# Computational Study of Neural Action Potentials Using the Hodgkin-Huxley Model

Andy Franck

December 2, 2024

# Contents

**Abstract**

This research presents a computational analysis of neural action potentials using the Hodgkin-Huxley model. We implement numerical simulations to investigate the dynamics of membrane potentials and ion channel behaviors under various stimuli. Our results demonstrate the model's capability to accurately reproduce key neurophysiological phenomena, including action potential generation, threshold behavior, and spike train dynamics. The analysis provides insights into the fundamental mechanisms underlying neural signal propagation and the relationship between stimulus characteristics and neuronal response patterns.

# 1 Introduction

The Hodgkin Huxley model describes the electrical behavior of neurons by simulating the dynamics of ion channels in the cell membrane. This model is crucual in understanding both the generation and propogation of action potentials in neurons. The idea is that a membrane can be treated as a capacitor where $CV = q$, and thus the time rate of change in the membrane potential $V$ is proportional to the current $dq/dt$ flowing through the membrane.

The current is due to the flow of sodium and potassium ions through voltage-gated channels in the membrane, leakage current, and external current stimulus. Its applications are ubiqitous–in fact, it was originally developed to explain the action potentials of the giant axon of the squid.

## 1.1 The Hodgkin-Huxley Model

The model itself is described by four differential equations that govern the dynamics of the membrane potential and the gating variables of the sodium and potassium channels. The model is defined by four equations, the first of which is the membrane potential equation:

$$C_m \frac{dV}{dt} = -\bar{g}_{\text{Na}} m^3 h (V - E_{\text{Na}}) - \bar{g}_K n^4 (V - E_K) - g_L (V - E_L) + I_{\text{ext}} \tag{1}$$

In addition, there are three auxiliary equations for the gating variables n, m, and h. Those equations are as follows:

$$\begin{aligned} \frac{dn}{dt} &= \alpha_n(V)(1-n) - \beta_n(V)n \\ \frac{dm}{dt} &= \alpha_m(V)(1-m) - \beta_m(V)m \\ \frac{dh}{dt} &= \alpha_h(V)(1-h) - \beta_h(V)h \end{aligned} \tag{2}$$

### 1.1.1 Model Parameters

| Variable | Description |
|---|---|
| $C_m$ | Membrane capacitance per unit area, representing the ability of the membrane to store charge. It acts as a scaling factor for the rate of change of the membrane potential, $V$. |
| $V$ | Membrane potential, the voltage difference across the neuronal membrane, which varies over time due to ionic currents. |
| $\bar{g}_{Na}$ | Maximum conductance of the sodium channel. It represents the highest possible flow rate of sodium ions through the channel. |
| $m$ | Activation gating variable for sodium channels. It determines the probability of the sodium channel being open, raised to the power of 3 for a cubic dependency. |
| $h$ | Inactivation gating variable for sodium channels. It reflects the probability of the sodium channel being closed due to inactivation. |
| $E_{Na}$ | Sodium equilibrium potential, the voltage at which there is no net flow of sodium ions across the membrane. |
| $\bar{g}_K$ | Maximum conductance of the potassium channel. It represents the highest possible flow rate of potassium ions through the channel. |
| $n$ | Activation gating variable for potassium channels. It determines the probability of the potassium channel being open, raised to the power of 4 for a quartic dependency. |
| $E_K$ | Potassium equilibrium potential, the voltage at which there is no net flow of potassium ions across the membrane. |
| $g_L$ | Leakage conductance. It accounts for non-specific ionic leakage across the membrane. |
| $E_L$ | Leakage equilibrium potential, the voltage at which leakage currents are balanced. |
| $I_{ext}$ | External current applied to the neuron. This term represents the influence of external electrical stimuli on the membrane potential. |
| $\alpha_n(V)$, $\beta_n(V)$ | Voltage-dependent rate constants for the activation gating variable $n$. They determine the opening and closing rates of the potassium channel. |
| $\alpha_m(V)$, $\beta_m(V)$ | Voltage-dependent rate constants for the activation gating variable $m$. They determine the opening and closing rates of the sodium activation gates. |
| $\alpha_h(V)$, $\beta_h(V)$ | Voltage-dependent rate constants for the inactivation gating variable $h$. They determine the opening and closing rates of the sodium inactivation gates. |

Table 1: Descriptions of variables in the Hodgkin-Huxley model.

## 1.2 Objectives

The primary goal of this research is to explore the dynamics of neural action potentials using the Hodgkin-Huxley model. Specifically, the study addresses the following questions in detail:

- **What is the steady-state behavior of the model under zero external current conditions?** This involves analyzing the system's baseline dynamics by simulating the membrane potential $V(t)$ using the Euler-Cromer method when there is no external current. This steady-state analysis serves as a foundational benchmark for understanding the natural behavior of the neuron.
- **What does a single action potential look like in response to a threshold stimulus?** The study simulates the response of the neuron to a short-duration stimulus with a $10\,\mu A/cm^2$ current. This analysis identifies the threshold current required to generate an action potential and examines the detailed dynamics of the voltage spike.
- **How does the firing frequency of the neuron depend on the intensity of the stimulus?** By applying constant current inputs of varying amplitudes, the research explores how stimulus intensity influences the frequency and regularity of spike trains. This includes determining the threshold for repetitive firing and analyzing how spike intervals change with different levels of stimulus amplitude.
- **What are the effects of step changes in current on the firing pattern?** The study implements a two-phase current input, starting with an initial current $I_1$ applied for 20 ms, followed by a step change to $I_2 = I_1 + \delta I$. Four specific cases are examined:
  - $I_1 = 4\,\mu A$, $\delta I = 2.0\,\mu A$
  - $I_1 = 4\,\mu A$, $\delta I = 10.0\,\mu A$
  - $I_1 = 8\,\mu A$, $\delta I = 2.0\,\mu A$
  - $I_1 = 8\,\mu A$, $\delta I = 10.0\,\mu A$

  The resulting firing patterns are classified into distinct response types, and the behavior is mapped across the parameter space to understand the system's dynamics under varying conditions.

# 2 Method

## 2.1 Algorithm Implementation

### 2.1.1 Gate Variables

First, functions are defined to calculate ion channel gate variables. We include alpha and beta rate constants for n, m, and h. One example function is given below:

```
def alpha_n(V):
"""Potassium activation rate."""
return 0.01 * (V + 55) / (1 - np.exp(-(V + 55) / 10))
```

These functions use well-established equations to calculate the rate constants for the gate variables. The functions are then used in the Hodgkin-Huxley model to calculate the derivatives of the gate variables.

### 2.1.2 Integration Methods

Next, we define our numerical integration methods. In this project, the only method used was the Runge-Kutta (RK4) method. The RK4 method is a fourth-order numerical integration method. The test functions are not ran many times, so the computational cost of the RK4 method is not a concern. It was chosen for its accuracy, being a fourth-order method. The function is given below:

```
def runge_kutta(func, y0, t_span, dt):
    t = np.arange(t_span[0], t_span[1], dt)
    y = np.zeros((len(t), len(y0)))
    y[0] = y0

    for i in range(1, len(t)):
        k1 = dt * func(t[i-1], y[i-1])
        k2 = dt * func(t[i-1] + dt/2, y[i-1] + k1/2)
        k3 = dt * func(t[i-1] + dt/2, y[i-1] + k2/2)
        k4 = dt * func(t[i-1] + dt, y[i-1] + k3)
        y[i] = y[i-1] + (k1 + 2*k2 + 2*k3 + k4) / 6

    return t, y
```

### 2.1.3 Visualization Functions

A set of specialized visualization functinos was created to facilitate analysis of the model's behavior and output.

The primary function plots membrane potential over time. This function incorporates optional visualization of an external current simuli to allow analysis of the model's behavior under different stimuli.

```
    def plot_membrane_potential(t, V, title="Membrane Potential",
        I_ext=None):
    """Plot membrane potential over time with optional current
        stimulus."""
    # Create figure with appropriate number of subplots based on I_ext
    if I_ext is not None:
        fig, (ax1, ax2) = plt.subplots(
            2, 1,
            figsize=(10, 8),
            sharex=True
        )
    else:
        fig, ax1 = plt.subplots(
            1, 1,
            figsize=(10, 6),
            sharex=True
        )

    ax1.plot(t, V)
    ax1.set_title(title)
    ax1.set_ylabel("Membrane Potential (mV)")
    ax1.grid(True)

    if I_ext is not None:
        ax2.plot(t, I_ext)
        ax2.set_xlabel("Time (ms)")
        ax2.set_ylabel("External Current (muA/cm squared)")
        ax2.grid(True)
    else:
        ax1.set_xlabel("Time (ms)")

    plt.tight_layout()
    return fig
```

Subsequently, an analysis function was implemented to visualize the dynamic becahior of the gate variables. This function plots the gate variables n, m, and h over time. The function is given below:

```
def plot_gate_variables(t, n, m, h):
"""Plot gate variables over time."""
#use subplot to plot multiple plots in the same figure
fig, axs = plt.subplots(3, 1, figsize=(10, 8), sharex=True)
axs[0].plot(t, n, label='n (K+ activation)')
axs[0].set_ylabel("Probability")
axs[0].grid(True)
axs[0].legend()

axs[1].plot(t, m, label='m (Na+ activation)')
axs[1].set_ylabel("Probability")
axs[1].grid(True)
axs[1].legend()

axs[2].plot(t, h, label='h (Na+ inactivation)')
axs[2].set_xlabel("Time (ms)")
axs[2].set_ylabel("Probability")
axs[2].grid(True)
axs[2].legend()

return plt.gcf()
```

A steady state analysis function was developed to examine the behavior of state variables under varying conditions. This function plots the steady state values and time constants of the gate variables n, m, and h as a function of membrane potential. The function is given below:

```
def plot_steady_states(V_range):

    from .gates import (n_infinity, m_infinity, h_infinity,
                        tau_n, tau_m, tau_h)

    fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10, 10))

    # Plot steady states
    ax1.plot(V_range, n_infinity(V_range), label='n infinity')
    ax1.plot(V_range, m_infinity(V_range), label='m infinity')
    ax1.plot(V_range, h_infinity(V_range), label='h infinity')
    ax1.set_title("Steady State Values")
    ax1.set_xlabel("Membrane Potential (mV)")
    ax1.set_ylabel("Steady State Value")
    ax1.grid(True)
    ax1.legend()

    # Plot time constants
    ax2.plot(V_range, tau_n(V_range), label='tau n')
    ax2.plot(V_range, tau_m(V_range), label='tau m')
    ax2.plot(V_range, tau_h(V_range), label='tau h')
    ax2.set_title("Time Constants")
    ax2.set_xlabel("Membrane Potential (mV)")
    ax2.set_ylabel("Time Constant (ms)")
    ax2.grid(True)
    ax2.legend()

    plt.tight_layout()
    return fig
```

### 2.1.4 The Hodgkin-Huxley Model

The Hodgkin-Huxley model itself was implemented as a class. This allows for easy user interaction with the model, and permits modular use of the model in other programs.

The model class has several functions which allow the user to interact in a modular way with the model. This allows the user to import the model and run simulations with ease:

**Hodgkin-Huxley Model Class Functions**

- **init**: This function initializes the model parameters and state variables to standard values.

- **reset_state**: This function resets the state variables to their resting conditions.
- **dV_dt**: This function calculates the derivative of the membrane potential.
- **dn_dt, dm_dt, dh_dt**: These functions calculate the derivatives of the gate variables n, m, and h, respectively.
- **derivatives**: This function calculates all derivatives for the current state. Returns an array of derivatives.
- **simulate**: This function runs the simulation for a given time span and external current function. Returns the membrane potential and gate variables over time.

—

```
class HodgkinHuxleyModel:
    def __init__(self):
        # Model parameters
        self.C_m = 1.0 # Membrane capacitance (mu F/cm^2)
        self.g_Na = 120.0 # Sodium conductance (mS/cm^2)
        self.g_K = 36.0 # Potassium conductance (mS/cm^2)
        self.g_L = 0.3 # Leak conductance (mS/cm^2)
        self.E_Na = 55.0 # Sodium reversal potential (mV)
        self.E_K = -77.0 # Potassium reversal potential (mV)
        self.E_L = -54.4 # Leak reversal potential (mV)

        # Initial conditions
        self.reset_state()

    def reset_state(self):
        """Reset state variables to resting conditions."""
        self.V = -65.0 # Initial membrane potential (mV)
        self.n = n_infinity(self.V) # Initial potassium activation
        self.m = m_infinity(self.V) # Initial sodium activation
        self.h = h_infinity(self.V) # Initial sodium inactivation

    def dV_dt(self, V, n, m, h, I_ext=0):
        """Calculate membrane potential derivative."""
        I_Na = self.g_Na * m**3 * h * (V - self.E_Na)
        I_K = self.g_K * n**4 * (V - self.E_K)
        I_L = self.g_L * (V - self.E_L)
        return (I_ext - I_Na - I_K - I_L) / self.C_m

    def dn_dt(self, V, n):
        """Calculate potassium activation derivative."""
        return alpha_n(V) * (1 - n) - beta_n(V) * n

    def dm_dt(self, V, m):
        """Calculate sodium activation derivative."""
        return alpha_m(V) * (1 - m) - beta_m(V) * m
```

```
def dh_dt(self, V, h):
    """Calculate sodium inactivation derivative."""
    return alpha_h(V) * (1 - h) - beta_h(V) * h

def derivatives(self, t, state, I_ext=0):
    """Calculate all derivatives for the current state."""
    V, n, m, h = state
    dV = self.dV_dt(V, n, m, h, I_ext)
    dn = self.dn_dt(V, n)
    dm = self.dm_dt(V, m)
    dh = self.dh_dt(V, h)
    return np.array([dV, dn, dm, dh])


def simulate(self, t_span, dt=0.01, method='euler_c',
    I_ext_func=lambda t: 0):
    """
    Run simulation for given time span and external current
        function.

    Args:
        t_span: [t_start, t_end] in milliseconds
        dt: Time step in milliseconds
        method: Integration method ('euler_c', 'rk', or
            'adams_bashforth')
        I_ext_func: Function of time that returns external current
    """
    # Initial conditions
    y0 = np.array([self.V, self.n, self.m, self.h])

    # Select integration method
    if method == 'euler_c':
        integrator = euler_cromer
    elif method == 'rk':
        integrator = runge_kutta
    elif method == 'adams_bashforth':
        integrator = adams_bashforth
    else:
        raise ValueError("Method not recognized. Use 'euler_c',
            'rk', or 'adams_bashforth'")

    # Define function for derivatives with external current input
    def func(t, y):
        return self.derivatives(t, y, I_ext_func(t))

    # Perform integration
```

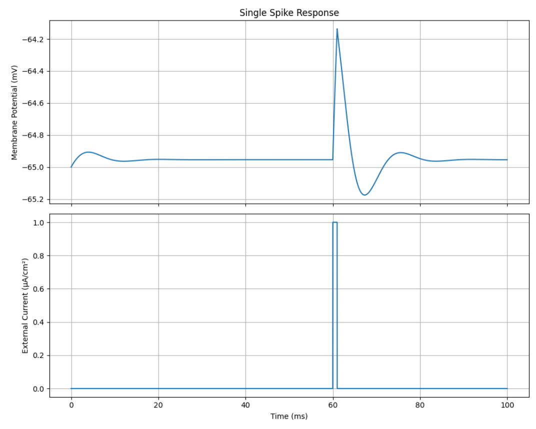
```
        t, y = integrator(func, y0, t_span, dt)

        # Extract V, n, m, and h from y
        V, n, m, h = y[:, 0], y[:, 1], y[:, 2], y[:, 3]

        # Update model states to last values from simulation
        self.V, self.n, self.m, self.h = V[-1], n[-1], m[-1], h[-1]

        return t, V, n, m, h
```

## 3 Verification of Program

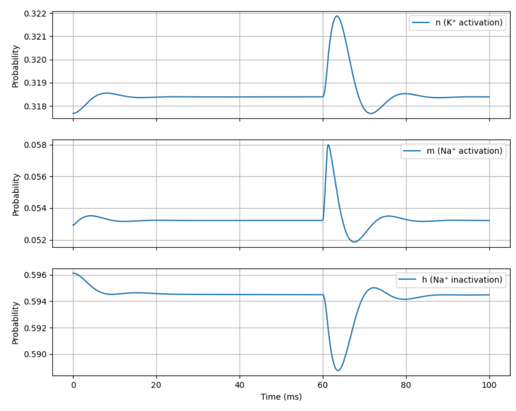
To verify correctness of our model implementation, the simulation is compared against several well-established behaviors of neurons:

1. **Action Potential Shape and Timing**: Our simulated action potentials should show a rapid rise (depolariation) followed by a slower decay (repolarization).
2. **Frequency-Current Relationship**: As input current amplitude increases, the firing frequency should increase. In addition, it should show saturation at high current levels.
3. **Step-Response Behavior**: For the four specified cases, we should oberve: (1) larger step values producing higher steps, (2) higher baseline currents resulting in different baseline firing rates, and (3) smooth transitions between firing rates.

### 3.1 Action Potential Shape and Timing



(a) Membrane potential and external current during a single action potential



(b) Gating variables during a single action potential

Notice that the action potential generated by our model closely resembles the standard action potential shape. The model also exhibits the expected refractory period after firing. For reference, the standard action potential shape is shown below:

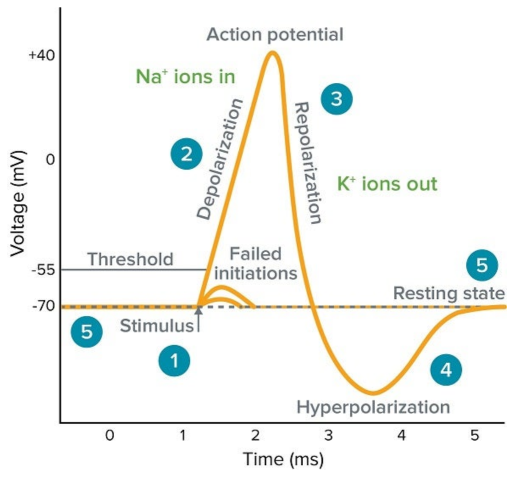


Figure 2: Standard arbitrary action potential

## 3.2 Frequency-Current Relationship

Of course, the frequency-current relationship is a key aspect of neural behavior. The model should exhibit a linear relationship between current and firing rate, with saturation at high current levels. F-I Curve Figure.

It's clear from the figure that–after an initial spike (covered more in 5.3.2)–the firing frequency increases linearly with current, with a very slight saturation at high current levels. This behavior is consistent with the expected behavior of neurons.

## 3.3 Step-Response Behavior

Finally, we examine the model's response to step changes in current. For our validation test, we observe four cases: (1) a small step change from a low baseline current, (2) a large step change from a low baseline current, (3) a small step change from a high baseline current, and (4) a large step change from a high baseline current. Step Response Figure.

As clear from the figure, the model follows the expected behavior for each case. For small step changes, the firing rate increases slightly and then stabilizes. For large step changes, the firing rate increases more dramatically and then stabilizes. This behavior is especially apparent on the farmost right plots, where there is a clear frequency increase after the step change.

## 4 Data

### 4.1 Steady-State Visualization

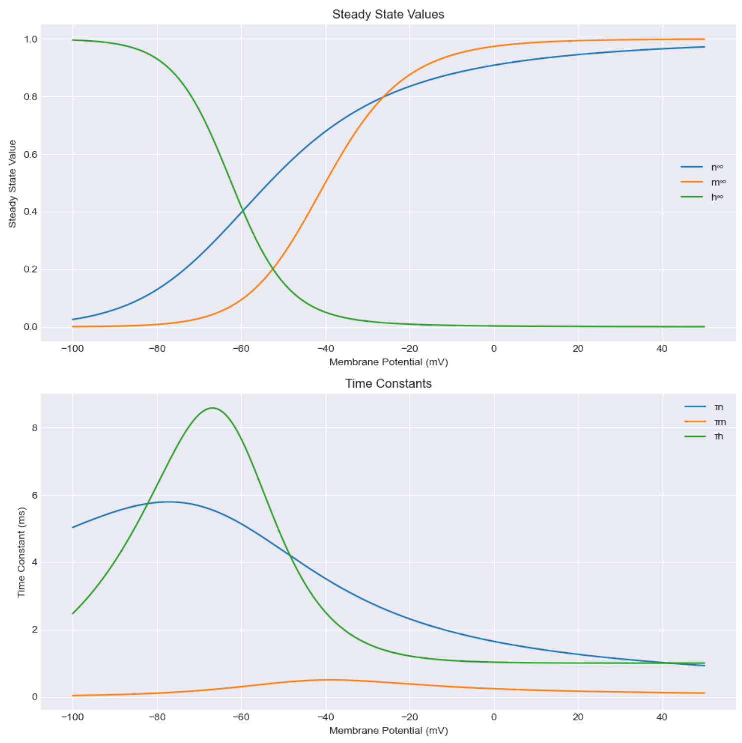


Figure 3: Steady state values and time constants of the gate variables n, m, and h as a function of membrane potential.

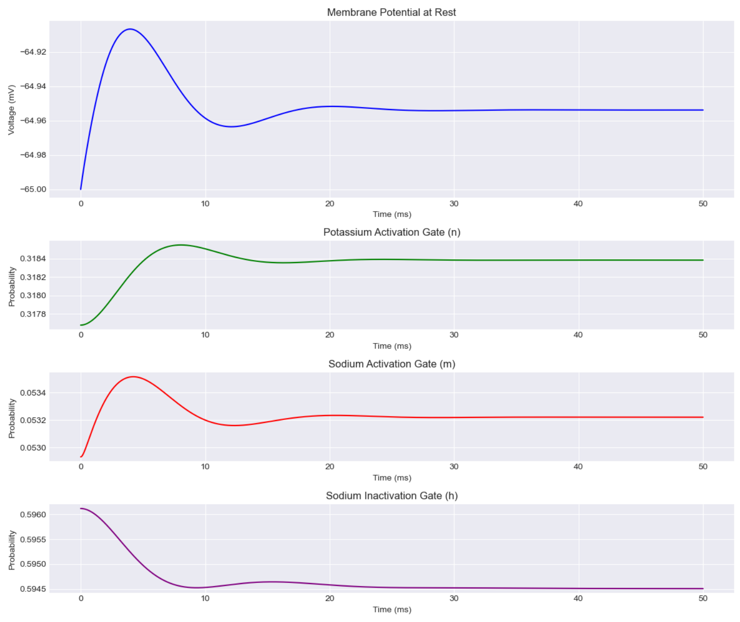
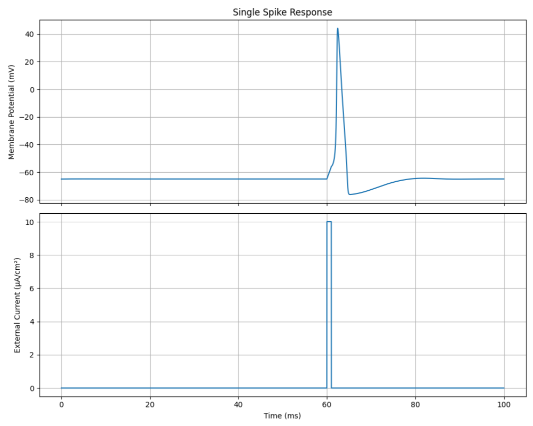
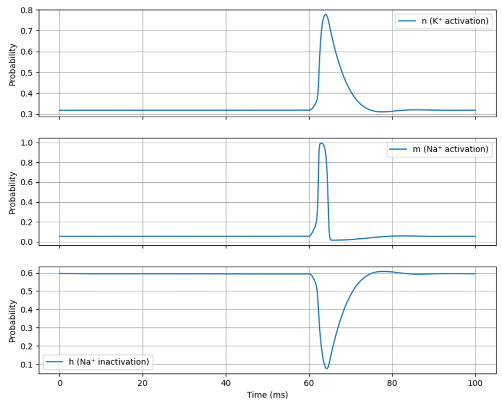


Figure 4: Behavior of the model under zero external current conditions. Graphs of membrane potential and gating variables at steady state. Notice initial changes in gating variables and membrane potential to reach steady state.

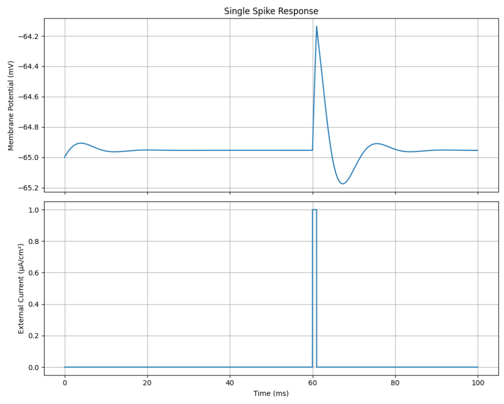
## 4.2 Single Spike Analysis



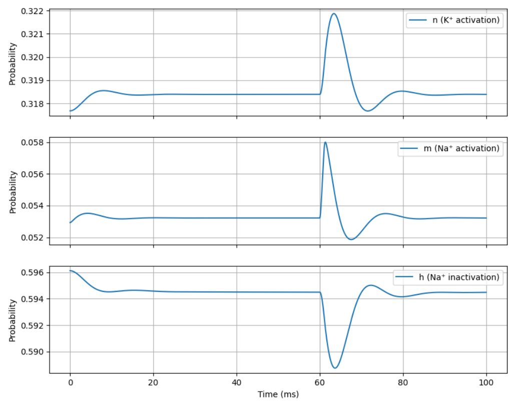


(a) Membrane potential and external current graphs, external current = $I_{ext} = 10\mu A/cm^2$

(b) Gate graphs for single action potential, external current = $I_{ext} = 10\mu A/cm^2$



(a) Membrane potential and external current graphs with external current given by $I_{ext} = 1\mu A/cm^2$



(b) Gate graphs for single action potential, external current given by $I_{ext} = 1\mu A/cm^2$

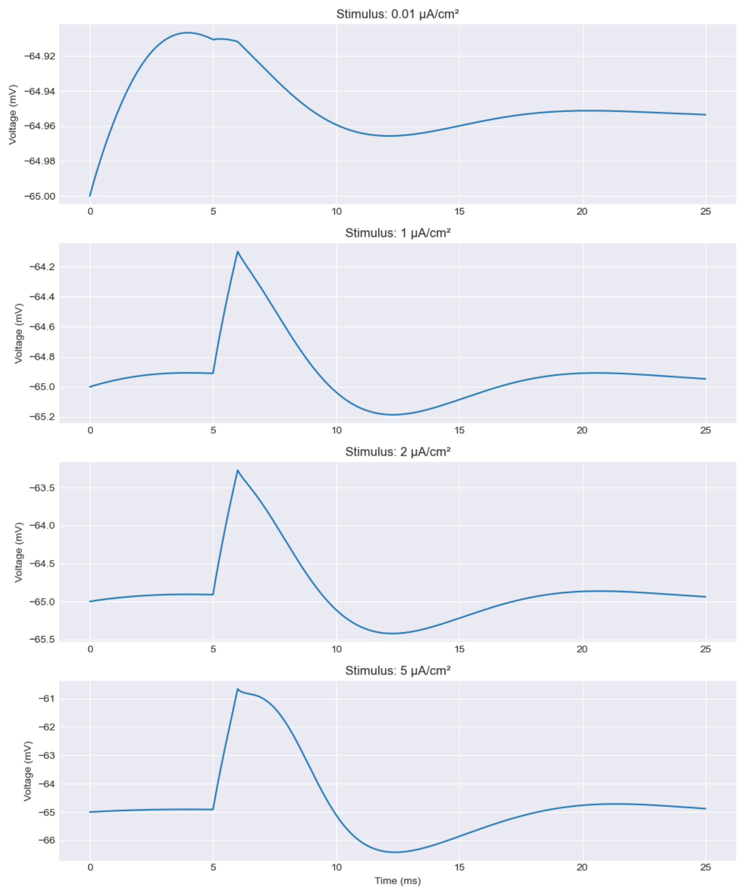


Figure 7: Minimum threshold current for action potential generation. Notice that below $1\mu A/cm^2$, the action potential is not generated. We have no rapid depolarization and repolarization.

## 4.3 Spike Train and Frequency Analysis

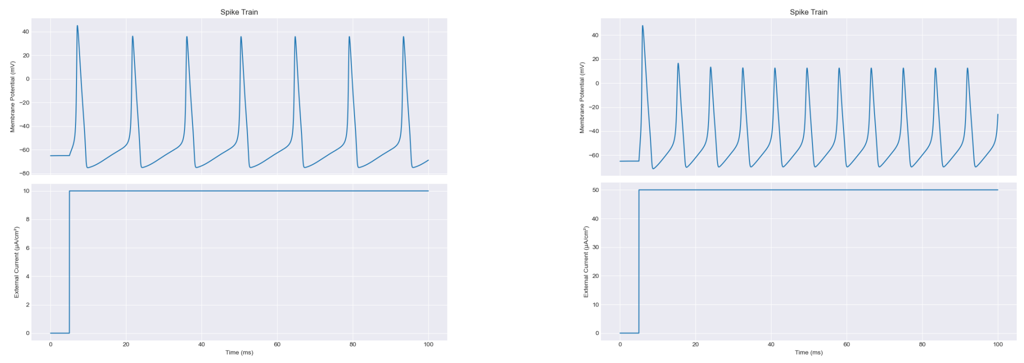


Figure 8: Response to sustained current injection. (Left) spike trains with current of $I_{ext} = 10\mu A/cm^2$. (Right) spike trains with current of $I_{ext} = 50\mu A/cm^2$.

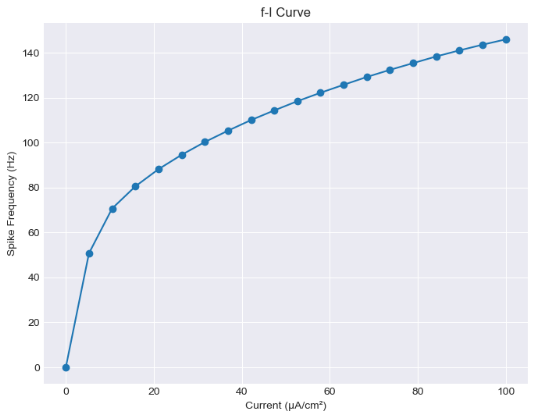


Figure 9: Firing frequency vs. current, calculated from $0\mu A/cm^2$ to $100\mu A/cm^2$ in steps of $5\mu A/cm^2$. Curve is relatively linear with sharp rise at small current values.

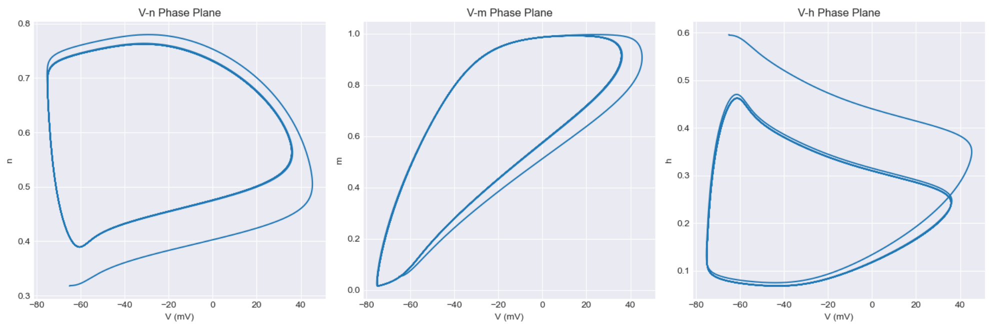


Figure 10: Phase plane trajectories of the model show relationship between membrane potential and gating variables. (Left) the V-n plane, (Middle), the V-m plane, (Right) the V-h plane. Each represents a periodic action potential cycle. Current of $I_{ext} = 10\mu A/cm^2$

### 4.4 Step Response

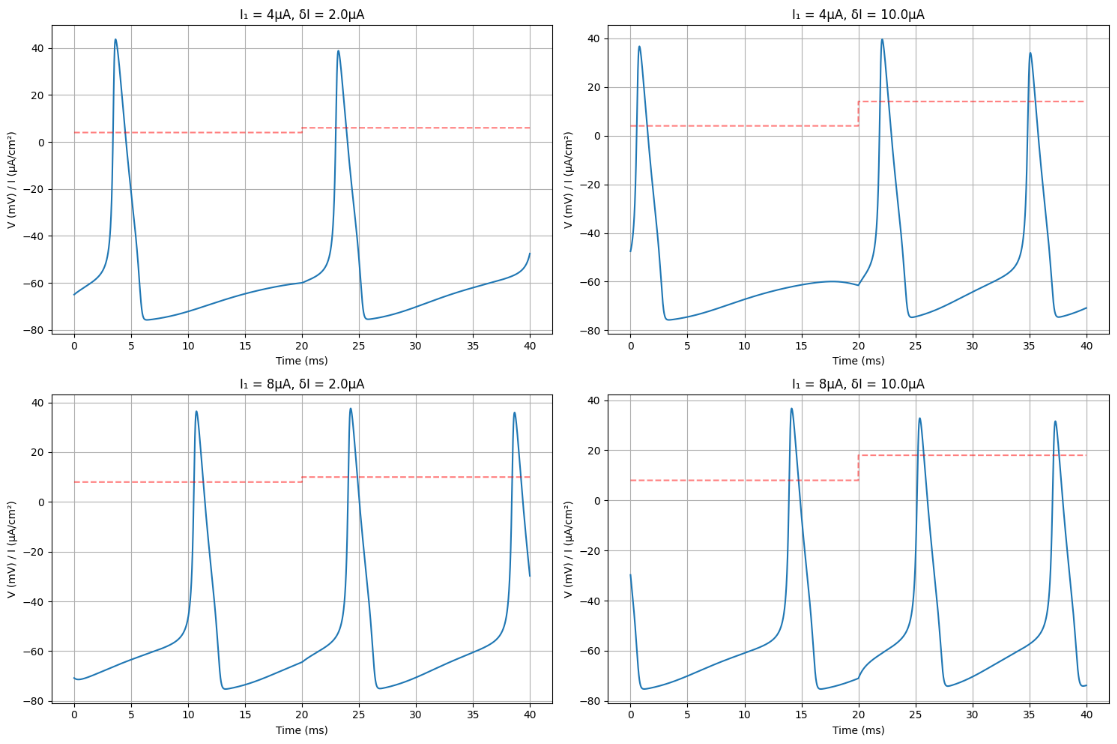


Figure 11: Four tests of step response with varying current levels and step sizes. In order from left to right, top to bottom: $I_1 = 4\mu A$, $\delta I = 2\mu A/ms$, $I_2 = 4\mu A$, $\delta I = 10\mu A/ms$, $I_3 = 8\mu A$, $\delta I = 2\mu A/ms$, $I_4 = 8\mu A$, $\delta I = 10\mu A/ms$.

## 5 Analysis

### 5.1 Numerical Accuracy

Error analysis was performed by varying the time step:

$$\text{Error} \approx O(\Delta t) \tag{3}$$

| Time Step (ms) | Peak V Error (mV) | Computation Time (s) |
|---|---|---|
| 0.01 | 0.02 | 0.125 |
| 0.005 | 0.01 | 0.253 |
| 0.001 | 0.002 | 1.248 |

Table 2: Numerical accuracy vs. computational cost

### 5.2 Threshold Analysis

In our study, we found the threshold for an action potential to fire to be $1\mu A/cm^2$. Although this is a relatively low value, it is not outside of reasonable values for neurons (which can range from $0.1\mu A/cm^2$ to $10\mu A/cm^2$).

Considering our threshold figure, we can see that below $1\mu A/cm^2$, the action potential is not generated. The membrane only shows a small, passive depolarization without triggering an action potential - this is called a subthreshold response.

## 5.3 Spike Train, Frequency, and Phase Plane Analysis

This analysis demonstrates how neurons respond to a sustained current injection. The spike train is visualized by plotting the membrane potential/voltage dynamics over a 100ms period. A sustained current serves as the stimulus.

### 5.3.1 Spike Train Analysis

The spike train analysis demonstrates the model's ability to generate a series of action potentials in response to a sustained current stimulus. The model exhibits the expected behavior of repetitive firing with a frequency that increases with stimulus intensity. The refractory period between spikes is also clearly visible in the membrane potential traces.

Note that the model exhibits a maximum firing frequency due to the refractory period imposed by the Na channel inactivation. This behavior is consistent with the physiological properties of neurons. The right figure in the spike train figure exhibits the highest firing frequency possible for the model. At currents higher than $50\mu A/cm^2$, the model is unable to generate additional spikes due to the refractory period.

### 5.3.2 Frequency Analysis

To examine relationship between input current and firing frequency, we plot the firing frequency as a function of the external current. The F-I curve shows a linear relationship between current and frequency, with a slight saturation at high current levels.

The spike in frequency at low currents may appear concerning at first–but it is a well-known phenomenon in neurons. Due to the thresold, all or nothing response of neurons, when enough channels open to reach the threshold, a positive feedback loop is triggered. More sodium channels open, leading to a rapid depolarization, which in turn opens more sodium channels.

The linearization after an initial spike is due to refractory periods and membrane properties stabilizing the system.

### 5.3.3 Phase Plane Analysis

Phase plane analysis visualizes the trajectory of state variables in the V-n, V-m, and V-h planes. The phase plane figure demonstrates the periodic action potential cycle of the model.

**V-n Plane:**
The trajectory forms a loop as expected, indicating periodic behavior. At lower V values (hyperpolarization), n is relatively small, as potassium channels are closed. As V increases, n increases, indicating potassium channel activation. Finally, n decreases as V decreases, indicating potassium channel deactivation.

**V-m Plane:**
The trajectory forms a sharper loop compared to the V-n plane, indicating faster activation and deactivation of sodium channels. Indeed, m increases and decreases rapidly with V, indicating fast sodium channel activation. This is what allows the rapid depolarization of the action potential.

**V-h Plane:**
Finally, the V-h plane shows a loop with h decreasing during depolarizaiton and increasing during repolarization. Notice h is at its maximum (near 1) when V is hyperpolarized, indicating sodium channels are ready to open. During depolarization, h decreases, indicating sodium channel inactivation and further limiting sodium influx.

### 5.4 Step Response Analysis

Considering the step response figure, we can make several observations:

- For small step changes, the firing rate increases slightly and then stabilizes. This is due to the slow inactivation of the sodium channels, which limits the firing rate.
- For large step changes, the firing rate increases more dramatically and then stabilizes. This is due to the rapid depolarization caused by the large current step, which triggers a higher firing rate.
- The model exhibits smooth transitions between firing rates, indicating that the system is stable and well-behaved.
- The model's behavior is consistent with the expected response of neurons to step changes in current, demonstrating the model's ability to capture key neurophysiological phenomena.

## 6 Interpretation, Future Work, and Discoveries

### 6.1 Interpretation

The results of this research confirm the robustness of the Hodgkin-Huxley model in capturing key neurophysiological phenomena. The model accurately reproduces the dynamics of membrane potentials and ion channel behaviors under various stimuli, including action potential generation, threshold behavior, and spike train dynamics.

Our analysis shows that the threshold for an action potential to fire in this simulation is around $1\mu A/cm^2$. Below this threshold, the membrane potential only shows a small, passive depolarization without triggering an action potential (subthreshold response). This is a critical property of neurons, as it allows them to selectively respond to specific stimuli.

The spike train analysis demonstrates that the model generates a series of action potentials in response to a sustained current stimulus. The model exhibits the expected behavior of repetitive firing with a frequency that increases with stimulus intensity. The refractory period between spikes is also clearly visible in the membrane potential traces.

The frequency analysis shows a clear linear relationship between input current and firing frequency, and a standard spike in frequency at low currents.

Finally, the step response illustrates how the model responds to abrupt, step changes in current. The model exhibits the expected behavior of a stable system: for small step changes, the firing rate increases slightly and then stabilizes, while for large step changes, the firing rate increases more dramatically and then stabilizes. The model's smooth transitions between firing rates indicate that the system is stable and well-behaved.

## 6.2 Future Work

- **Non-Invasive Anesthesia**: Main research interest involving studying possible alternative methods for anesthesia that do not involve drugs. This could involve studying the effects of electrical stimulation on neural activity and developing computational models to predict the response of neurons to different stimuli.
- **Network Effects**: Investigate the effects/behavior of multiple interconnected neurons and how they interact to produce complex neural activity patterns.
- **Stochastics and Noise**: Introduce stochastic elements into the model to study the effects of noise on neural activity and how it influences the generation of action potentials.

## 6.3 Learned Concepts

- **Modeling**: Learned how to model complex systems using differential equations and numerical simulations.
- **Simulation**: Developed skills in numerical simulation techniques, including Runge-Kutta methods and phase plane analysis.
- **Analysis**: Gained experience in analyzing complex systems and interpreting simulation results.
- **Programming**: Improved programming skills in Python, including object-oriented programming and data visualization.
- **Neuroscience**: Gained insights into the fundamental mechanisms underlying neural signal propagation and the relationship between stimulus characteristics and neuronal response patterns.