 20; 30, and 40 cases as five separate graphs in a single figure (i.e., five plots stacked vertically). Note that you will do

this by plotting VV, which is 4,001 data-points with 4 data-points for each time step if you use the step-size $\tau = 0.25$ in the code (you can choose a different step-size, but no larger than 0.25). Thus, the 4,001 data-points represent time from 0 to 1,000. The MATLAB code plots both VV and *spike_ts* for illustration - you will only plot VV.

2. A graph plotting R vs. I. Again, remember that when you count spikes over 800 time-steps, you will actually be counting spikes over 3,200 data points, but will divide by 800, not 3,200 because there are 4 data-points for each time-step. You can get the spike count from the *spike_ts* array.

Make sure all your graphs are titled properly, and all axes are labeled with appropriate values on all ticks, etc. When you discuss a graph in your text, it should be cited by its title or label (e.g., 'Figure 1.2(a)'), not as "the figure below" or "the figure on the last page". You will need to add this labeling – it is not in the code.

Keep in mind that you are simulating nonlinear differential equations. The given code uses Euler's method to do this, which requires integrating the state variables in small steps. Thus, each actual time step is simulated as 4 micro-steps of update. The step-size $tau = 0.25$ indicates this. Due to this, the time-series generated over 1,000 time-steps has 4,001 data-points – one initial value for $t = 0$ and then 4 micro-steps for each actual step. However, time should be labeled on the graphs in actual units, not the micro-steps used by the simulator. Thus, the point marked t=100 in your graph will actually be the 401st point of the generated data, and your time axis should go from 0 to 1,000.

Again, in the given code, I is set to remain 0 for the first 50 steps. You should change this so I starts at the desired value for that trial from step 1 and then remains fixed.

Problem 2. (50 points) From Figure 2 in Izhikevich (2003), read the a, b, c, d values for the fast spiking (FS) neuron and repeat the simulations of Problem 1.

Required Plots:
Plot the same two figures as in Problem 1, but in the second figure, also plot the R v. I curve for the RS neuron from Problem 1 (so this figure will have two curves in the same graph – one for RS and the other for FS).

Discuss your results briefly (10 lines max). In particular, discuss the differences (if any) in the mean spike rate plots of R vs. I from Problem 1 and Problem 2.

Problem 3. (100 points) In this problem, you should ask ChatGPT (or some other LLM that you prefer) to generate the code for simulating a regular spiker izhikevich neuron, running it for 1000 time-steps with inputs of I from 0 to 40 in steps of 1, and plotting the spiking rate versus I using the last 800 time-steps to calculate the rate – basically repeating Problem 1 except for the 5 individual time-series. You should leave the choice of how the code is implemented to the system rather than specifying a method, the value of τ, etc. You can ask for a specific language – Python, Matlab, Java, etc. – but if you don't, it will use Python, which is fine. Download the code and run it to generate the figure. Compare the figure generated with the one you obtained in Problem 1 by plotting both curves on the same graph. Briefly discuss what differences, if any, you see between the results (likely not much).

If the code did not run, or the figure produced was incorrect, figure out how to fix it, and then run it to get the correct figure. Keep track of what the errors made by the system were and how you fixed them. Keep in mind that the system may not have included all the libraries such as numpy and matplotlib in its code because it has them built-in. You will need to add these in if needed when you run the code yourself.

Document the entire process – the transcript of the whole session, the code generated (correct or incorrect)

– in your answer. If you needed to fix the code, add in your code along with a brief description of what you fixed. Make sure you write the name and version number of the LLM used.

Keep in mind that ChatGPT and other LLMs don't always respond the same way to the same prompt, and their ability to respond correctly depends a lot on the prompt. Make your prompt as clear as you can. If you need to do the prompting in 2 or 3 steps, that is fine. Just include all of that in your transcript.

Reference Material:

The accompanying *HW2_Material.zip* file has three items:

1. Izhikevich paper from 2003. This describes the model and you should read it.
2. Izhikevich paper from 2004. This longer paper gets more into the underlying theory. You can read as much of the paper as you want. It uses different names for neuron models than those used in the 2003 paper.
3. MATLAB program to implement a single neuron. You can use this, or re-implement in a language of your choice. It is a very short, simple program.

If you cannot access the material, please send me mail at. Ali.Minai@uc.edu.

Instructions for the Report:

Submit a report of your work all problems written like a publication in *a single column format* in PDF. Please look at the technical report by Bengio included with the last homework, and use a similar format, though some small variations are all right. The code you wrote should be submitted as a separate file.

The report should meet the following guidelines:

1. **The report file should be named HW2_report.pdf**.
2. Answers to all problems should be in a single report, but begin a new page for each problem. For example, if the answer to Problem 1 takes 1.5 pages, leave the remaining part of the second page blank and begin the answer to Problem 2 on page 3. *Number all pages consecutively.*
3. Use *11 point text, single spaced* for the report. Figure captions can be in a smaller font size. Use Times Roman, Calibri, or some other standard font suitable for a professional document. No weird calligraphic fonts please!
4. The answer to each problem should have a heading with the problem number.
5. There should be three separate sections in the answer to Problems 1 and 2: **Problem Statement** (1-2 lines), **Results**(graphs with captions), and **Discussion** (as directed in problems). Each section heading should be in bold letters as in the sample by Bengio. The **Problem Statement** section should not simply copy the problem statement in the homework, but should say what is being done in that problem. The discussion should refer to the figures by number (see below). For Problem 3, you should have four sections: **Methods**, **Result**, **Discussion**, and **Transcript**. In the first section, you will state which system you used, what was your prompt and why. The Results section will just have the graph with the two rate curves on the same graph. The Discussion session will summarize how things worked out, if you needed to fix errors, what fix did you make, and how you interpret the results. These sections together should be no more than 1 page including the figure. The transcript can be as long as needed and should begin on a new page labeled as **Transcript**.
6. *The figures must include all the plots required for the problem.* Make sure the plots are connected properly with the text.

7. All figures should be integrated into the text rather than being put on separate pages (refer to the Bengio document or the way Figure 1 is embedded in this document). Each figure must be given a number, e.g., the first figure in the answer to Problem 1 could be Figure 1.1. Each figure should have a caption underneath the figure, e.g.,

   Figure 1.2: *Plot of mean firing rate as a function of input I. The firing rate is calculated by averaging over the last 800 time steps of a 1,000 time step simulation.*

   All axes in each graph must be labeled. Font sizes for any symbols or tick labels must be large enough to be legible. If plotting multiple plots on the same graph, each plot must be clearly distinguished by different colors, line types or symbols. Lines should be thick enough to be clearly visible.

8. . The answers to the first two problems should be no more that 2 pages each *including all figures.* You should figure out how best to fit the figures on the page so they are legible but fit within the limits. The answer to problem 3 will have a

9. *All of the above should be in the report document, and NOT in the code. You should not expect anyone to read your code to see your results, comments, discussion, etc.*

10. **Code:** *The entire code used to generate your results for Problems 1 and 2 must be submitted with your homework in a separate, runnable file.* It should be commented to identify every significant part of the algorithm by which you generated your results. You can use parts of the Izhikevich code, but your program must include a statement at the top acknowledging that part of the code came from this source. The code for Problem 3 should just be part of the transcript of that problem.

Submission Instructions:

The homework is due at 11:59 PM US Eastern Time on the due date.

You should submit your report on-line through Canvas. Your submission will have three documents: 1) A report **HW2_report.pdf** as described above; 2) A file (or .zip or .rar file) with all your source code; and 3) A brief README file with instructions on how to compile (if needed) and run the code. If Canvas does not let you submit a file with the type postfix (.py, .m, etc.) of a source code file, try changing the postfix manually to .docx or .txt and submit it. State in the README file what the postfix should be changed to in order to run the code.

Grading:

Points will be awarded for correctness of the results, proper plots, and the clarity of description and conclusions.

You may consult your colleagues for ideas, but *please write your own programs and your own text.* Both the text of the reports and the code will be checked randomly for similarity with that of other students. *Any text, graphs, or code found to be copied will lead to an automatic zero on the entire homework for all students involved.* Repeat offenses in the future will incur more severe consequences.