1 Artificial Neurons

For this part of the assignment you will simulate a neuron based on the leaky integrate-and-fire model (section 4.1 in the lecture notes). Make sure the file `nsimulator.m/py` is in your working directory. The simulator documentation is available in the usual way, and the script `nsim-example.m/py` contains an example of what you can do with the simulator (and how).

1 a* Create a simulator with a single neuron, setting the initial membrane potential value to $-60\,\text{mV}$. Run the simulation for $0.2\,\text{s}$ while injecting a current of $0\,\text{nA}$. Repeat the simulation for $1\,\text{nA}$ and $10\,\text{nA}$. Describe what happens to the resting membrane potential as you vary the input current.

b* Repeat the experiments above, this time using different initial membrane potential values: $-80\,\text{mV}$ and $0\,\text{mV}$. Is there any difference in the neuron's behavior? Why or why not?

c* Create a simulator for two neurons. Inject $2.5\,\text{nA}$ of current into Neuron 1 and $0\,\text{nA}$ into Neuron 2. Set the membrane potential of Neuron 1 to $-60\,\text{mV}$, and the membrane potential of Neuron 2 to $-70\,\text{mV}$. Make Neuron 1 inhibit Neuron 2. Simulate for $0.1\,\text{s}$. Repeat the simulation for different values of injected current into Neuron 2: $1\,\text{nA}$ and $10\,\text{nA}$. Compare the behavior of Neuron 2 to the behavior of the neuron in the previous part (single neuron simulation). What is the effect of Neuron 1 on Neuron 2?

d* Repeat the experiment above for an excitatory connection. What is the effect of Neuron 1 on Neuron 2?

e* Create a simulator for two neurons. Inject $2.5\,\text{nA}$ into each one and initialize the membrane potential to $-60\,\text{mV}$ for each one. Connect the two neurons so each excites the other. Simulate without recording for $1\,\text{s}$, and then record for $0.1\,\text{s}$. Do this several times. Do you observe a phase offset between the spikes of the two neurons as seen in the figure below?

If not, remove the calls to `nsim.set_mpotential`, allowing the initial membrane potentials to be initialized randomly (between $-50\,\text{mV}$ and $-70\,\text{mV}$). Simulate several times again. Did this make a difference? Repeat for inhibitory connections. Do the neurons tend to synchronize their spikes?

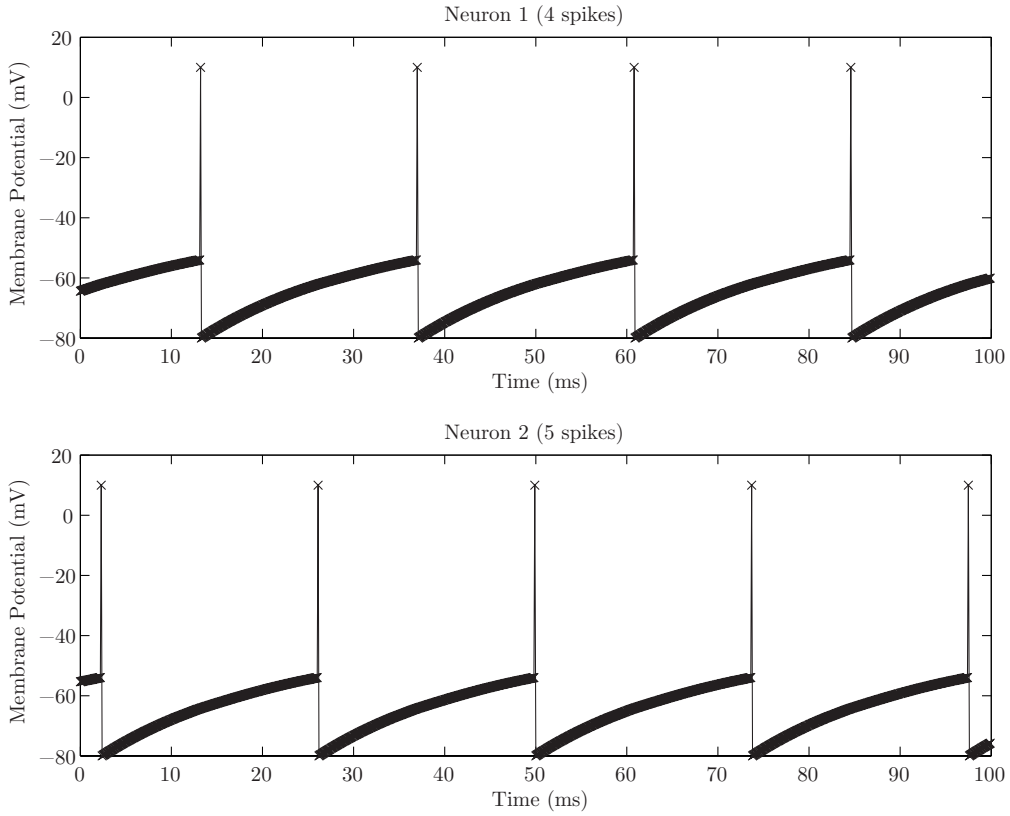


Figure 1: Example of phase offset between two neurons.

2 Neural Networks as Linear Systems

2 a Build a 5×5 tridiagonal matrix with 1s along the main diagonal, as well as along the diagonals above and below it (and 0 everywhere else). Then multiply it by the 5×1 vector $v = [1\ 2\ 3\ 4\ 5]^T$. What is the result? We can think of the matrix as a linear function, or a linear transformation, being applied to an input vector, which gives us another vector as output. What kind of operation is done by this matrix on the input vector?

– b Test the homogeneity of this operation. What happens to the result if we apply the same transformation to $2v$? What about if we multiply v by an arbitrary constant scalar, α?

– c To test the additivity property, apply the matrix now to the vector $a = [0\ 1\ 2\ 3\ 4]^T$ and also to $b = [1\ 1\ 1\ 1\ 1]^T$. What is the result of each operation? Add these two output vectors together and compare with the result from a.

– d* Build a matrix that, when operating on the vector $w = [1\ 2\ 3\ 4\ 5\ 6]^T$, outputs a 2-by-1 vector containing as its first and second entries the means of the first and second halves of w, respectively.

– e* In class, we learned about the *Limulus* and experimented with a simple feedforward model of processing in the lateral neural plexus. In this exercise, we want to represent these types of models in matrix form. That is, we can think of the input and output as vectors in $\mathbb{R}^n$, where n is the number of cells in the layer. (For simplicity, we will consider both layers to have an equal number of cells.) Thus, any linear transformation from input to output can be represented as an $n \times n$ matrix. An appropriate matrix, when applied to the input, will result in the desired output.

Recall equation (5.4) from the notes; in its general form, it can be written as:

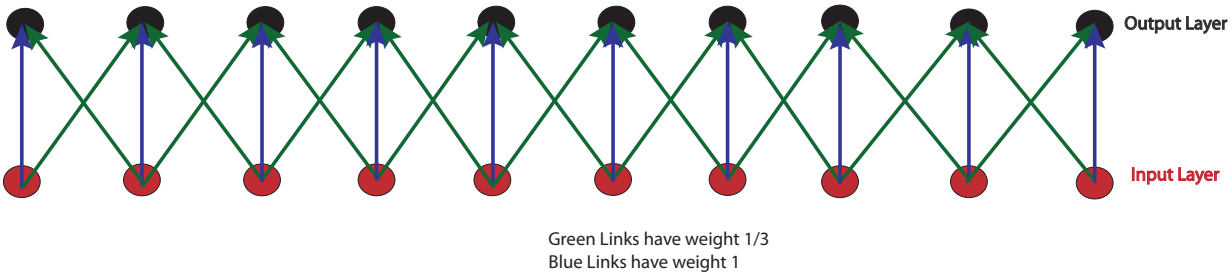
$$F_i = e_i + \sum_{j \in \mathcal{N}(i)} a_{i,j} e_j,$$

where $\mathcal{N}(i)$ is the neighborhood of influence of cell i and each $a_{i,j}$ is negative for inhibitory connections, or positive for excitatory ones. Let $n = 10$ and the neighborhood of influence be one cell on each side of the current cell i, e.g. if $i = 5$, $\mathcal{N}(i) = \{4, 6\}$. For boundary cells, i.e. $i = 1$ or $i = 10$, let the neighborhood be $\{2\}$ and $\{9\}$, respectively. As in the lecture, let the weights $a_i = -0.2$, $1 \leq i \leq 10$.

Represent the above equation in matrix form. (*Tip: to avoid having to enter each value manually, you can start with a generic* 10×10 *matrix and use for-loops to fill in the values.*)

f* Create the 10×1 input vector $[10\ 10\ 10\ 10\ 10\ 20\ 20\ 20\ 20\ 20]^T$ and plot it using the `plot` command. What kind of function does this vector look like? Apply your matrix to this input vector. What do you observe about the resulting output vector? (Ignore the boundary values!) If we interpret the values in the vectors as light intensities, another way to visualize them is by using `imagesc/plt.imshow` on each vector separately using the `gray` colormap (for the output vector, you will get a better result by not displaying the first and last entries). Remark qualitatively on the results.

g* Now consider the following feedforward model:



(Note the green links have positive weights, $+1/3$.)
Find the matrix representing the action of this network. Apply it to the vectors $[10\ 10\ 10\ 10\ 10\ 20\ 20\ 20\ 20\ 20]^T$ and $[10\ 8\ 10\ 8\ 10\ 8\ 10\ 8\ 10\ 8]^T$. Qualitatively, what does this network do? (Again, ignore the boundary values.)

h* Suppose we built a multi-level feedforward model using the model in e first and then used output from that network as input into the input layer of the model in f. Remark on what you expect the combined action of this system to be. How can we build a single matrix representing it? Apply this matrix to the vector $[10\ 10\ 10\ 10\ 10\ 10\ 10\ 10\ 10\ 10]^T$ to verify your remark.