Excercise 2:

Contents

question

Hands-on Exercise: # 2

SNS 2023

2. Construct an integrate-and-fire model responding to a "noisy" input representing the *in vivo* environment. This model is based on the equation

$$\tau_{\mathrm{m}} \frac{dV}{dt} = -V + E_{\mathrm{eff}}$$

with $\tau_{\mathrm{m}} = 10$ ms. The threshold and reset potentials for the model are $V_{\mathrm{th}} = -54$ mV and $V_{\mathrm{reset}} = -80$ mV. E_{eff} is given by

$$E_{\mathrm{eff}} = -56.0 + \sigma_V \sqrt{\frac{2\tau_{\mathrm{m}}}{\Delta t}}\ \mathrm{randn}(1\,, \mathrm{length(t)})$$

where Δt is the time step size in your program, σ_V is a parameter (see below), randn is the matlab random number generator for a normal distribution, and t is the vector of times in your program. Consider values of σ_V in the range $0 \leq \sigma_V \leq 10$ mV.

a) Turn off the spike generation mechanism in your model (by setting V_{th} to an extraordinarily large value, for example). Plot the standard deviation of the membrane potential fluctuations that arise from different σ_V values in this range as a function of σ_V. If you are doing things right, the standard deviation of V should be equal to σ_V over the entire range.

b) Plot the average firing rate of the neuron (defined by counting spikes over a sufficiently long time interval and dividing by the duration of that interval) as a function of σ_V for values in this range.

Part a

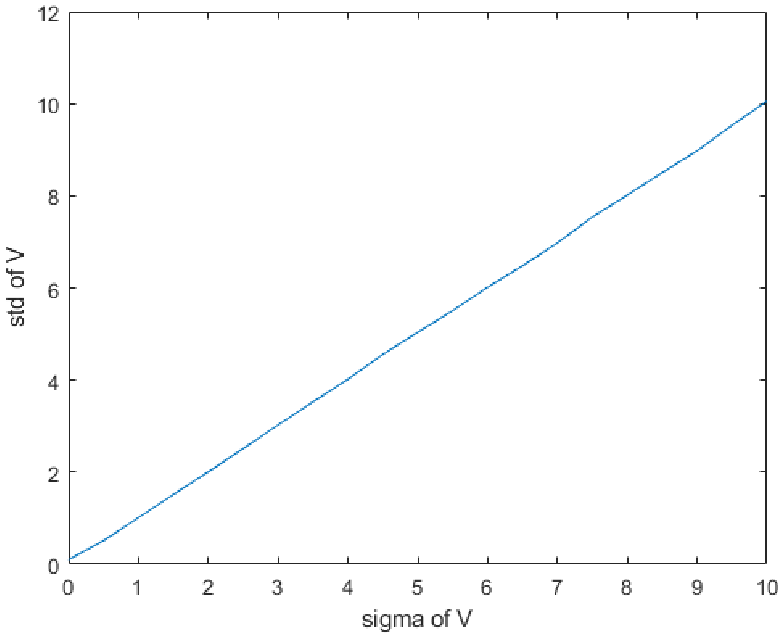
```
close all
clear
clc

Taw_m = 10; %in milli seconds
V_th = -54;  %in milli volts
V_reset = -80;  %in milli volts

tot_data_points = 3000000;
dt = 0.1; %time step in ms

sigma_v = 0:0.5:10;
std_of_V = zeros(1,length(sigma_v));
V = zeros(1, tot_data_points);
V(1) = V_reset;
for j = 1:length(sigma_v)
      Eff = -56 + sigma_v(j)*sqrt(2*Taw_m/dt)*randn(1, tot_data_points);
      for i = 1:tot_data_points
            V(i+1) = V(i) + (dt/Taw_m)*(-V(i) + Eff(i));
      end
      std_of_V(j) = std(V);
      V(1) = V_reset;
end
```

```
figure
plot(sigma_v, std_of_V),    xlabel('sigma of V'),    ylabel('std of V')
```



we can see that without implementing the threshold, the response of the neuron is linear

Part b

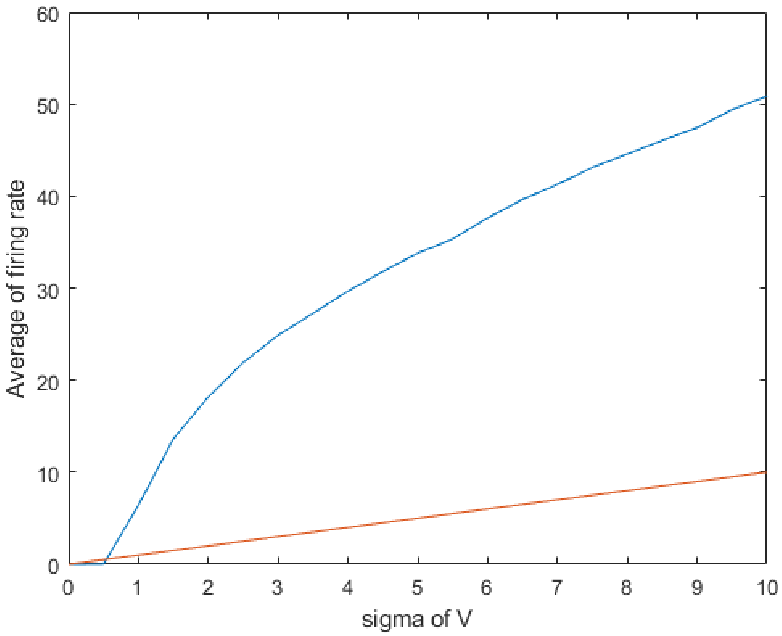
```
close all
clear
clc

Taw_m = 10; %in milli seconds
V_th = -54;  %in milli volts
V_reset = -80;  %in milli volts

tot_data_points = 3000000;
dt = 0.1; %time step in ms

sigma_v = 0:0.5:10;
std_of_V = zeros(1,length(sigma_v));
V = zeros(1, tot_data_points);
V(1) = V_reset;
a=0;  %this represents the number of spikes

for j = 1:length(sigma_v)
      Eff = -56 + sigma_v(j)*sqrt(2*Taw_m/dt)*randn(1, tot_data_points);
      for i = 1:tot_data_points
            V(i+1) = V(i) + (dt/Taw_m)*(-V(i) + Eff(i));
            if V(i+1) > V_th
                V(i+1) = V_reset;
                a = a+1;
            end
      end
      std_of_V(j) = a/(tot_data_points * dt/1000);
      % std_of_V(j) = std(V);
      a=0;
      V(1) = V_reset;
end
figure
plot(sigma_v, std_of_V),      xlabel('sigma of V'),    ylabel('Average of firing rate')
hold on
plot(sigma_v, sigma_v)
```



But, with adding the threshold and reseting to our model, the response beacme nonlinear

---

Published with MATLAB® R2023a