CNS2025: Homework 5

Due: 2025-10-08 23:59

The objective of the current homework is to examine how well the linear–nonlinear Poisson cascade (LNP) model with kernel, that is constructed using the spike-triggered average (STA) of a simulated integrate-and-fire neuron, can be used to predict the firing rate of the neuron when it is exposed to a new, colored stimulus.

We will compare the output of the linear kernel with the firing rate estimated from the new spike train from the neuron to see if `ReLU` with proper scaling can predict the estimated firing rate.

Exercise 1

Use the function `simulate_lif_model()` in the code file [`code05.ipynb`]() without specifying the `input_current` (a white-noise stimulus input will be used by default) to generate a spike train for 10^4 seconds. Using the `spike_train` and `input_current` returned by the function `simulate_lif_model()`, calculate and plot the STA of the stimulus for the neuron. (The time range of `np.array([-0.5,0.1])` should be enough to hold the STA)

Exercise 2

Using the `inputs_alt` in the "Filtered stimulus" section as the input current to the `simulate_lif_model()` function and generate a new spike train for 10s. Using a Gaussian sliding window with standard deviation σ = 0.05 s to calculate the time-dependent firing rate from the generated spike train.

Question: If you supply a different seed to the `simulate_lif_model()`, will you get a different spike train?

Exercise 3

Apply the STA obtained in Exercise 1 as the kernel to convolve with the input stimulus of Exercise 2 following the linear-nonlinear model and obtain an estimated firing rate through a properly-scaled `ReLU` nonlinear function. Compare the results with that from Exercise 2 (For example, you can make a scatter or point plot using firing rate values from Exercise 2 as y and the convolution results of LNP model as x.) and find the best multiplication factor (scale) of the `ReLU` function to reproduce the firing rate.

Exercise 4

After Exercise 3, we have a complete LNP model. Use this model to obtain the time-dependent firing rate from the `inputs_alt` input and use this rate to generate 100 spike-trains (using the `gen_spikes()` function with different seeds) in the event-based representation. Make a firing raster plot using the results.
