UNIVERSITY OF
LINCOLN

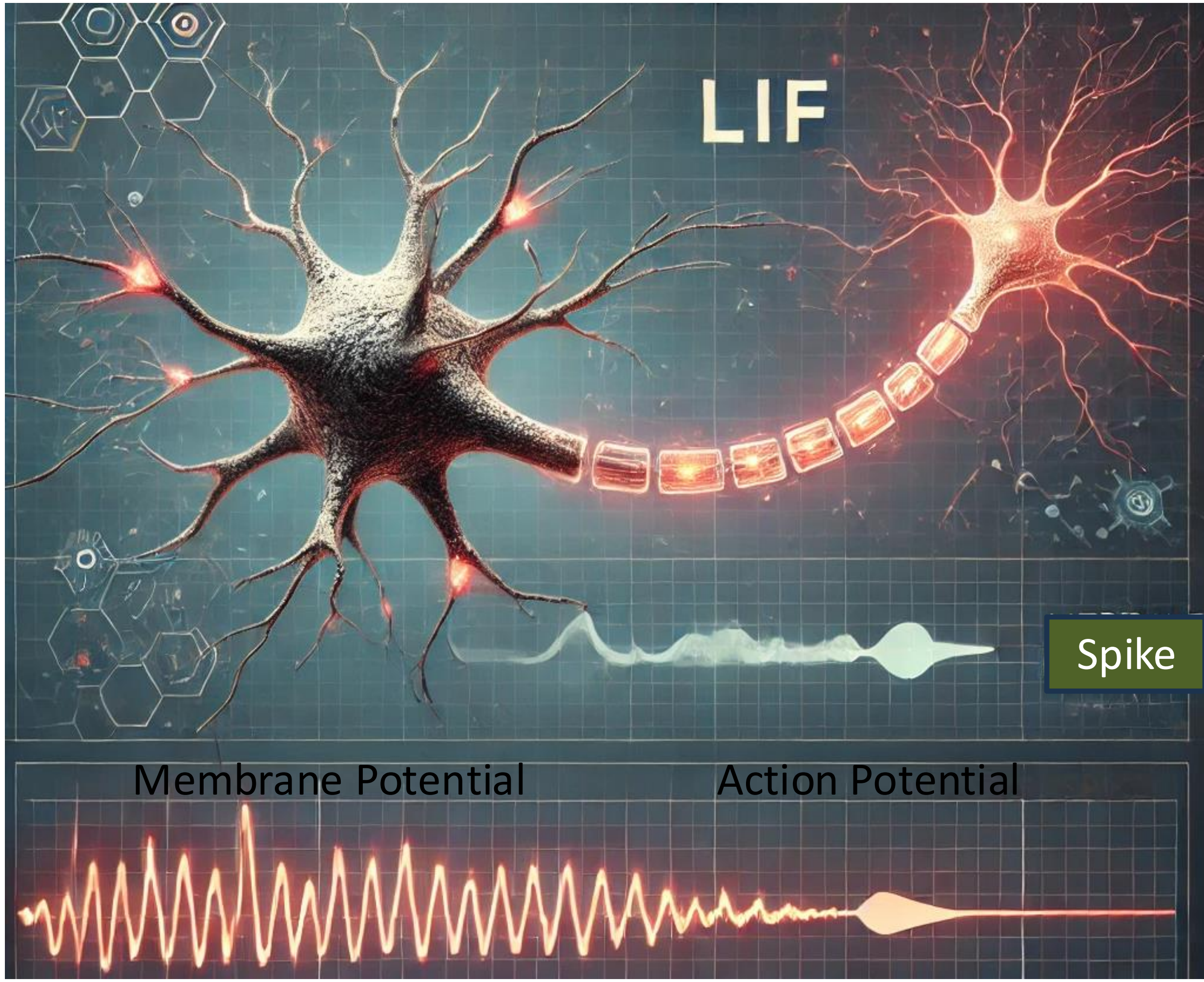
The Leaky Integrate and Fire (LIF) Model

CMP9783 – Neural Computing Week 4

cfrantzidis@lincoln.ac.uk

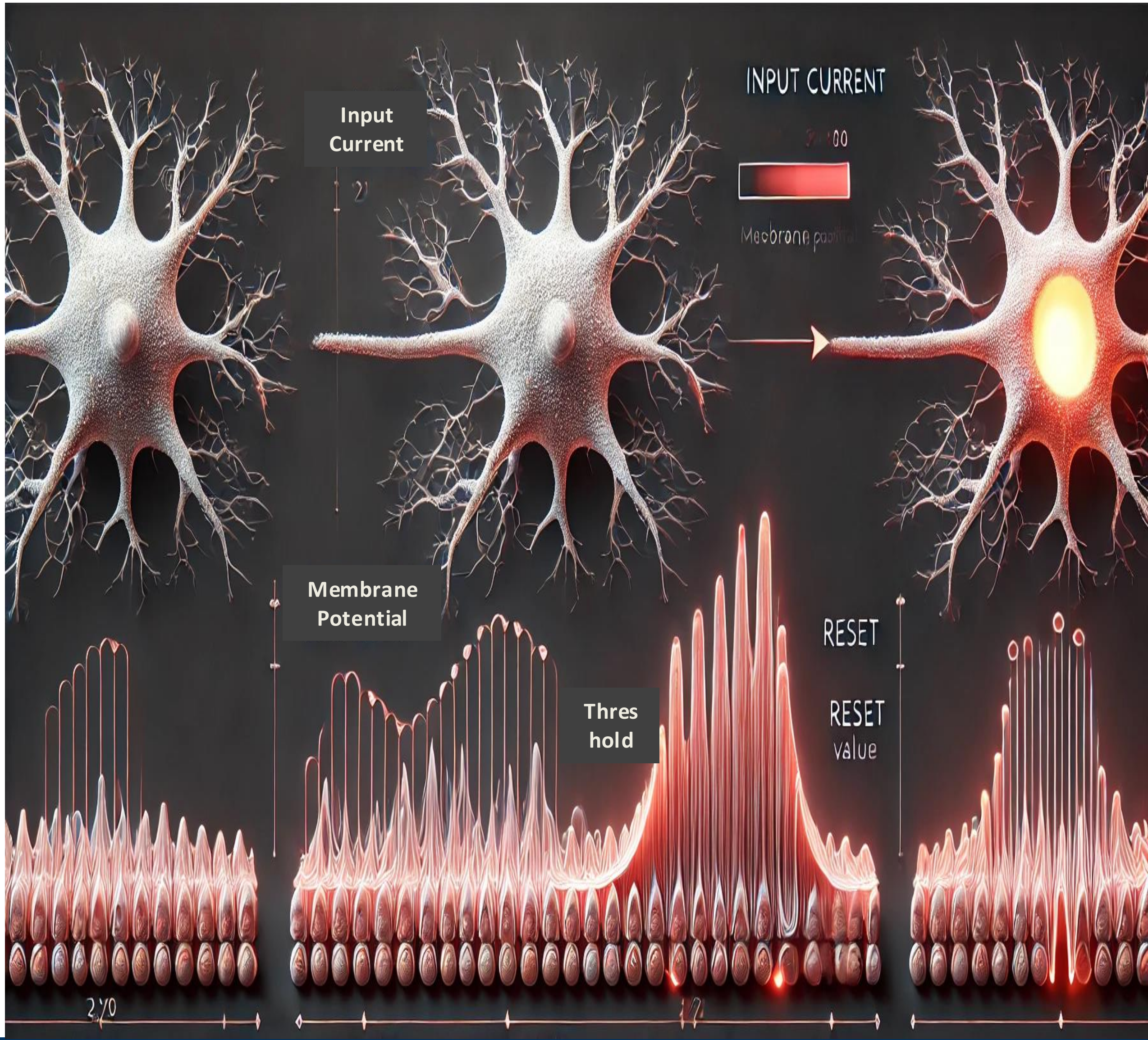
The LIF Definition

- It is a simplified mathematical model used to describe the electrical behavior of neurons.
- It is one of the most basic models of neuron activity in computational neuroscience.
- The LIF model focuses on how a membrane potential evolves over time in response to incoming stimuli and how it generates action potential or “spikes”



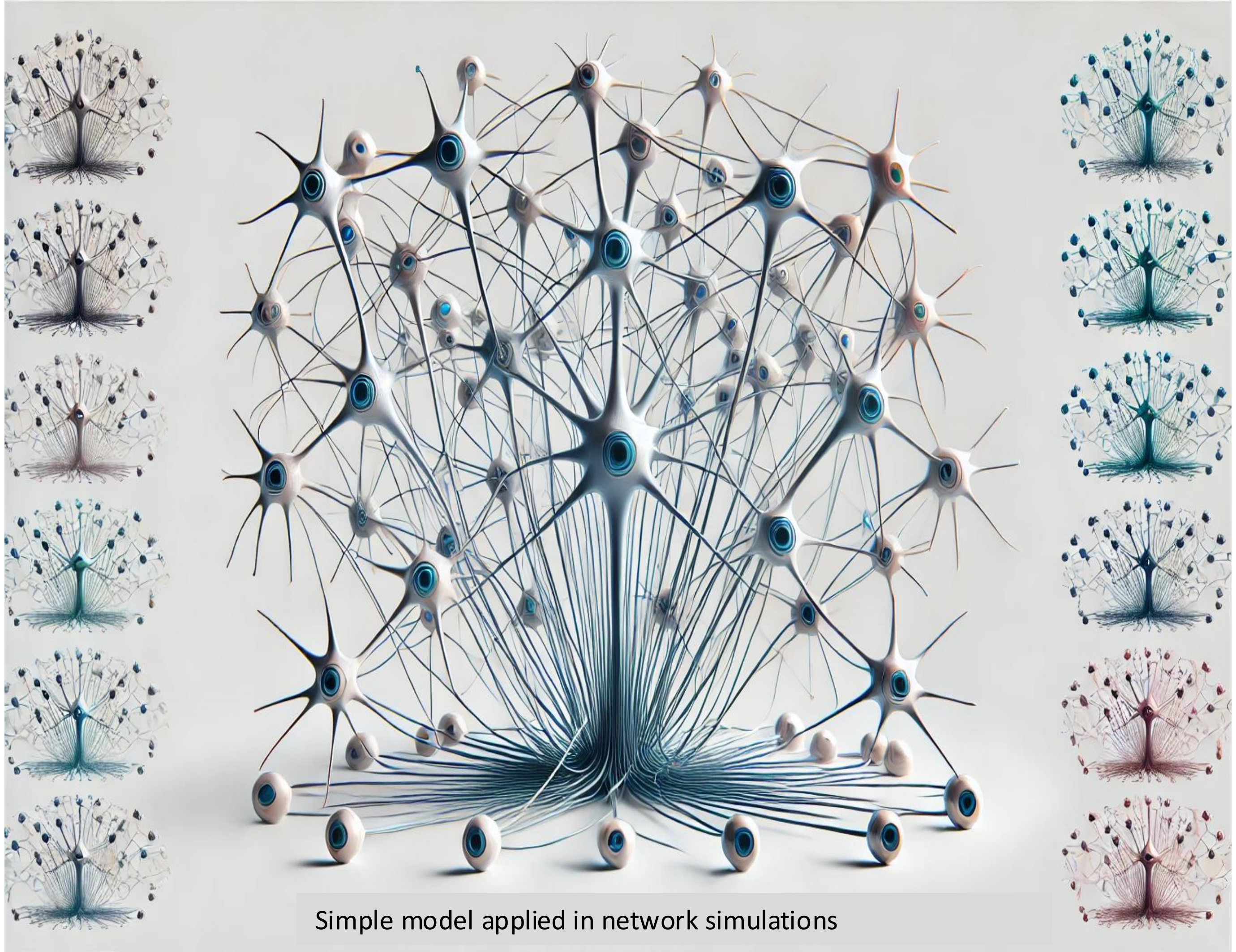
Key Components

1. **Membrane Potential V(t)**: It represents the voltage difference across the neuron’s membrane. The neuron’s membrane potential is affected by the input currents it receives from other neurons or external stimuli.
2. **Leak Term**: The membrane is “**leaky**”, meaning it loses some charge over time, like a capacitor with a resistor. This leak is proportional to the difference between the membrane potential and the resting potential, pulling the membrane potential back toward the resting state.
3. **Input Current I(t):** It is either the external or the synaptic input that drives the neuron’s membrane potential. If the input is strong enough, it can cause the membrane potential to rise.
4. **Threshold**: When the membrane potential reaches a certain threshold value, the neuron “fires”, producing an action potential (or spike). After firing, the membrane potential is reset to a lower value (often the resting potential or some reset value).
5. **Reset**: After each spike, the membrane potential is reset to a baseline or reset value, and the process starts again.



Why is the LIF Model Useful?

1. **Simple and Computationally Efficient**: The LIF model is widely used in computational neuroscience because of its simplicity and low computational cost compared to more complex neuron models like the Hodgkin-Huxley model.
2. **Captures Key Aspects of Neuronal Firing**: Despite being simple, the LIF model captures essential characteristics of neuronal behavior, such as the generation of spikes in response to inputs and the refractory period following a spike.
3. **Ideal for Networks**: The LIF model is often used in simulations of large networks of neurons to study collective dynamics like synchronization, oscillations and information processing in the brain.



Simple model applied in network simulations

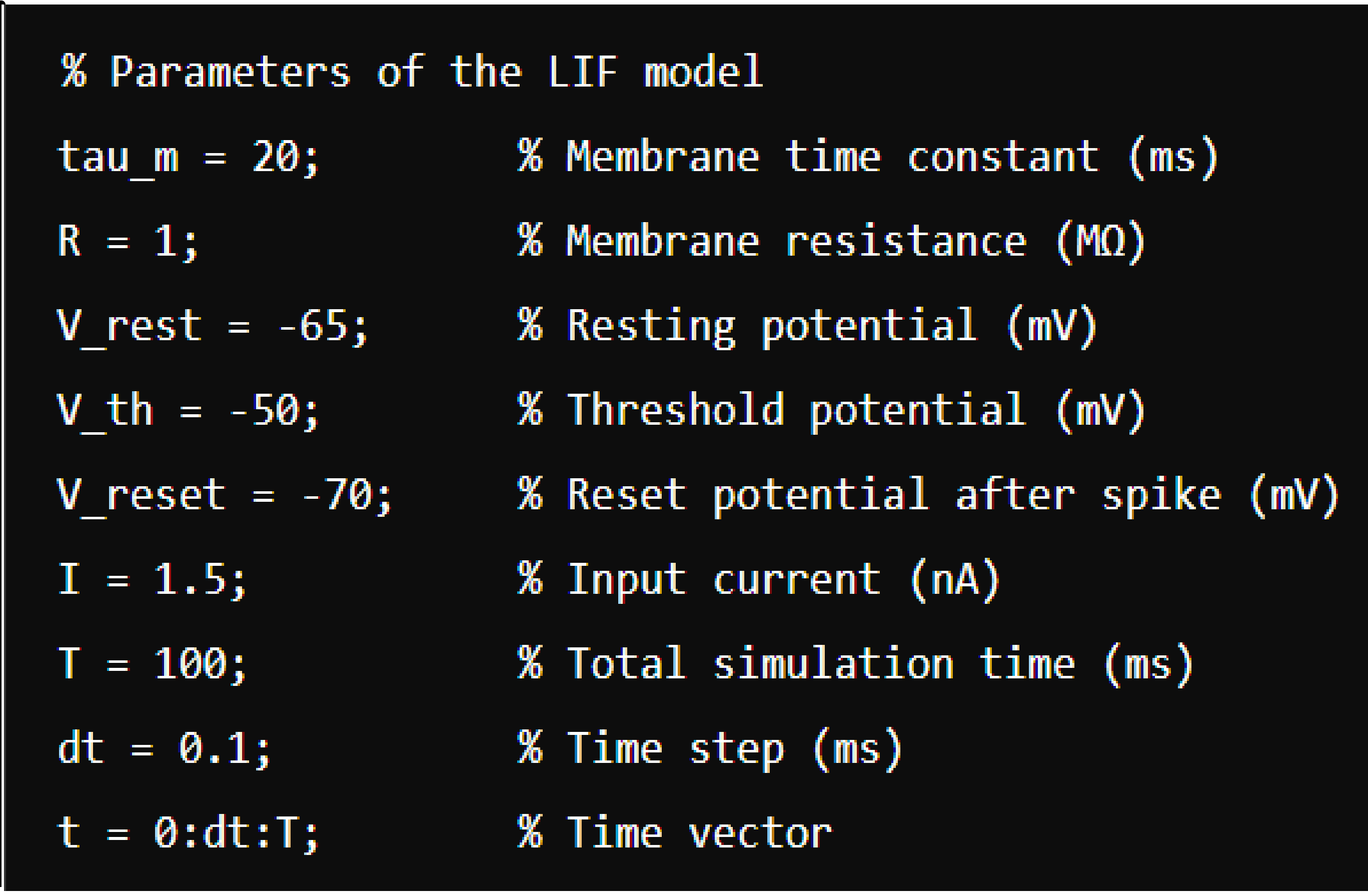
The Model Parameters (1)

Define Parameters:

1. Membrane time constant ($\boldsymbol{\tau_m}$)
2. Membrane resistance (**R**)
3. Resting membrane potential ($\mathbf{V_{rest}}$)
4. Threshold potential ($\mathbf{V_{th}}$)
5. Reset potential ($\mathbf{V_{reset}}$)
6. External Input Current (**I**)

Define the time vector

1. The simulation will iterate over small time steps
2. Compute changes in membrane potential in each step



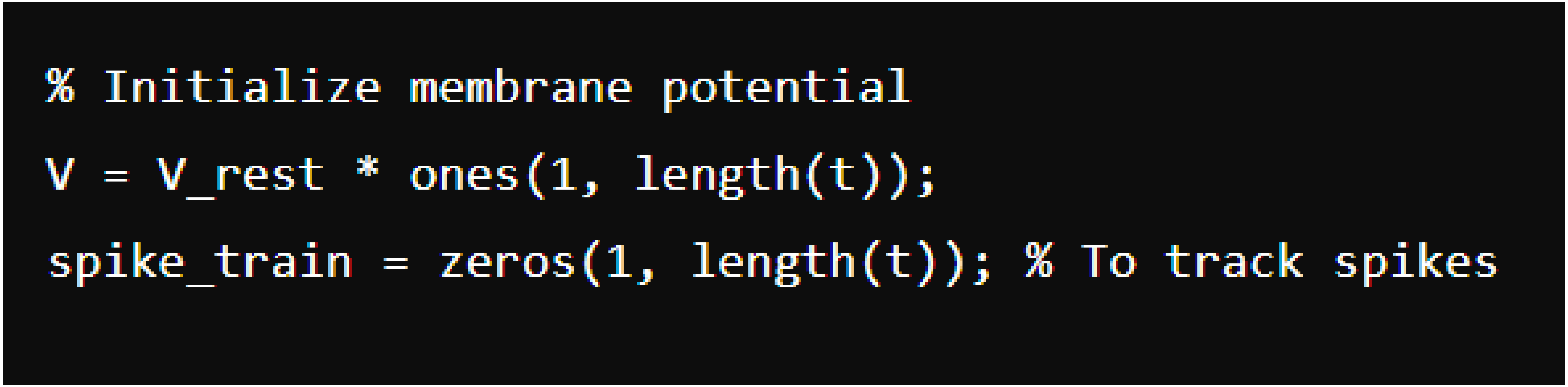
```
% Parameters of the LIF model
tau_m = 20;        % Membrane time constant (ms)
R = 1;             % Membrane resistance (MΩ)
V_rest = -65;      % Resting potential (mV)
V_th = -50;        % Threshold potential (mV)
V_reset = -70;     % Reset potential after spike (mV)
I = 1.5;           % Input current (nA)
T = 100;           % Total simulation time (ms)
dt = 0.1;          % Time step (ms)
t = 0:dt:T;        % Time vector
```

The Model Parameters (2)

- The **membrane time constant** (**τ_m**), typically in milliseconds, represents how fast the membrane potential responds to the input current. A higher value means the neuron is slower to respond.
- The **membrane resistance** (**R**) defines how much the neuron membrane resists the flow of electric current. It’s set to 1 MΩ here.
- The **resting potential** (**V_{rest}**) is the **baseline membrane potential** of the neuron when it is not receiving any input. In this case, it is set to -65 mV.
- When the membrane potential reaches the **threshold potential** value (**V_{th}**), the neuron will fire a spike. In this model, it is set to -50 mV.
- Once the neuron fires a spike, the membrane potential is reset to a lower (**reset potential after spike**) value (**V_{reset}**). Here, it resets to -70 mV.
- The **input current** (**I**), measured in nanoamperes (nA) is the constant input current applied to the neuron.
- The **total simulation time T** (time duration) is set to 100 ms.
- The **time step (dt)** used in the simulation, represents how much time advances in each iteration of the loop. It is set to 0.1 milliseconds.
- The **time vector (t)** represents the simulation time points, ranging from 0 to T (100 ms) with a step size of dt (0.1 ms).

Model Initialization

- The array (**V**) stores the membrane potential values at each time step. Initially, it is set to the resting potential $\mathbf{V_{rest}}$ at all time points.
- The array ‘**spike_train**’ keeps track of when spikes (action potentials) occur. It starts with all zeros and will store a ‘**1**’ at time points where spikes occur.



```
% Initialize membrane potential
V = V_rest * ones(1, length(t));
spike_train = zeros(1, length(t)); % To track spikes
```

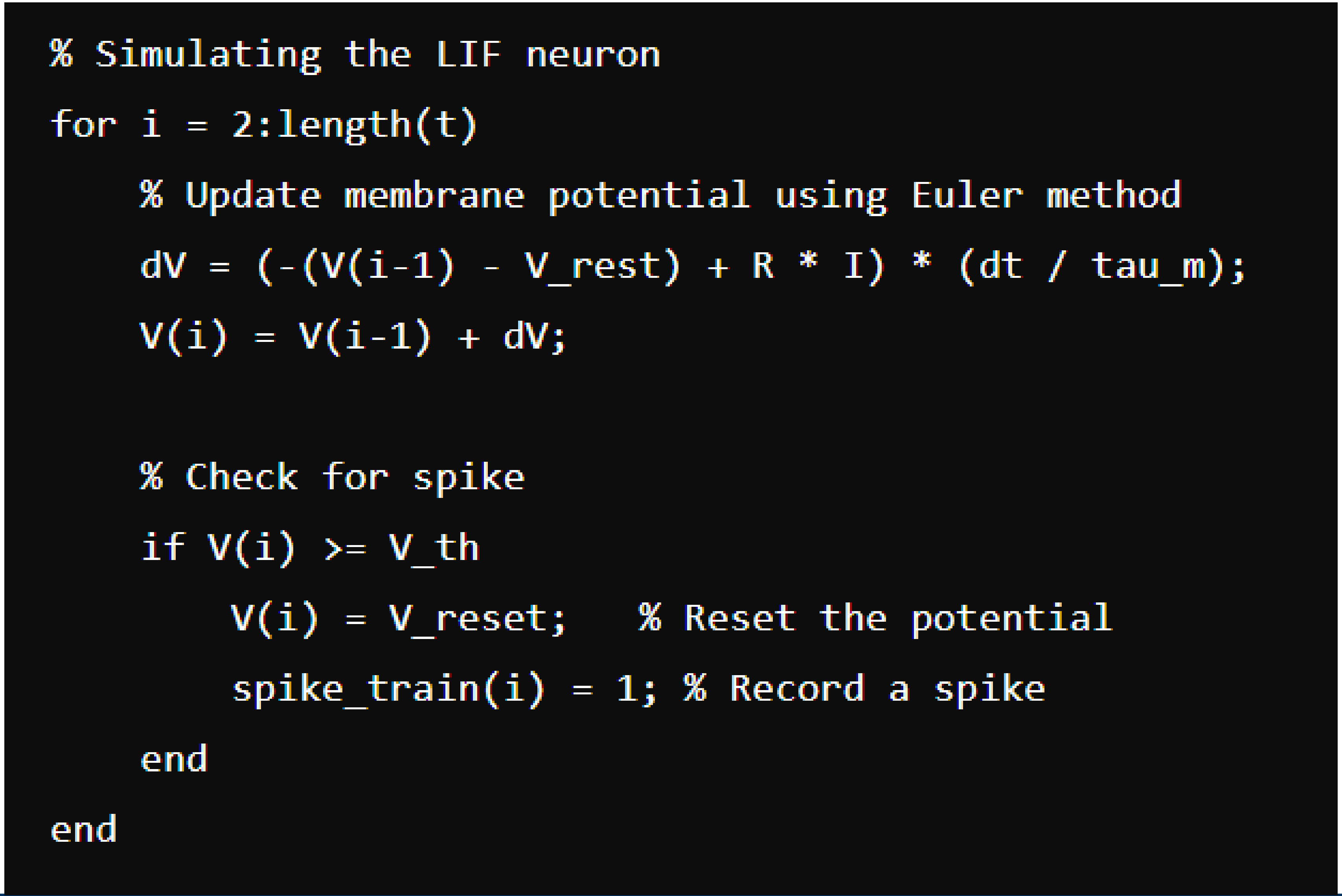
Simulation Loop

- This part of the code runs the simulation for the entire time period.
- **Euler method for membrane potential update**: The LIF model describes how the membrane potential changes over time.
- In each iteration of the loop, the change in membrane potential dV is calculated using the equation:

$$dV = \left(-\frac{(V(i-1) - V_{rest})}{\tau_m} + \frac{R \cdot I}{\tau_m} \right) \cdot dt$$

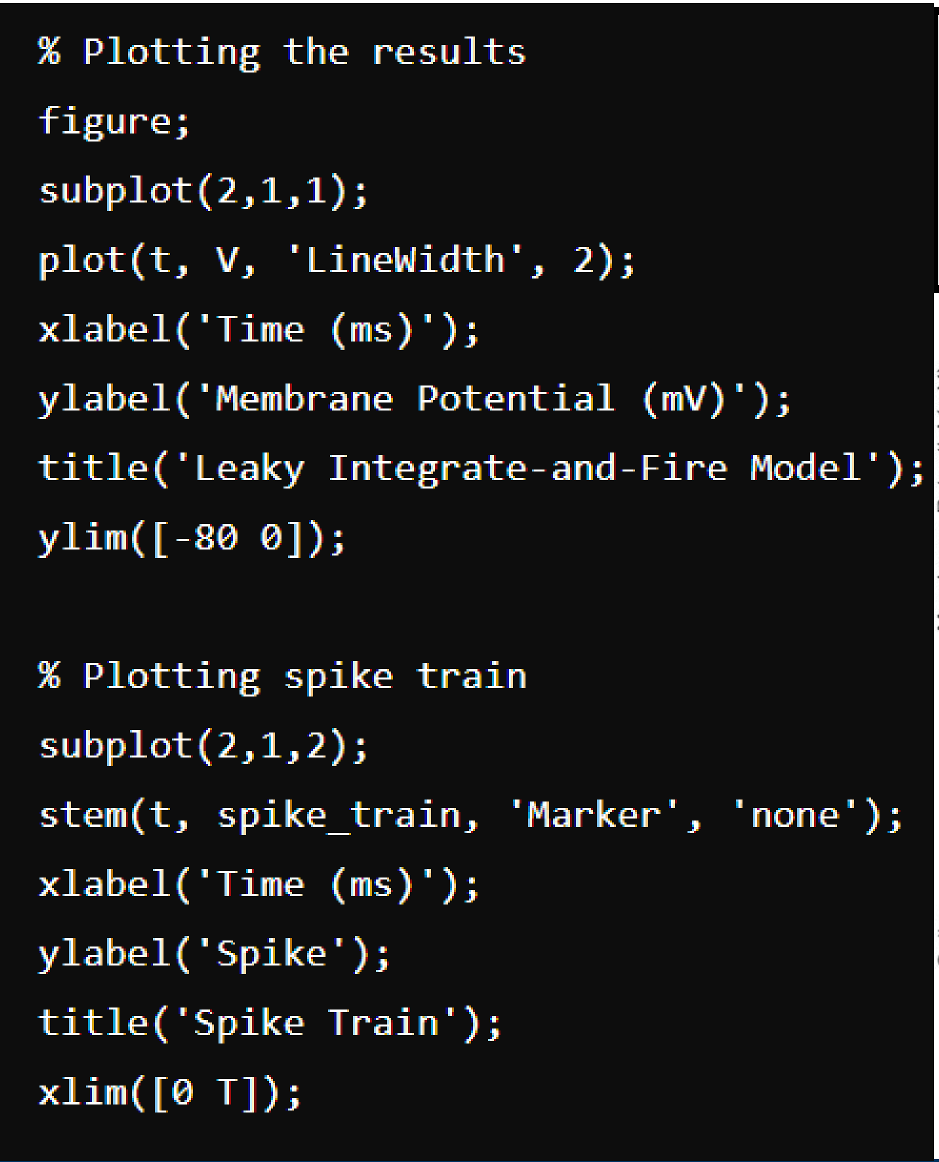
- This is a simplified form of the LIF equation where the neuron integrates the input current and decays back to the resting potential when no current is applied.
- If the membrane potential exceeds the threshold $\mathbf{V_{th}}$, a spike is generated.
- The membrane potential is then reset to $\mathbf{V_{reset}}$ and a spike is recorded in ‘**spike_train**‘ through the insertion of **1**’

Simulation Loop: The code



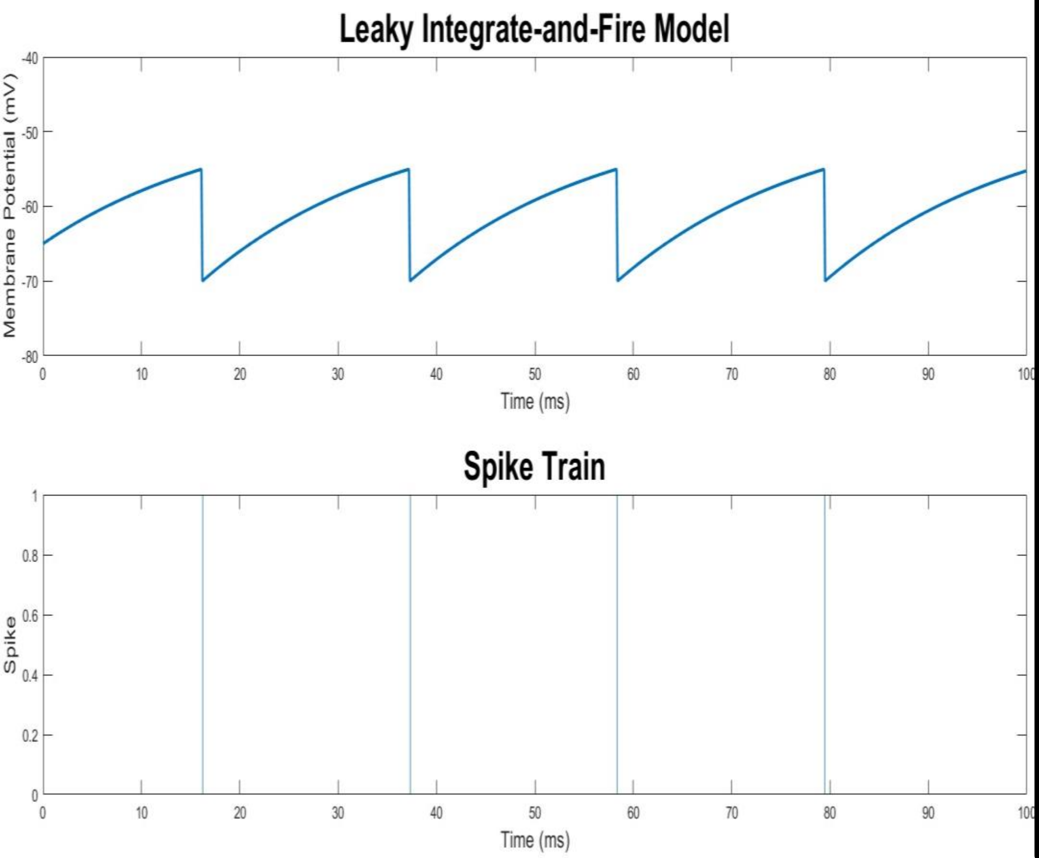
```
% Simulating the LIF neuron
for i = 2:length(t)
    % Update membrane potential using Euler method
    dV = (-(V(i-1) - V_rest) + R * I) * (dt / tau_m);
    V(i) = V(i-1) + dV;

    % Check for spike
    if V(i) >= V_th
        V(i) = V_reset;   % Reset the potential
        spike_train(i) = 1; % Record a spike
    end
end
```

Visualization



```
% Plotting the results
figure;
subplot(2,1,1);
plot(t, V, 'LineWidth', 2);
xlabel('Time (ms)');
ylabel('Membrane Potential (mV)');
title('Leaky Integrate-and-Fire Model');
ylim([-80 0]);

% Plotting spike train
subplot(2,1,2);
stem(t, spike_train, 'Marker', 'none');
xlabel('Time (ms)');
ylabel('Spike');
title('Spike Train');
xlim([0 T]);
```

- The membrane potential is plotted over time, showing how it evolves under the input current and spikes at the threshold.
- The **spike train plot** shows the spike times (marked by vertical lines) as a function of time.



- The membrane potential will increase with time due to the input current.
- When the potential reaches the threshold, the neuron will “fire” (spike) and the potential will reset to the lower value, before beginning to rise again.

Refractory Period

- A refractory period is the time after a spike during which a neuron cannot fire again.
- To implement this, the membrane potential is held constant for a short period after a spike.
- We will introduce a refractory period parameter, track the time after each spike and prevent the neuron from spiking again within that time.
- The ‘**refractory_period**’ parameter represents the time (in milliseconds) during which the neuron **cannot fire** after a spike.
- The ‘**ref_time**’ variable tracks how long the neuron has been in the refractory state.
- If the neuron is in the refractory period (**ref_time > 0**), the membrane potential is held at the reset potential and no spiking is allowed.
- The refractory period decreases over time and eventually expires, allowing the neuron to fire again.
- During the simulation we still see spikes at regular intervals, but now there is a **pause** (refractory period) after each spike before the neuron can fire again.

Refractory Period : Code Explanation

- The ‘**refractory_period**’ variable sets the duration of the refractory period in milliseconds. In this example, it is set to 5 ms.
- The ‘**ref_time**’ variable keeps track of how much time is left in the refractory period after the neuron fires a spike.
- When ‘**ref_time**’ is greater than zero, the neuron is in the refractory state.
- If the neuron is in the refractory state (**ref_time > 0**), the membrane potential is held at the reset potential (**V_reset**).
- The refractory period timer (**ref_time**) is decremented by **dt** (0.1 ms) on each iteration.
- Once **ref_time** reaches ‘**0**’, the neuron can start firing again.
- If the neuron is not in the refractory period (**else block**), the membrane potential is updated using the Euler method:

$$dV = \left(-\frac{(V(i-1) - V_{\text{rest}})}{\tau_m} + \frac{R \cdot I}{\tau_m} \right) \cdot dt$$

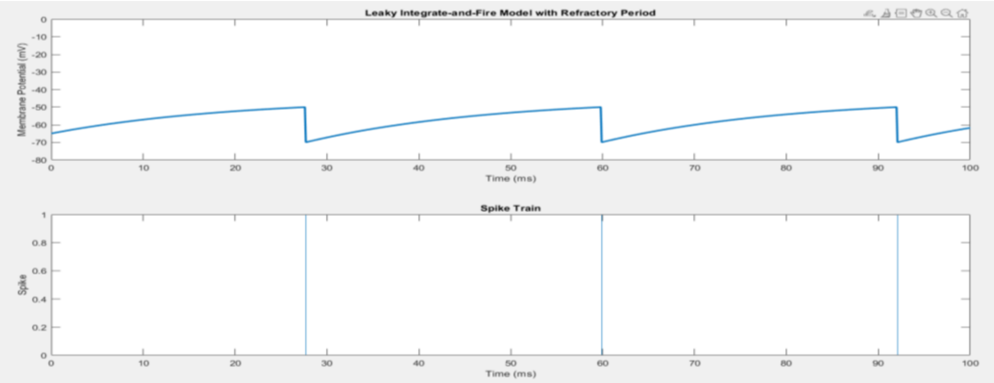
- This equation integrates the input current to determine how the membrane potential changes.
- If the membrane potential crosses the threshold (**V_th**), the neuron fires a spike:
  - ❑ The membrane potential is set to **V_reset**.
  - ❑ A spike is recorded in **spike_train** by setting the current index to 1.
  - ❑ The refractory timer (ref_time) is set to the duration of the refractory period (10 ms).

Key Concepts in Refractory Period Simulation

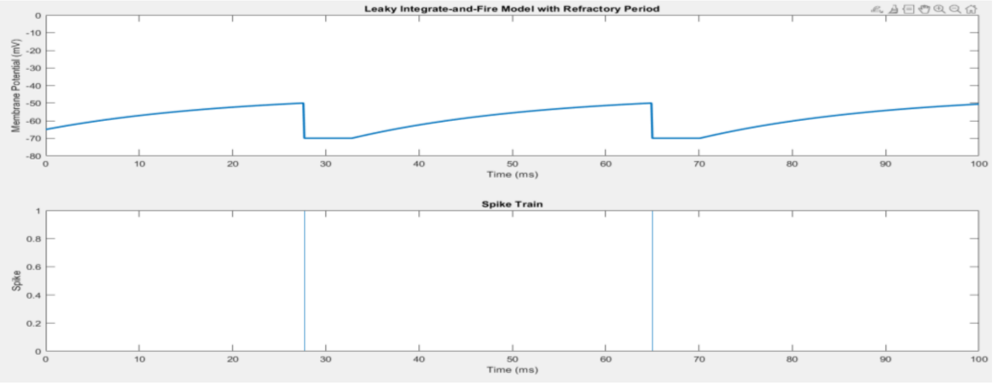
- After the neuron fires a spike, the membrane potential is held at **V_reset** for the duration of the refractory period (10 ms).
- During this time, the neuron **cannot fire** another spike, and the membrane potential does not change.
- The variable '**ref_time**' counts down the refractory period.
- When it reaches '**0**', the neuron resumes normal behavior (integration of the input current).
- When the membrane potential exceeds the threshold **V_th**, the neuron fires a spike.
- Then, the potential is immediately reset to **V_reset**.
- At this point, the refractory period starts (**ref_time = 10 ms**), preventing further spiking for the next 10 ms.
- While the neuron is not in the refractory state, its membrane potential is governed by the input current and the membrane's leaky properties (**decay back toward V_rest**).
- This results in the gradual rise of the membrane potential until it reaches the threshold, at which point a spike is generated.

The effect of absolute refractory period

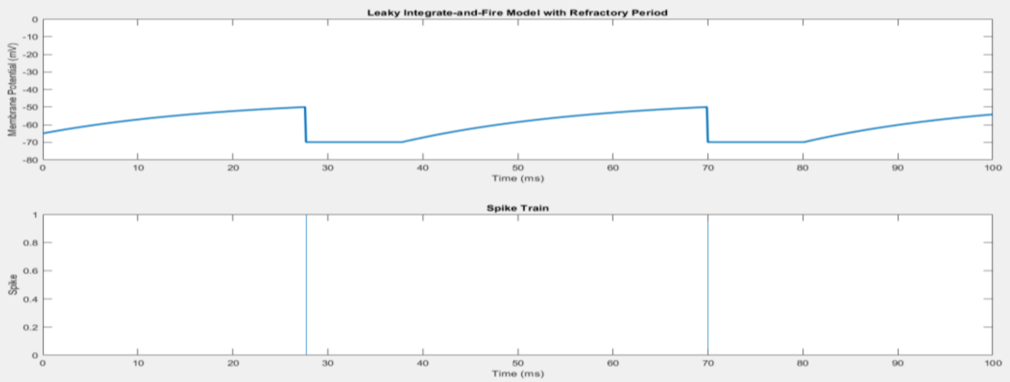
No Refractory Period

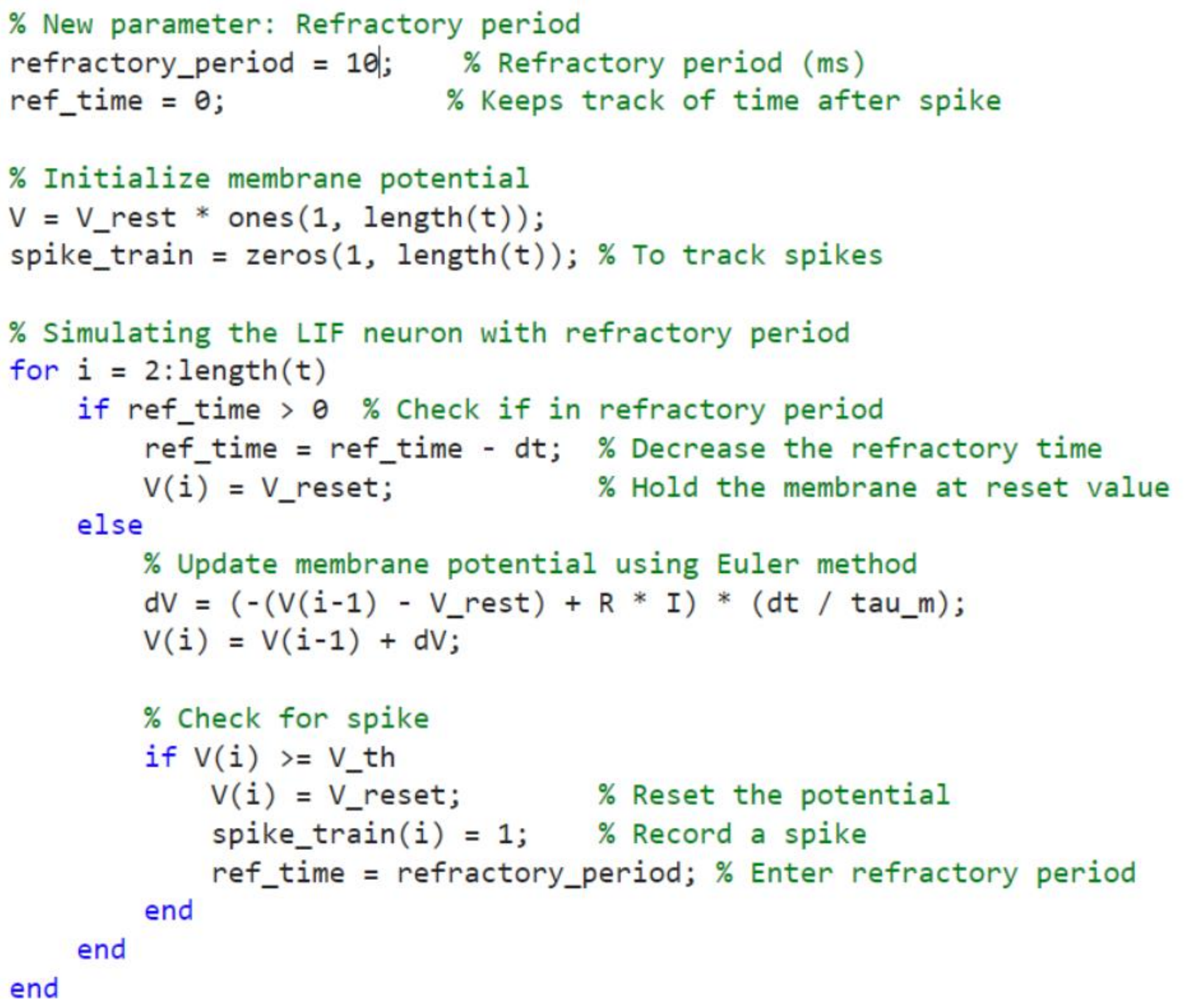


Refractory Period : 5 ms



Refractory Period : 10 ms

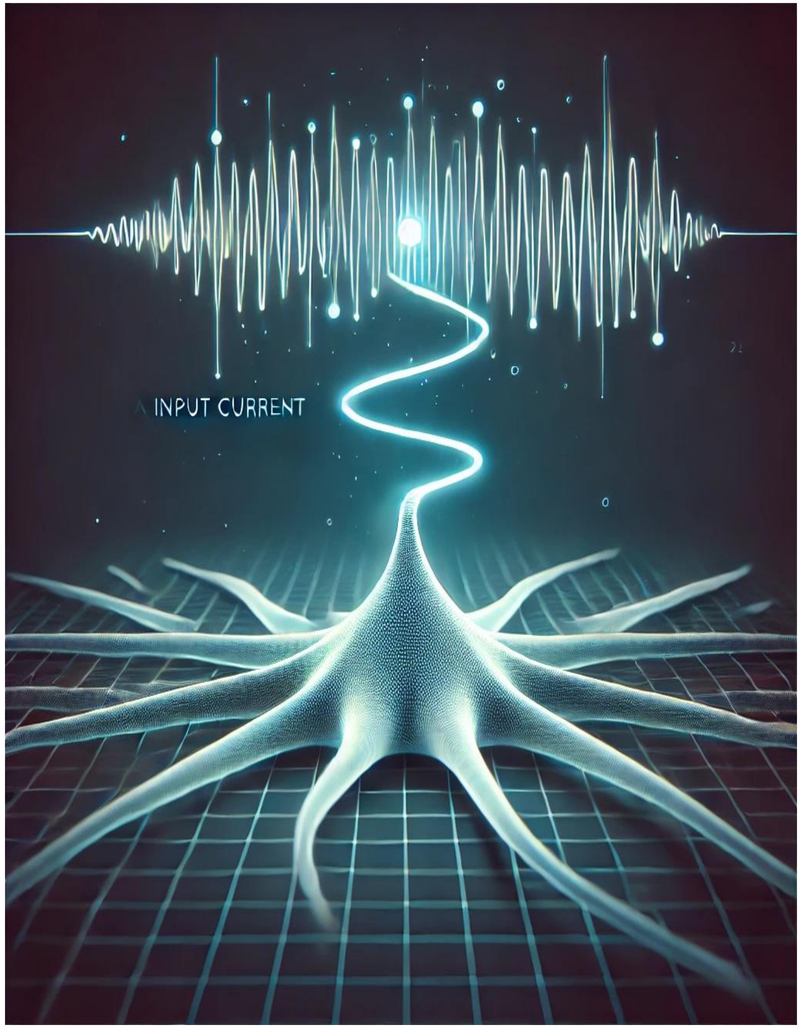


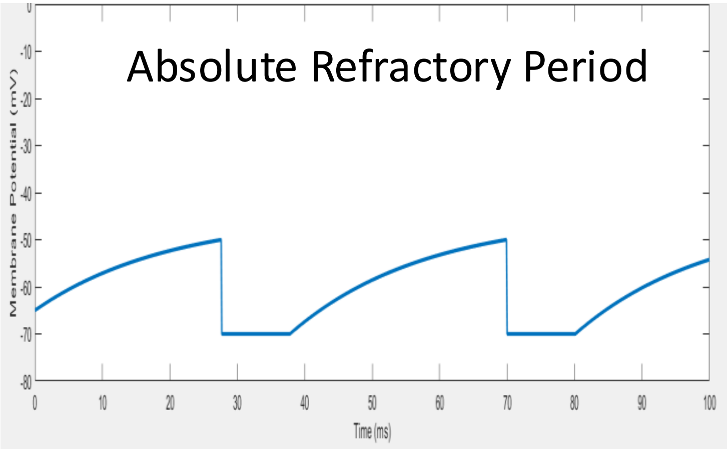


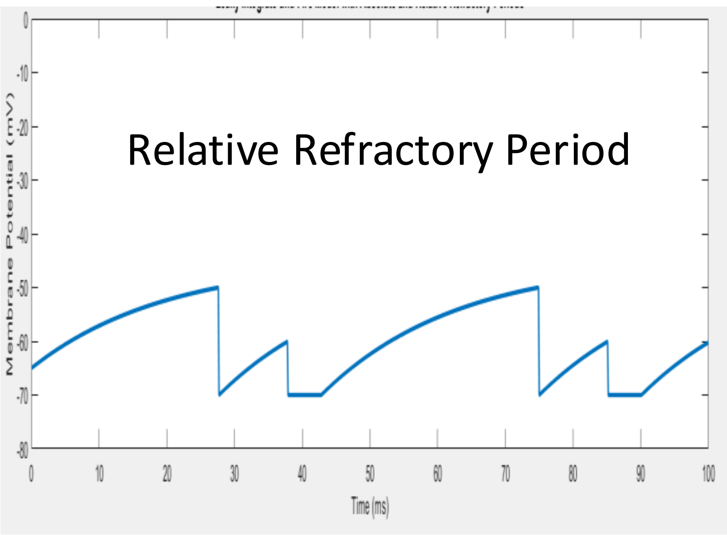
```
% New parameter: Refractory period
refractory_period = 10;    % Refractory period (ms)
ref_time = 0;             % Keeps track of time after spike

% Initialize membrane potential
V = V_rest * ones(1, length(t));
spike_train = zeros(1, length(t)); % To track spikes

% Simulating the LIF neuron with refractory period
for i = 2:length(t)
    if ref_time > 0  % Check if in refractory period
        ref_time = ref_time - dt;  % Decrease the refractory time
        V(i) = V_reset;            % Hold the membrane at reset value
    else
        % Update membrane potential using Euler method
        dV = (-(V(i-1) - V_rest) + R * I) * (dt / tau_m);
        V(i) = V(i-1) + dV;

        % Check for spike
        if V(i) >= V_th
            V(i) = V_reset;          % Reset the potential
            spike_train(i) = 1;      % Record a spike
            ref_time = refractory_period; % Enter refractory period
        end
    end
end
```

The Relative Refractory Period

- After the absolute refractory period, there is a phase where the neuron can fire another spike, but it requires a stronger stimulus (i.e. the input current must be larger).
- The neuron is less excitable but not completely unresponsive.
- We implement this in code by introducing two time-windows:
  - ❑ The duration during which the neuron cannot fire any spikes (‘**absolute_refractory_period**’).
  - ❑ The time following the absolute refractory period when the neuron can still spike, but its threshold is increased temporarily (‘**relative_refractory_period**’).




Absolute Refractory Period


Relative Refractory Period

Refractory Period Parameters

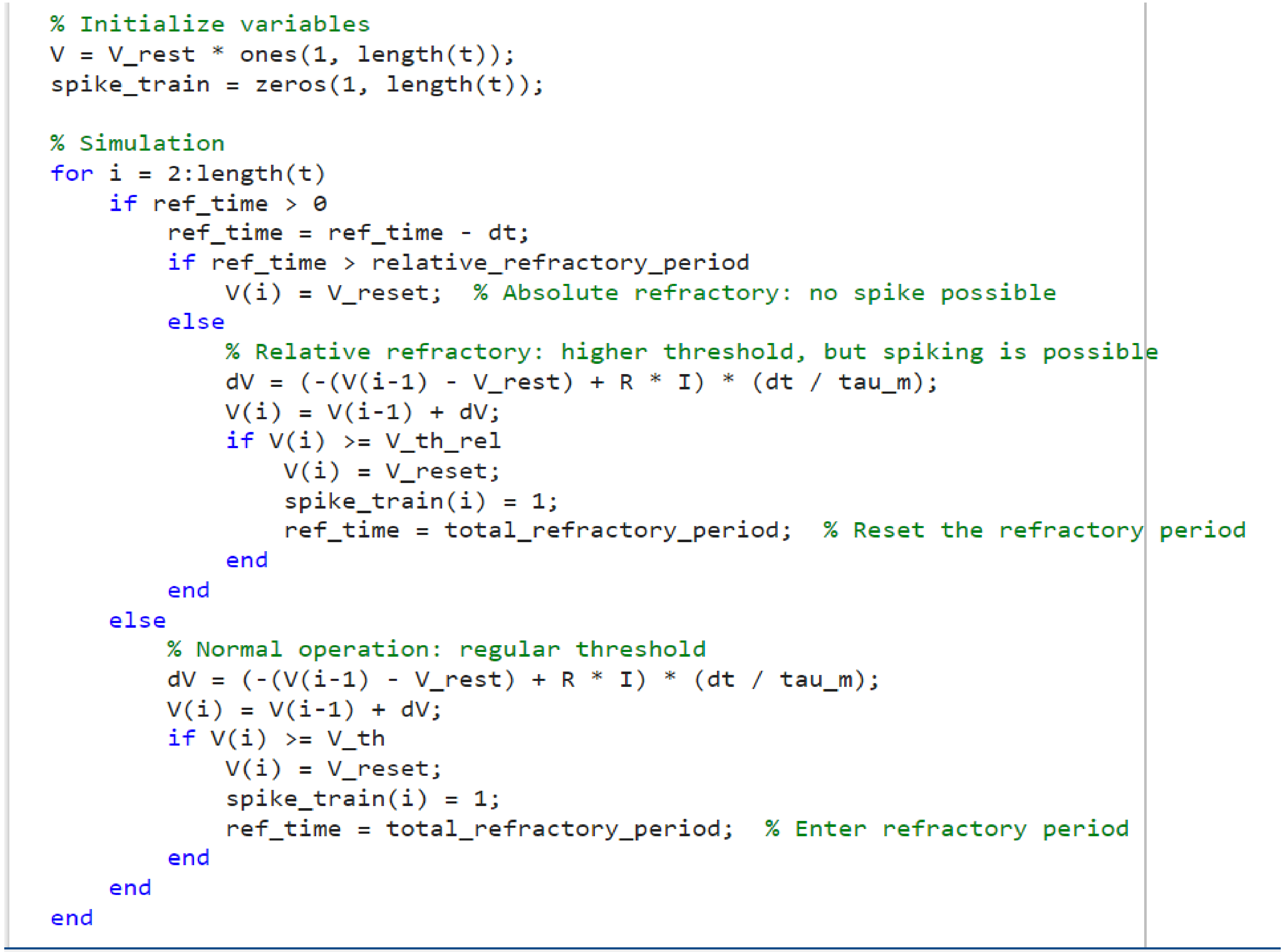
- The ‘**absolute_refractory_period**’ is the time (in ms) during which the neuron cannot fire any spike, no matter how strong the input is. This is set to 5 ms.
- The ‘**relative_refractory_period**’ is the time (in ms) after the absolute refractory period, during which the neuron can fire, but it is harder to do so. The threshold is temporarily increased. This is set to 500 ms. This unusually long duration is set to ensure that spikes can occur during the higher threshold (relative refractory) period.
- The ‘**ref_time**’ is a timer variable to track the elapsed time since the last spike guiding refractory period.
- The normal threshold potential ‘**V_th**’ is set to -50 mV. The neuron generates a spike when its membrane potential exceeds this threshold under normal conditions.
- The higher threshold potential ‘**V_th_rel**’ during the relative refractory period is set to -40 mV. This is a more stringent condition that the neuron must meet to generate a spike during the relative refractory period.

Loop Simulation Code (1)

1. **Loop Initialization ‘for i=2:length(t)’:** This loop iterates over each time step starting from the second element (to allow referencing the previous time step).
2. The ‘**if ref_time>0**’ checks if the neuron is in a refractory period (**ref_time** being a countdown from the last spike)
3. The ‘**ref_time = ref_time – dt;**’ decreases the refractory timers by the time step, moving closer to the end of the refractory period.
4. The ‘**if ref_time > relative_refractory_period**’ checks if the current ‘**ref_time**’ is still within the absolute refractory period (where no spikes can occur at all).
5. The ‘**V(i) = V_reset**’ sets the membrane potential to **V_reset**, clamping it during absolute refractoriness.
6. The ‘else’ block executes if the neuron is in the relative refractory period.
7. The ‘**dV = (-(V(i-1)-V_rest + R * I) * (dt/tau_m);**’ calculates the change in membrane potential ‘**dV**’ using the leaky integrate-and-fire formula, which considers the resting potential, input current and membrane resistance.
8. The ‘**V(i) = V(i-1) + dV;**’ updates the membrane potential by adding ‘**dV**’ to the previous potential.
9. The ‘**if V(i) >= V_th_rel**’ checks if the updated potential reaches or exceeds the higher threshold set for the relative refractory period.
10. The ‘**V(i) = V_reset;**’ resets the potential to **V_reset** after a spike.
11. The ‘**spike_train(i)=1**’ records a spike in the spike train.
12. The ‘**ref_time = total_refractory_period;**’ resets the refractory timer to the total duration (absolute + relative).

Loop Simulation Code (2)

- The second (final) ‘**else**’ block handles the normal operation when not in any refractory period.
- The **if-condition** is similar to the relative period but checks against the normal spiking threshold ‘**V_th**’.
- It resets and records spikes as before but using the normal threshold.
- The loop is run vectorized over the time array ‘**t**’, ensuring efficient computation.



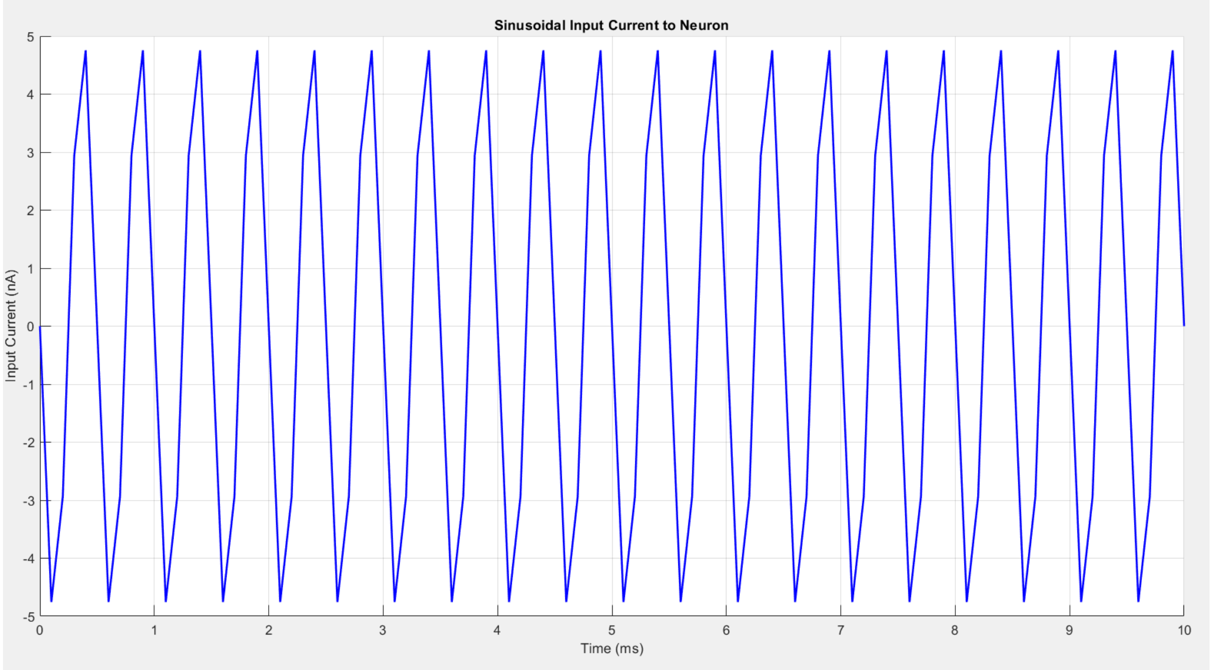
```
% Initialize variables
V = V_rest * ones(1, length(t));
spike_train = zeros(1, length(t));

% Simulation
for i = 2:length(t)
    if ref_time > 0
        ref_time = ref_time - dt;
        if ref_time > relative_refractory_period
            V(i) = V_reset;  % Absolute refractory: no spike possible
        else
            % Relative refractory: higher threshold, but spiking is possible
            dV = (-(V(i-1) - V_rest) + R * I) * (dt / tau_m);
            V(i) = V(i-1) + dV;
            if V(i) >= V_th_rel
                V(i) = V_reset;
                spike_train(i) = 1;
                ref_time = total_refractory_period;  % Reset the refractory period
            end
        end
    else
        % Normal operation: regular threshold
        dV = (-(V(i-1) - V_rest) + R * I) * (dt / tau_m);
        V(i) = V(i-1) + dV;
        if V(i) >= V_th
            V(i) = V_reset;
            spike_train(i) = 1;
            ref_time = total_refractory_period;  % Enter refractory period
        end
    end
end
```

Refractory Period Summary

- During absolute refractoriness, the neuron’s potential is clamped, preventing any change regardless of incoming current.
- During relative refractoriness, the neuron can still integrate incoming signals but requires a higher threshold to fire again, reflecting a decreased sensitivity post-spike.
- Outside refractory periods, normal integration and spiking behavior resume based on the regular threshold.

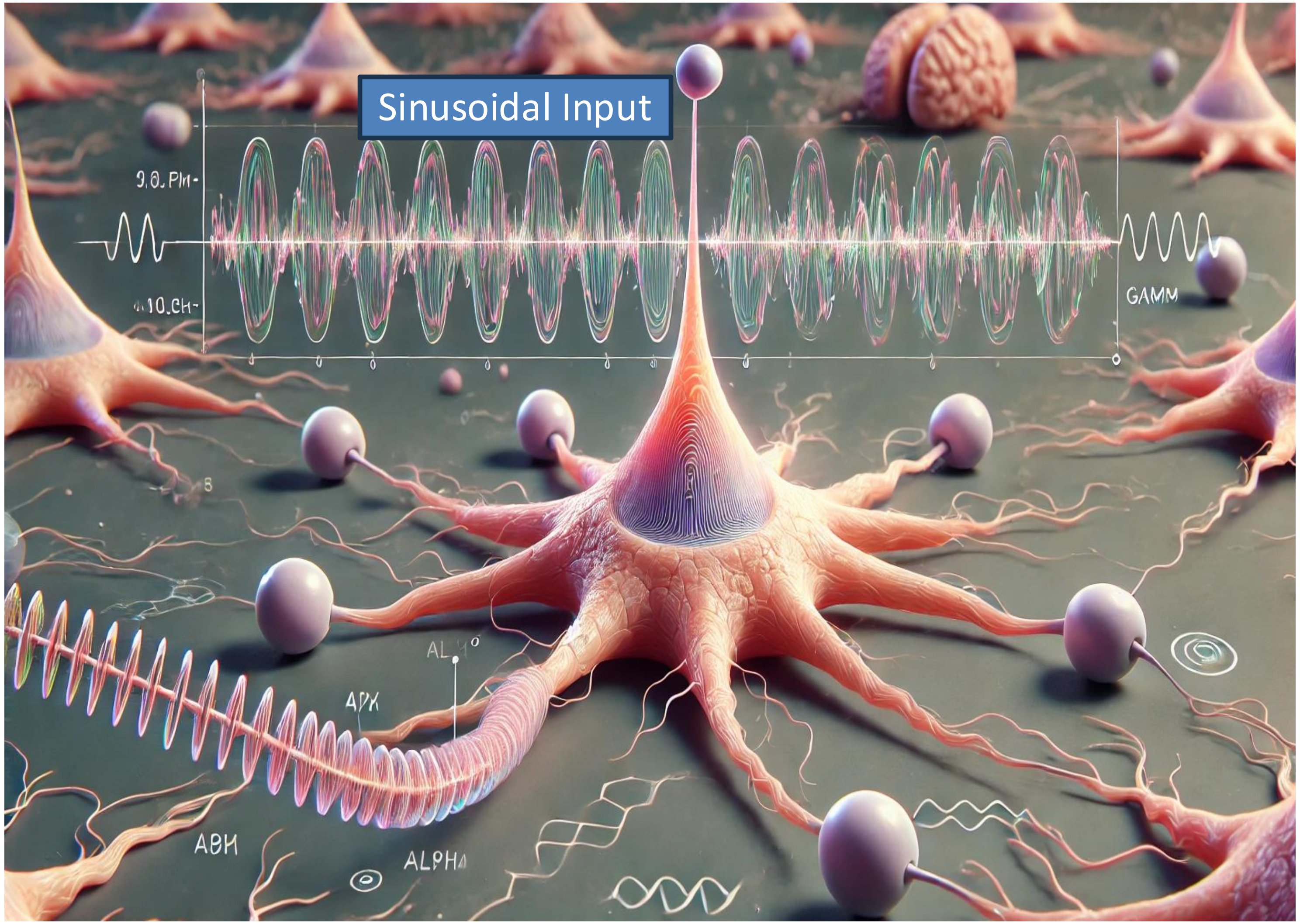
Modeling Variable Input Current

- Instead of using a constant current, we will apply a time-varying input current.
- This allows us to explore how the membrane potential responds to more realistic or dynamic stimuli, such as:
  1. A sinusoidal input current (mimicking oscillatory input).
  2. A random input current (mimicking noisy synaptic input from other neurons).



Sinusoidal Input Current

- It is often used in computational neuroscience to model oscillatory input to a neuron.
- It mimics rhythmic signals that the brain may receive or produce.
- These oscillatory inputs are a way to study how neurons respond to periodic stimuli.
- The oscillatory inputs are relevant to neural behavior and brain rhythms associated with cognition, sensory processing and motor control.

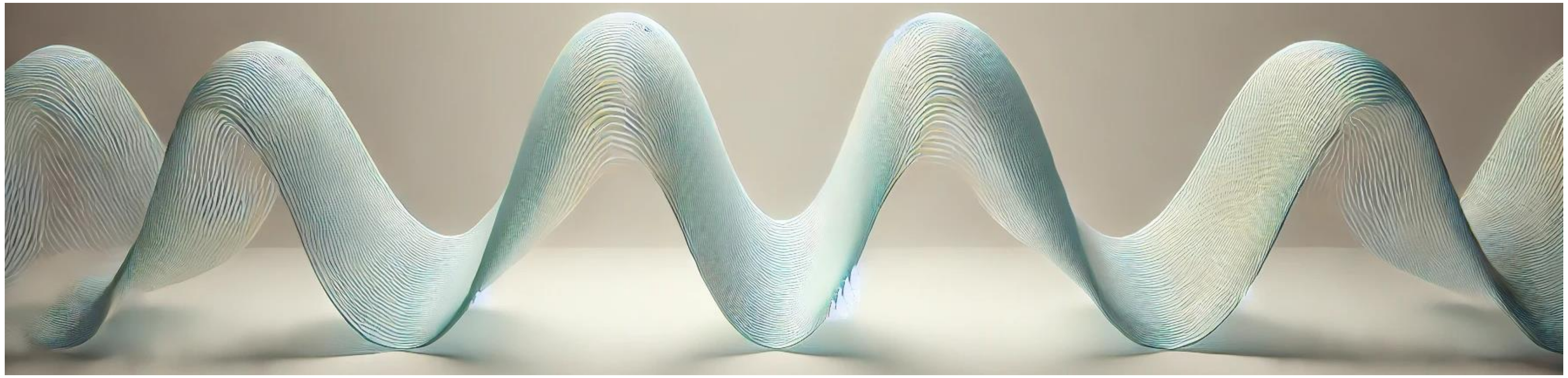


Sinusoidal Input

Key Aspects of Sinusoidal Input in Neurons (1)

- A sinusoidal input current can be described mathematically as: $I(t) = I_0 \cdot \sin(2\pi f t + \phi)$
    - ❑ I_0 is the amplitude of the input current (strength of the input)
    - ❑ F is the frequency of oscillation, measured in Hz (cycles per second)
    - ❑ ϕ is the phase offset (initial phase shift of the oscillation)
    - ❑ t is time
- The neuron’s response to sinusoidal input can vary.
- It depends on the frequency and amplitude of the input, as well as the properties of the neuron (membrane time constant, ion channel dynamics).

Amplitude over time

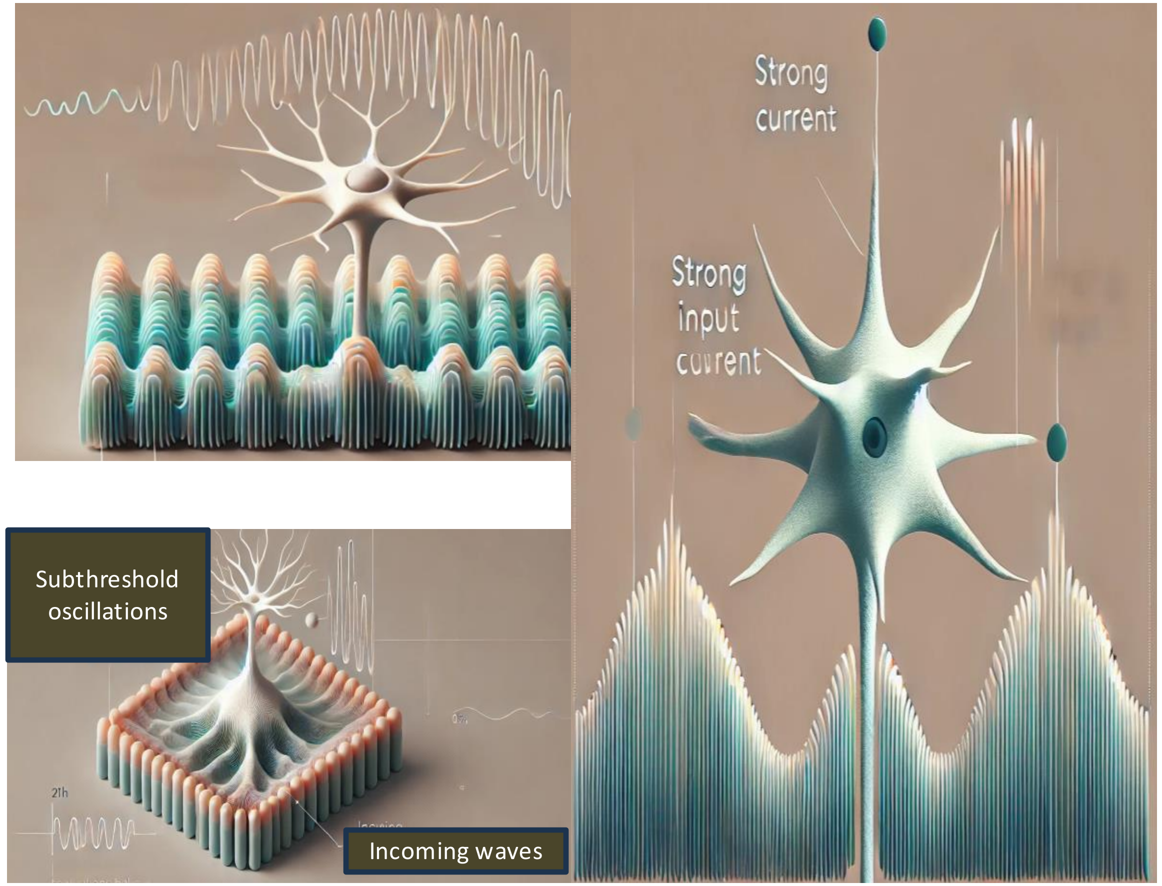


Key Aspects of Sinusoidal Input in Neurons (2)

- Sinusoidal currents can be injected to study how neurons encode oscillatory input.
- This can reveal phase locking or frequency selectivity in the neuronal response.
- Sinusoidal inputs are often used to explore how networks of neurons synchronize or how single neurons behave in response to oscillatory stimuli, such as cortical gamma oscillations.
- By varying the frequency f, one can determine how the neuron responds to different oscillatory inputs.
- This provides insights into the neuron’s frequency tuning and how it processes time-varying signals.
- A neuron might respond preferentially to certain frequency ranges (low-pass, high-pass, band-pass or band-stop behavior).
- Examples in real neural systems are the theta rhythms (4-8 Hz) in the hippocampus and gamma rhythms (30 – 100 Hz) in the cortex.
- Modeling neurons with sinusoidal inputs at these frequencies can help researchers understand the neural mechanisms underlying these rhythms.

Key Phenomena

1. **Resonance:** Some neurons exhibit resonance, meaning they are more responsive to inputs at certain frequencies. This occurs when the input frequency matches the neuron’s natural oscillatory properties.
2. **Subthreshold Oscillations**: At certain input frequencies and amplitudes, the neuron may display subthreshold oscillations without firing action potentials, reflecting the membrane potential’s rhythmic fluctuations in response to the input.
3. **Action Potential Firing**: If the input current is strong enough, it can drive the neuron to spike rhythmically, potentially locking the spiking frequency to the input frequency (or some harmonic of it).



Subthreshold oscillations

Incoming waves

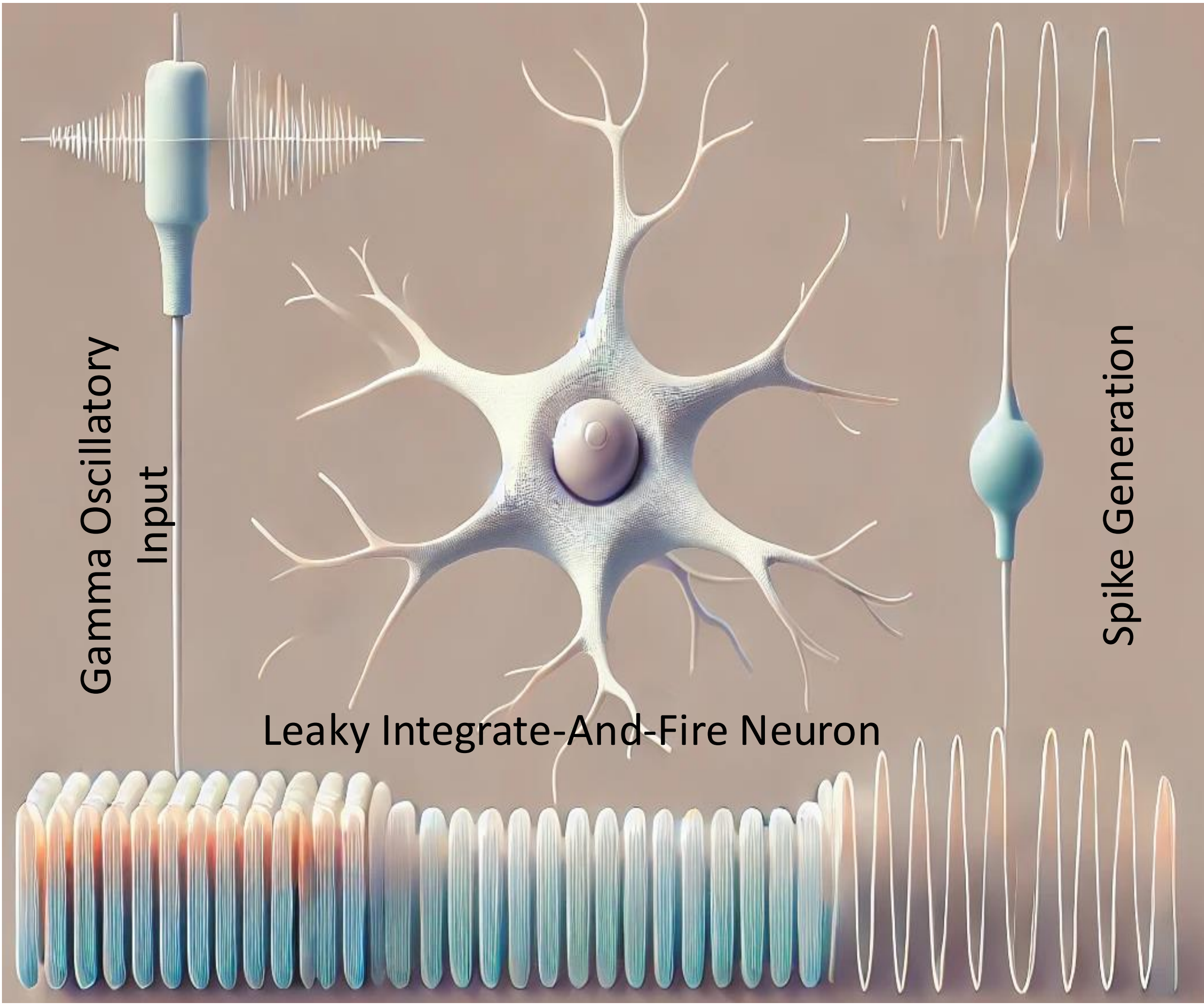
The high frequency effect

Why high frequency reduces spiking?

- The neuron integrates the input over time.
- When the input oscillates at a higher frequency, the membrane potential doesn’t have enough time to accumulate the input current before the oscillatory signal starts to decline again.
- For low frequency inputs, the oscillations are slower, so the neuron has more time to integrate the input and thus can more easily reach the threshold.
- In contrast, higher-frequency inputs rise and fall more quickly, giving the neuron less time to accumulate the input, resulting in fewer spikes.
- Noise can help boost the neuron’s membrane potential when the input current is oscillating rapidly.
- As the frequency increases, even noise may not be enough to bring the neuron to threshold consistently.

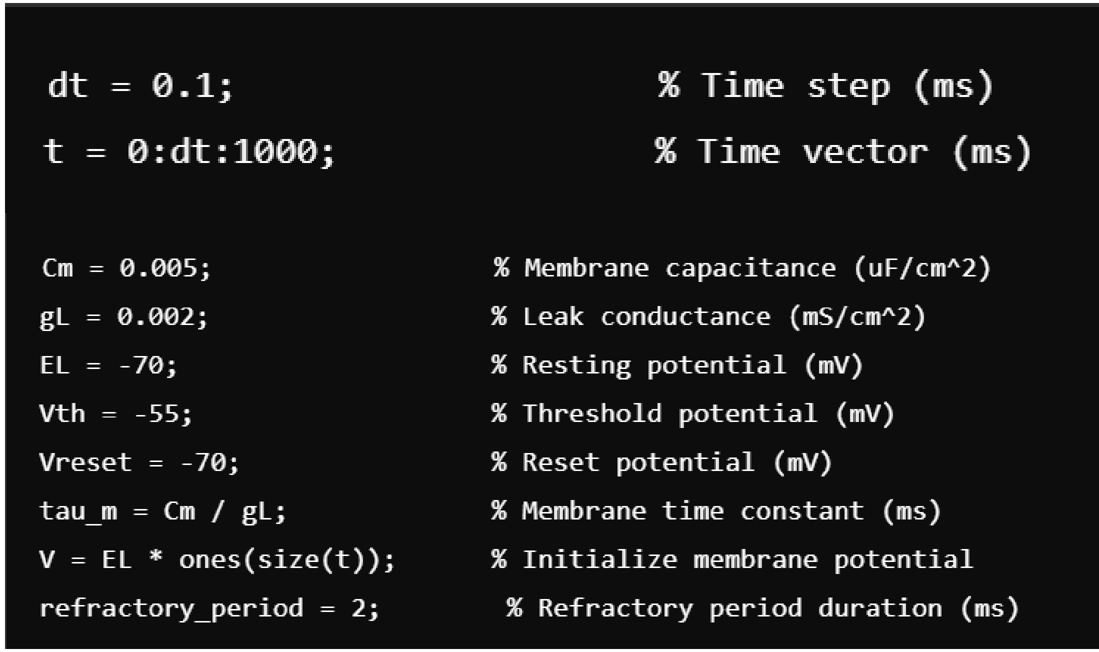
Model Analysis - Overview

- Analysis of a flexible model for studying the dynamics of a leaky integrate-and-fire neuron in response to oscillatory and noisy inputs.
- By adjusting the membrane properties, input amplitude, noise and frequency, this model can be used to explore a wide range of neuronal behaviors.
- This model helps explore how neurons respond to rhythmic input.
- We apply the model for the gamma frequency range (30 – 100 Hz), which is associated with cognitive processes like attention and memory.



The Model Parameters

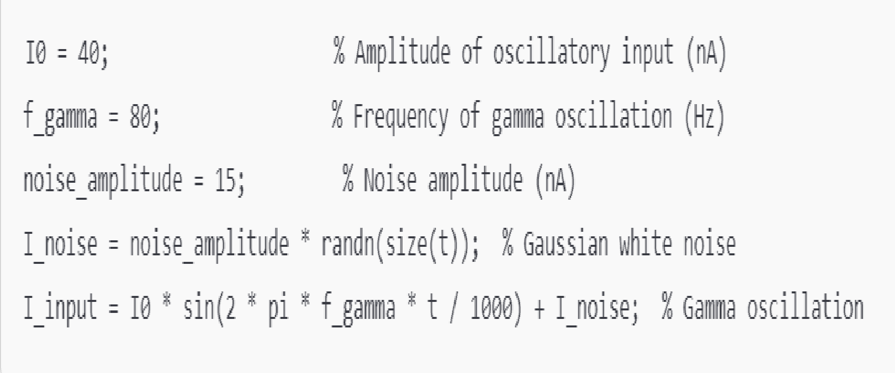
- The simulation runs with a time step of 0.1 milliseconds for a total duration of 1000 ms (1 second).
- This provides a high temporal resolution for modeling neural activity.
- The **membrane capacitance** $\mathbf{C_m = 0.005\ \mu F/cm^2}$ governs how much charge the neuron can hold, affecting how quickly it integrates input.
- The **leak conductance** $\mathbf{g_L = 0.002\ mS/cm^2}$ determines the rate at which the membrane potential leaks back to the resting potential E_L=-70 mV.
- The neuron spikes when the membrane potential reaches the **threshold potential**, which is the $\mathbf{V_{th} = -55\ mV}$.
- The **membrane time constant** $\boldsymbol{\tau_m = \frac{C_m}{g_L}}$ controls how fast the membrane potential reacts to changes in input.
- After each spike, the neuron enters a **refractory period** (2 ms), where it cannot spike again.



```
dt = 0.1;                          % Time step (ms)
t = 0:dt:1000;                     % Time vector (ms)


Cm = 0.005;                    % Membrane capacitance (uF/cm^2)
gL = 0.002;                    % Leak conductance (mS/cm^2)
EL = -70;                      % Resting potential (mV)
Vth = -55;                     % Threshold potential (mV)
Vreset = -70;                  % Reset potential (mV)
tau_m = Cm / gL;               % Membrane time constant (ms)
V = EL * ones(size(t));        % Initialize membrane potential
refractory_period = 2;          % Refractory period duration (ms)
```

Input Current Parameters

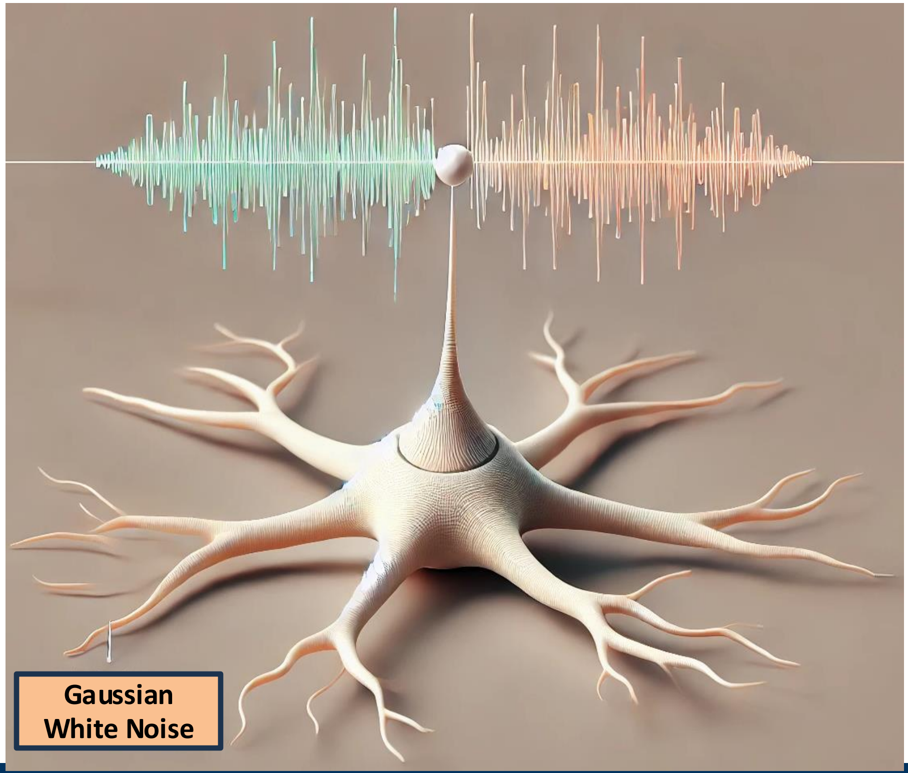
- The input is a sinusoidal wave of amplitude I_0 = 40 nA and frequency f_γ = 80 Hz, simulating gamma oscillations.
- Gaussian white noise with a standard deviation of noise amplitude 15 nA is added to the input to simulate synaptic variability.



```
I0 = 40;                    % Amplitude of oscillatory input (nA)
f_gamma = 80;               % Frequency of gamma oscillation (Hz)
noise_amplitude = 15;        % Noise amplitude (nA)
I_noise = noise_amplitude * randn(size(t));  % Gaussian white noise
I_input = I0 * sin(2 * pi * f_gamma * t / 1000) + I_noise;  % Gamma oscillation
```

Why is Gaussian White Noise Used?

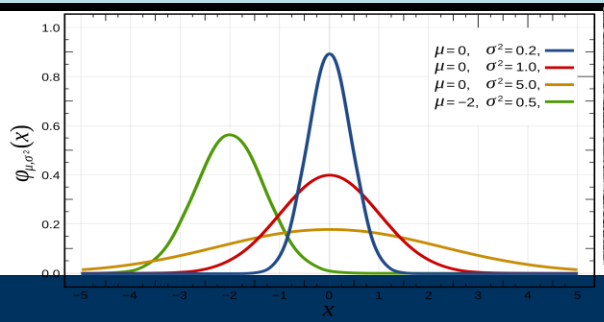
- In neuroscience, synaptic input to neurons is not always regular.
- It often contains randomness due to the probabilistic nature of neurotransmitter release.
- Gaussian white noise is a random signal with values drawn from a normal (Gaussian) distribution.
- These values are uncorrelated over time (white noise)
- Gaussian white noise is often used to simulate this random variability in input currents to neurons.
- In simulations, Gaussian white noise adds realistic variability to signals or inputs

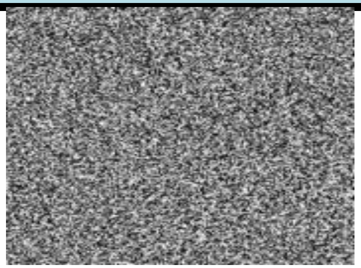


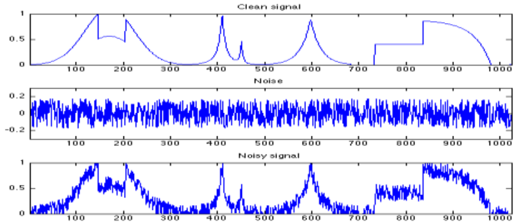
Key Characteristics of Gaussian White Noise

- Most of the values are **centred** around a **mean** (often set to zero).
- The probability of extreme values (very high or very low) **decreases symmetrically** as you move away from the mean.
- The distribution has a **bell-shaped** curve.
- The **spread of values** around the mean is determined by the standard deviation.
- SD is often referred to as the **noise amplitude**.
- Mathematically, the Gaussian distribution is defined as:
- The mean value is denoted as **μ**.
- The standard deviation (noise amplitude) is denoted as **σ**.
- The white noise contains all frequencies of a signal in equal power.
- This means that the noise is uncorrelated over time.
- Each noise value at a given time point is independent of the values at other time points.
- In the time domain, white noise is unpredictable and varies rapidly.

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

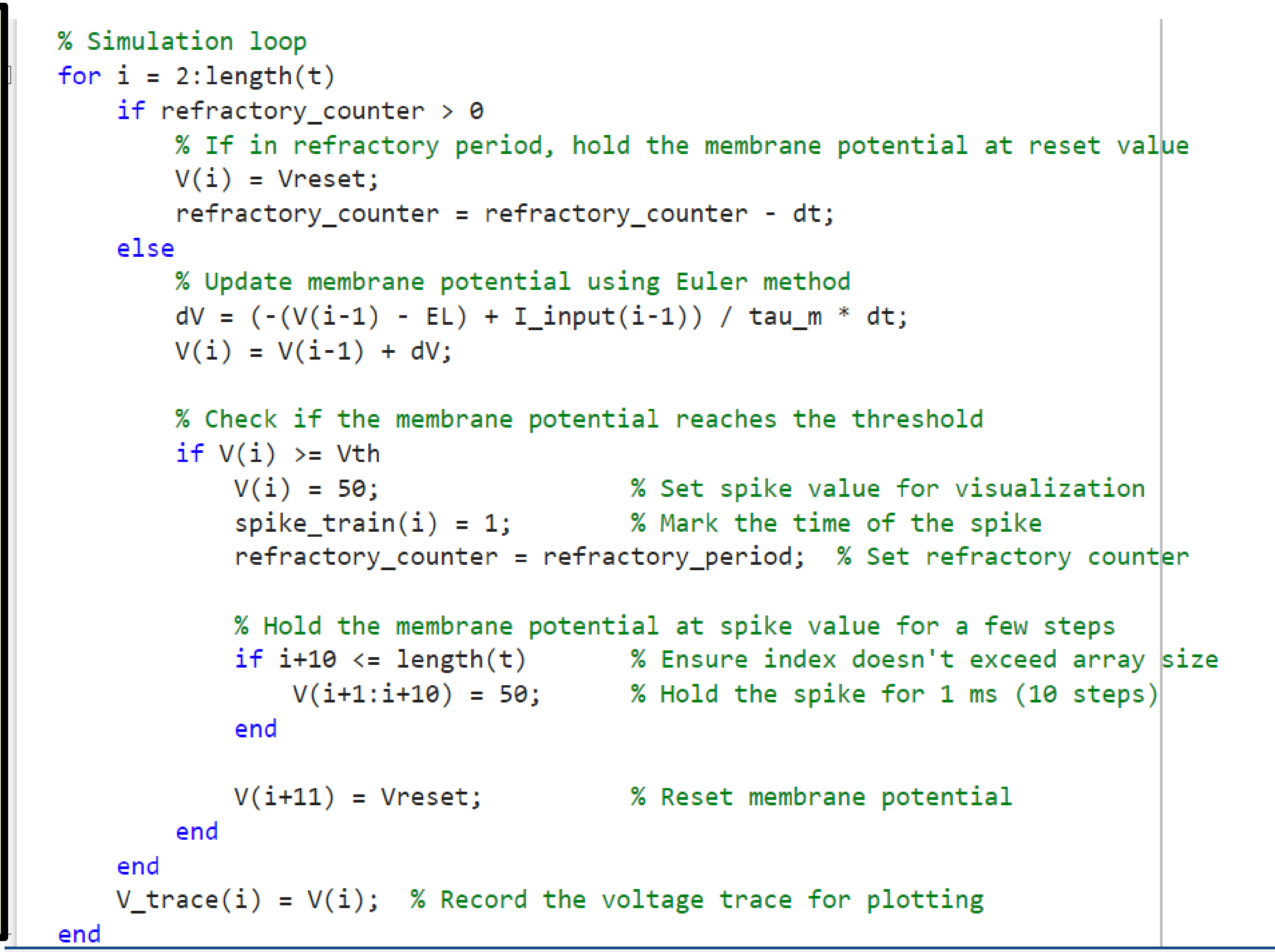






The simulation loop

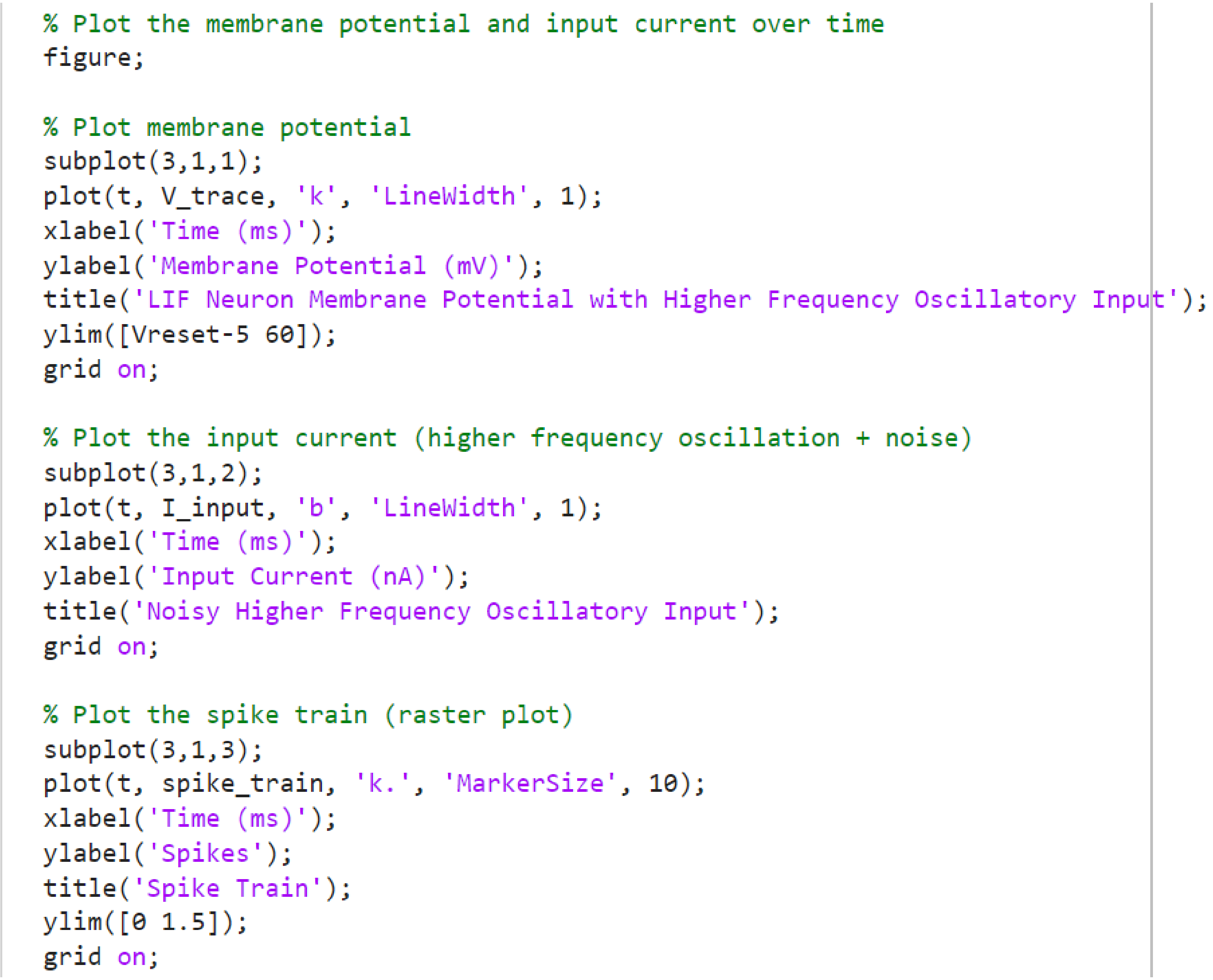
- The membrane potential **V(t)** is updated at each time step using the **Euler** method, where it integrates the oscillatory and noisy input.
- If the membrane potential reaches the threshold **V_{th} = -55 mV**, a spike is generated, and the membrane potential is reset to **V_{reset} = -70 mV** after a brief period.
- After spiking, the neuron cannot spike again for a short refractory period (2 ms), during which the membrane potential remains at the reset value.



```
% Simulation loop
for i = 2:length(t)
    if refractory_counter > 0
        % If in refractory period, hold the membrane potential at reset value
        V(i) = Vreset;
        refractory_counter = refractory_counter - dt;
    else
        % Update membrane potential using Euler method
        dV = (-(V(i-1) - EL) + I_input(i-1)) / tau_m * dt;
        V(i) = V(i-1) + dV;

        % Check if the membrane potential reaches the threshold
        if V(i) >= Vth
            V(i) = 50;                         % Set spike value for visualization
            spike_train(i) = 1;                % Mark the time of the spike
            refractory_counter = refractory_period;  % Set refractory counter

            % Hold the membrane potential at spike value for a few steps
            if i+10 <= length(t)               % Ensure index doesn't exceed array size
                V(i+1:i+10) = 50;              % Hold the spike for 1 ms (10 steps)
            end

            V(i+11) = Vreset;                  % Reset membrane potential
        end
    end
    V_trace(i) = V(i);  % Record the voltage trace for plotting
end
```

Visualization

- The **Membrane Potential Plot** shows how the neuron’s membrane potential evolves over time, indicating when the neuron spikes.
- The **Input Current Plot** displays the oscillatory and noisy input current driving the neuron.
- The **Spike Train Plot** shows the time points when the neuron spikes.



```
% Plot the membrane potential and input current over time
figure;

% Plot membrane potential
subplot(3,1,1);
plot(t, V_trace, 'k', 'LineWidth', 1);
xlabel('Time (ms)');
ylabel('Membrane Potential (mV)');
title('LIF Neuron Membrane Potential with Higher Frequency Oscillatory Input');
ylim([Vreset-5 60]);
grid on;

% Plot the input current (higher frequency oscillation + noise)
subplot(3,1,2);
plot(t, I_input, 'b', 'LineWidth', 1);
xlabel('Time (ms)');
ylabel('Input Current (nA)');
title('Noisy Higher Frequency Oscillatory Input');
grid on;

% Plot the spike train (raster plot)
subplot(3,1,3);
plot(t, spike_train, 'k.', 'MarkerSize', 10);
xlabel('Time (ms)');
ylabel('Spikes');
title('Spike Train');
ylim([0 1.5]);
grid on;
```