Introduction: Hodgkin–Huxley Model

The Hodgkin–Huxley model is a mathematical description of the electrical activity of the neuron. It was developed by Alan Hodgkin and Andrew Huxley in 1952, and it has been widely used to understand and predict the behavior of neurons. The model captures the essential features of action potential generation and propagation, including the role of voltage-gated ion channels.

by Anton Pasternak

Background: The Neuron and Action Potential

Neurons are specialized cells that transmit information throughout the nervous system. The action potential is a brief electrical signal that travels along the axon of a neuron.

It is generated by the rapid opening and closing of voltage-gated ion channels, allowing ions to flow across the cell membrane. This results in a transient change in the electrical potential of the neuron, known as depolarization.

Koch’s
"Biophysics of Computation"

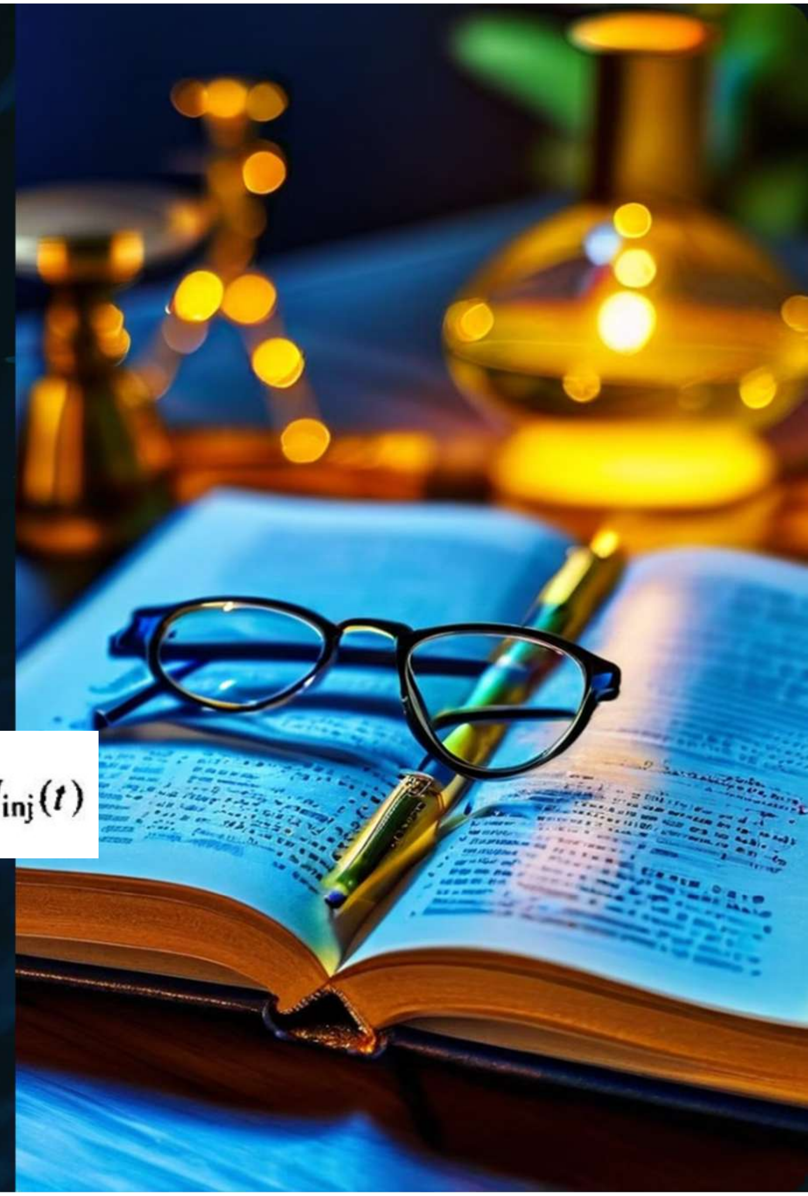
The book, "Biophysics of Computation: Information Processing in Single Neurons," by Christof Koch, is a valuable resource for understanding the Hodgkin–Huxley model.

Koch meticulously explains the theoretical foundation of the model and provides a comprehensive overview of its implementation and analysis.
This is the main equation that I’ve focused on:

$$C_m \frac{dV}{dt} = \overline{G}_{\mathrm{Na}} m^3 h (E_{\mathrm{Na}} - V) + \overline{G}_{\mathrm{K}} n^4 (E_{\mathrm{K}} - V) + G_m (V_{\mathrm{rest}} - V) + I_{\mathrm{inj}}(t)$$

In order to achieve an appropriate recreation of the model, I had to look for a lot more data and equations, left by Koch.

Next are all the equations and parameters I’ve gathered from Koch.



Key Equations and Parameters

The Hodgkin–Huxley model is a system of four coupled differential equations that describe the dynamics of the membrane potential and the gating variables.

The equations are defined by parameters that represent the maximum conductances of different ion channels and the time constants for activation and inactivation (I've called them just "gates" in my code).

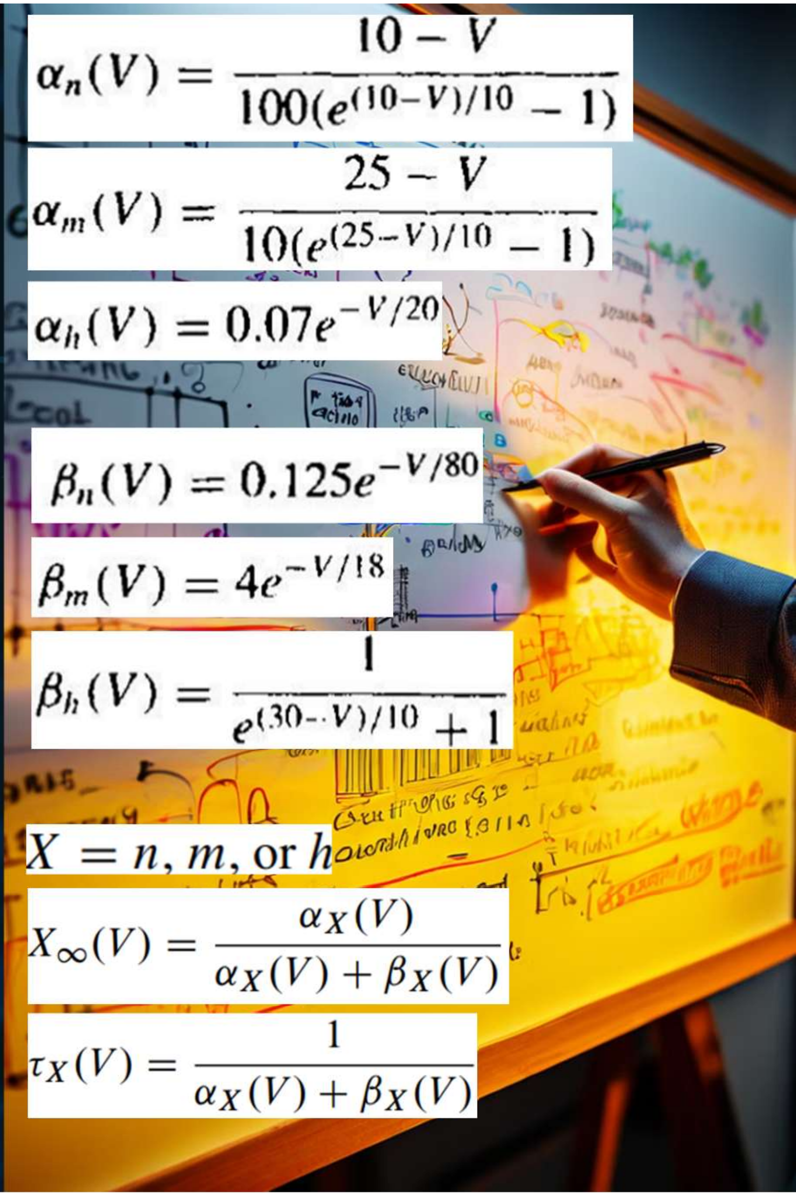
Potassium Gates:

$$\frac{dn}{dt} = \alpha_n(V)(1-n) - \beta_n(V)n = (n_\infty(V) - n)/\tau_n(V)$$

Sodium Gates:

$$\frac{dm}{dt} = \alpha_m(V)(1-m) - \beta_m(V)m = (m_\infty(V) - m)/\tau_m(V)$$

$$\frac{dh}{dt} = \alpha_h(V)(1-h) - \beta_h(V)h = (h_\infty(V) - h)/\tau_h(V)$$



$$\alpha_n(V) = \frac{10 - V}{100(e^{(10-V)/10} - 1)}$$

$$\alpha_m(V) = \frac{25 - V}{10(e^{(25-V)/10} - 1)}$$

$$\alpha_h(V) = 0.07e^{-V/20}$$

$$\beta_n(V) = 0.125e^{-V/80}$$

$$\beta_m(V) = 4e^{-V/18}$$

$$\beta_h(V) = \frac{1}{e^{(30-V)/10} + 1}$$

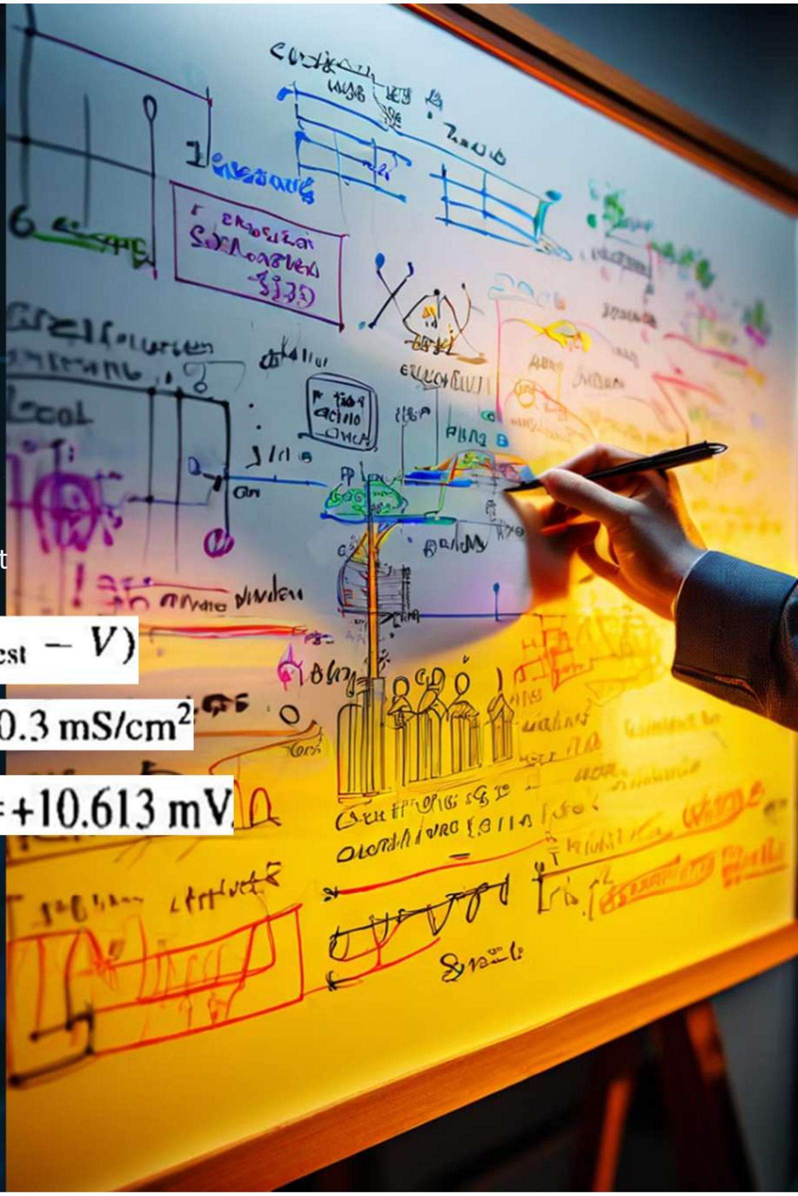
$X = n, m, \text{ or } h$

$$X_\infty(V) = \frac{\alpha_X(V)}{\alpha_X(V) + \beta_X(V)}$$

$$\tau_X(V) = \frac{1}{\alpha_X(V) + \beta_X(V)}$$

Key Equations and Parameters

I’ve also taken the currents equations from Koch’s book and the parameters he provided as well.



Potassium Current

$$I_K = \overline{G}_K n^4 (V - E_K)$$

$$\overline{G}_K = 36 \text{ mS/cm}^2$$

$$E_K = -12 \text{ mV}$$

Sodium Current

$$I_{Na} = \overline{G}_{Na} m^3 h (V - E_{Na})$$

$$\overline{G}_{Na} = 120 \text{ mS/cm}^2$$

$$E_{Na} = 115 \text{ mV}$$

Leak Current

$$G_m (V_{rest} - V)$$

$$G_m = 0.3 \text{ mS/cm}^2$$

$$V_{rest} = +10.613 \text{ mV}$$

$$C_m = 1 \ \mu\text{F/cm}^2$$

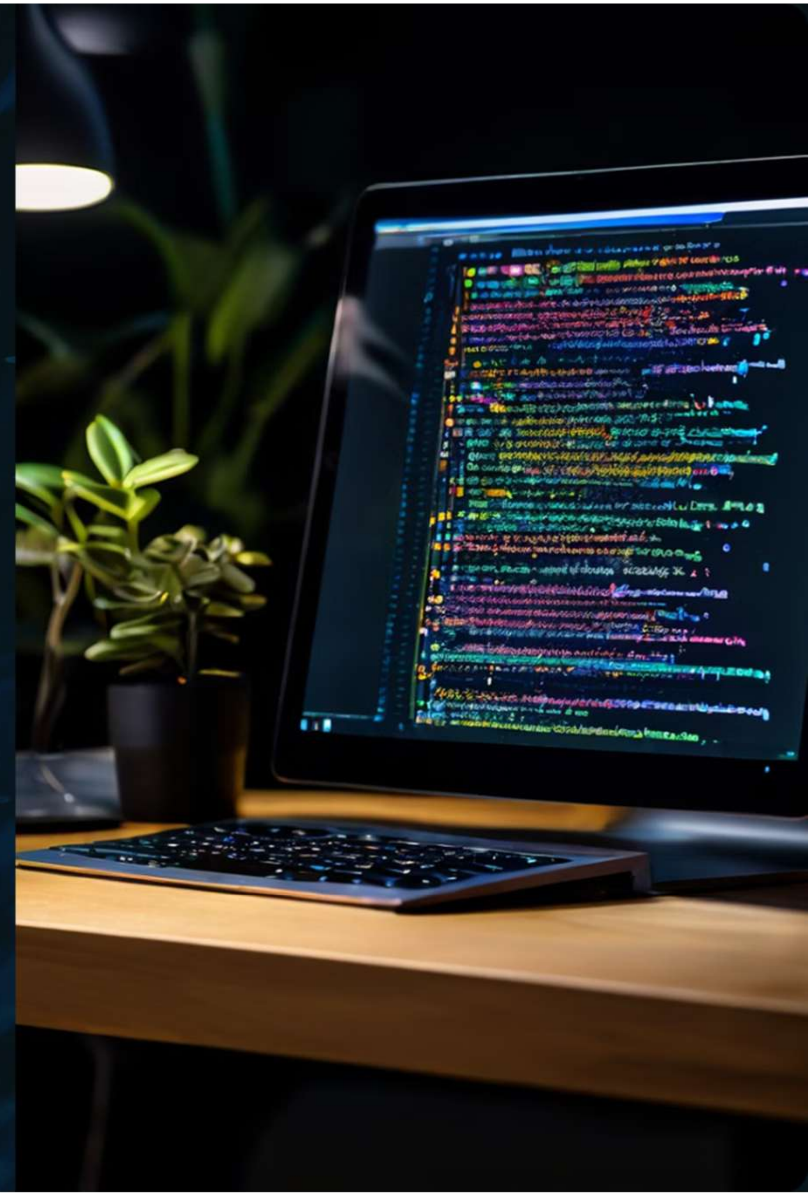
Membrane capacitance

Model Implementation in Python

The Hodgkin–Huxley model was implemented in Python using the NumPy and SciPy libraries and was plotted using Matplotlib library.
This allowed for efficient numerical integration of the differential equations governing the model. The code was written to be modular and extensible, allowing for easy modification and experimentation with different parameters and initial conditions.

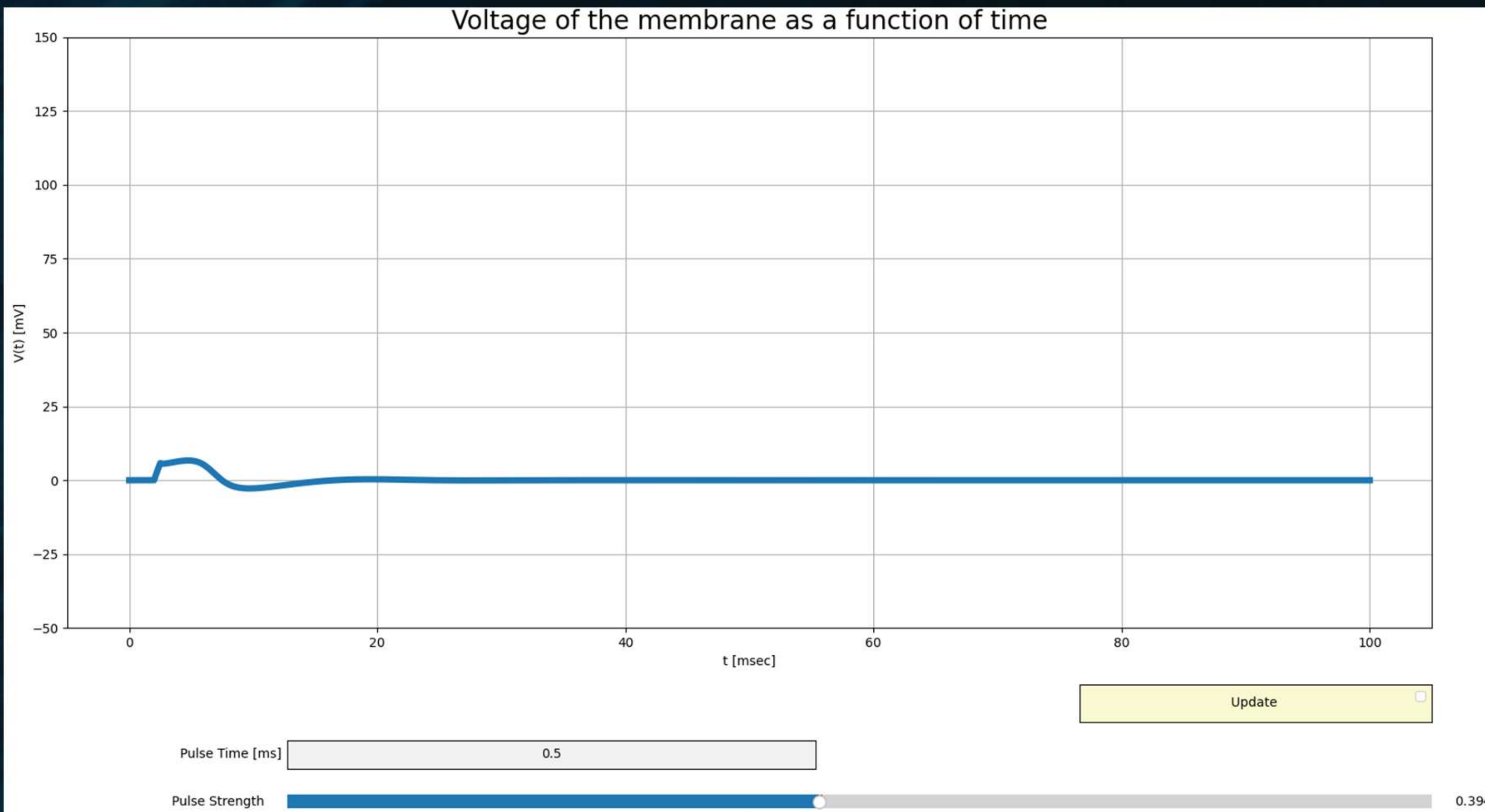
I wanted to have the same "baseline" (starting point) for my exploration of the model, as Koch's data, so I took the liberty to "adjust" a little the way I inject current to the system.

I found that in my model to generate an action potential when injecting current for 0.5msec, a current amplitude of 13.28nA is needed but in Koch's book it was supposed to be 0.4nA. I've checked everything and found no error, (probably the difference comes from temperature variables that were no included here) so I decided to add an amplifier middle-man between the injection needle itself and its "control panel". The amplifier takes the wanted current be to injected by the needle, amplifies it by a constant that "fixes" the sufficient current (that needs to be input into the "control panel") for action potential to be 0.4nA instead of 13.28nA (every injection is amplified by 33.2).



Simulation Results and Analysis: The action potential

The action potential is the interesting behavior of the neuron that accurse when the right conditions are met. Here we can see the effect of an externally injected current, on the membrane voltage of the neuron.

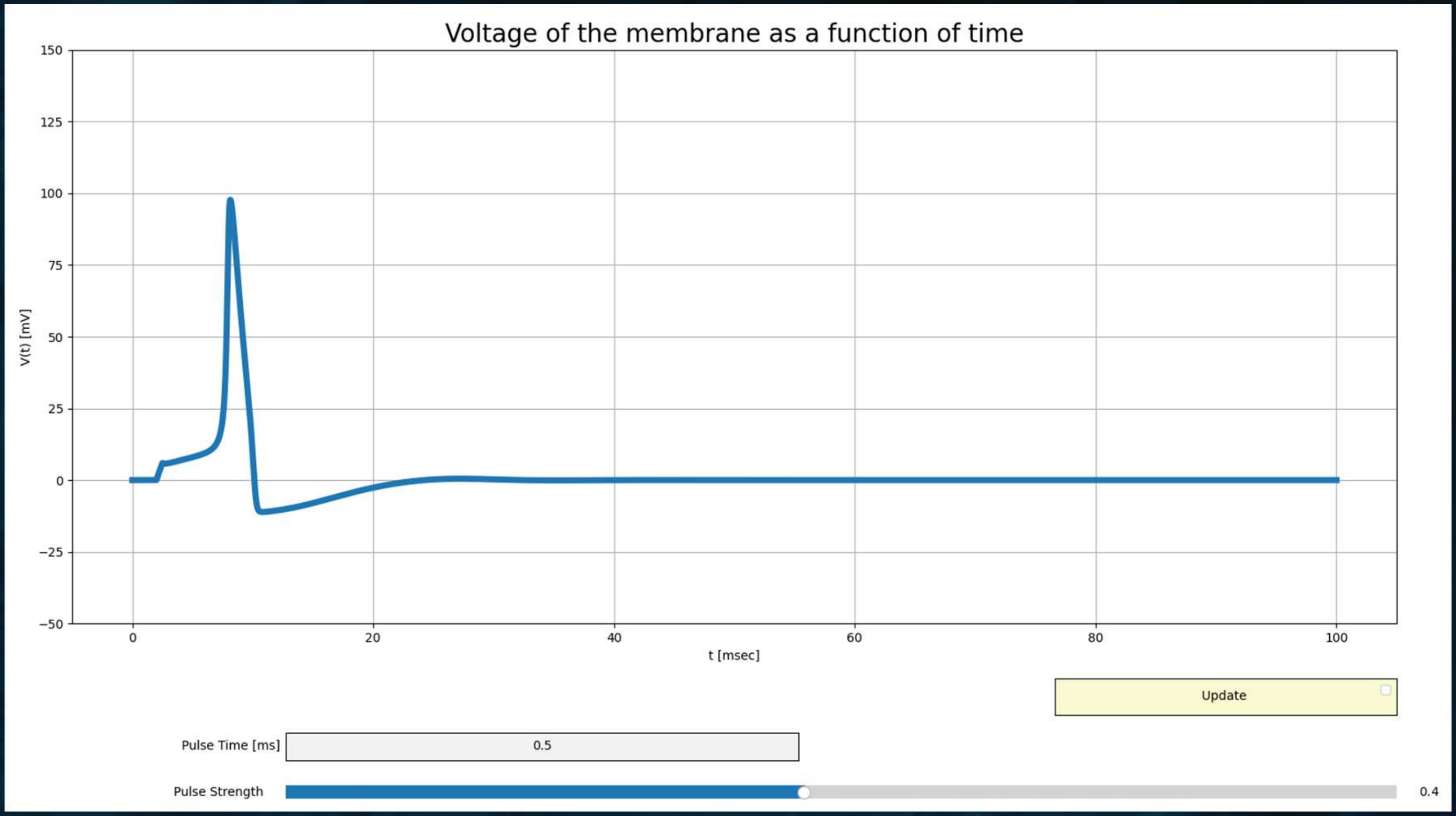


This is a plot of the voltage of the membrane as a function of time (and data about the injection that can be changed at the widget below the plot).

In this graph we can see that the input current was around 0.39nA and was 0.5msec long.

The neuron goes through depolarization and repolarization, but no action potential was generated here, yet.

Simulation Results and Analysis: The action potential



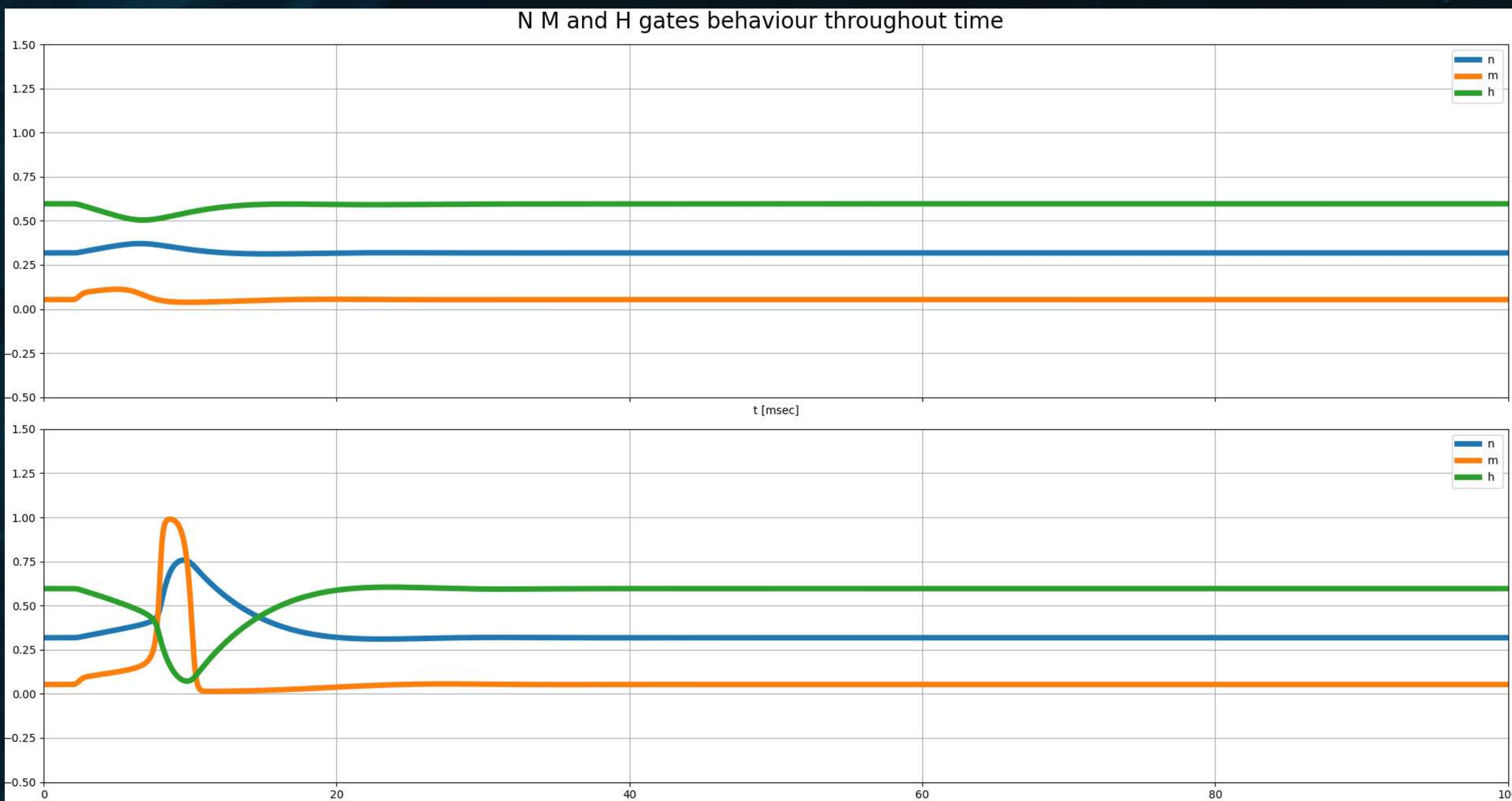
In contrast, in this graph we can see that the injection current was sufficient to generate an action potential in the neuron.

The current here is a little higher: 0.4nA.

Simulation Results and Analysis:

The N, M and H Gates

The simulation results reveal the dynamics of the gating variables N, M, and H, which represent the activation of sodium channels, activation of potassium channels, and inactivation of sodium channels, respectively.

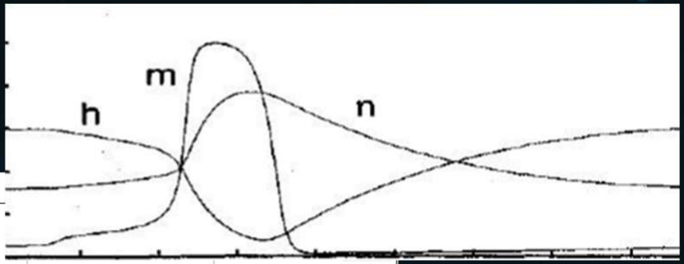


Here we can see the way that the gates behave through the whole time (100msec).

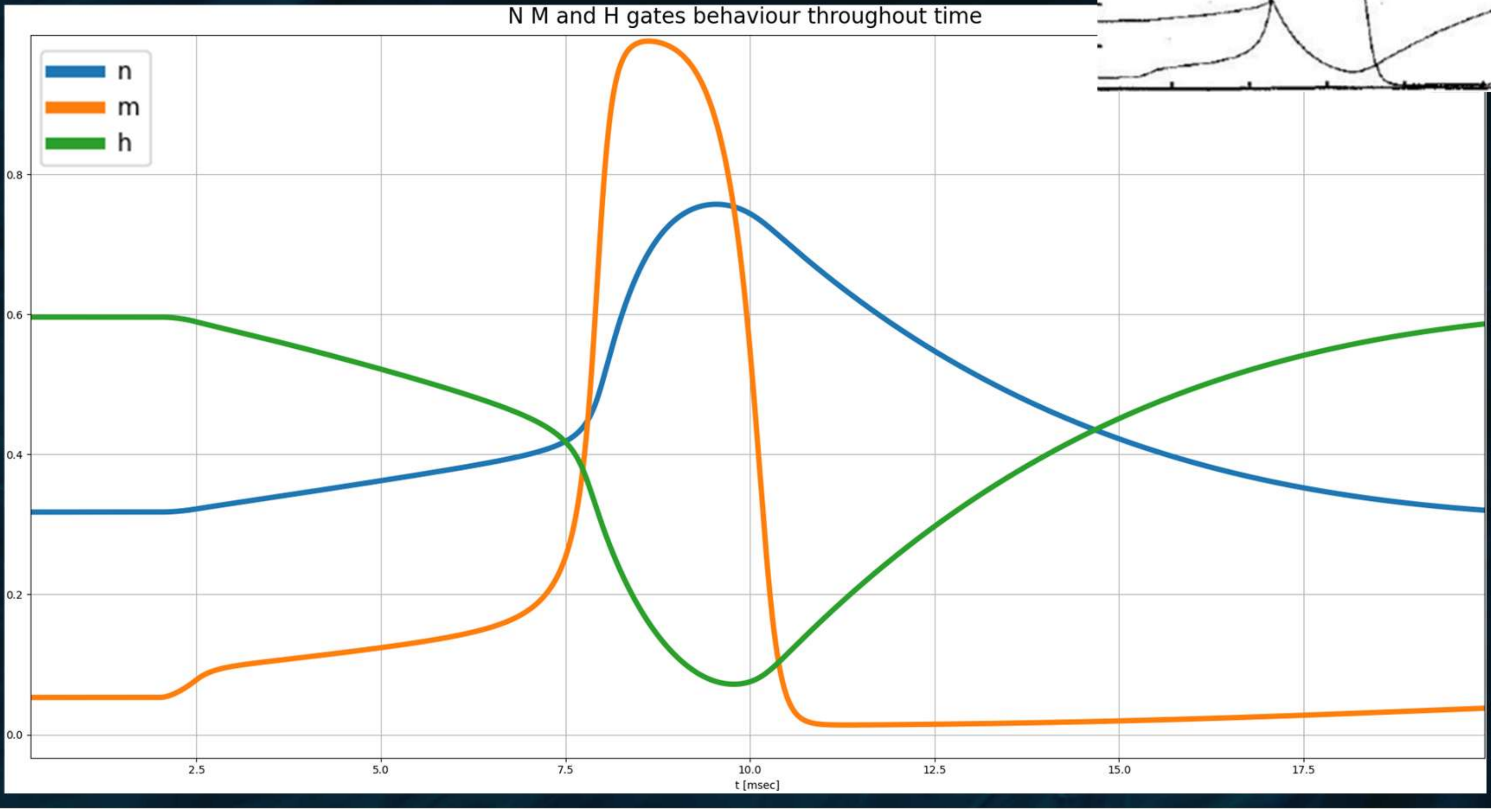
The first plot is the behavior of the gates at the insufficient current injection.

This second plot is of the slightly larger injection that generated the action potential.

Simulation Results and Analysis: The N, M and H Gates



(From Koch)



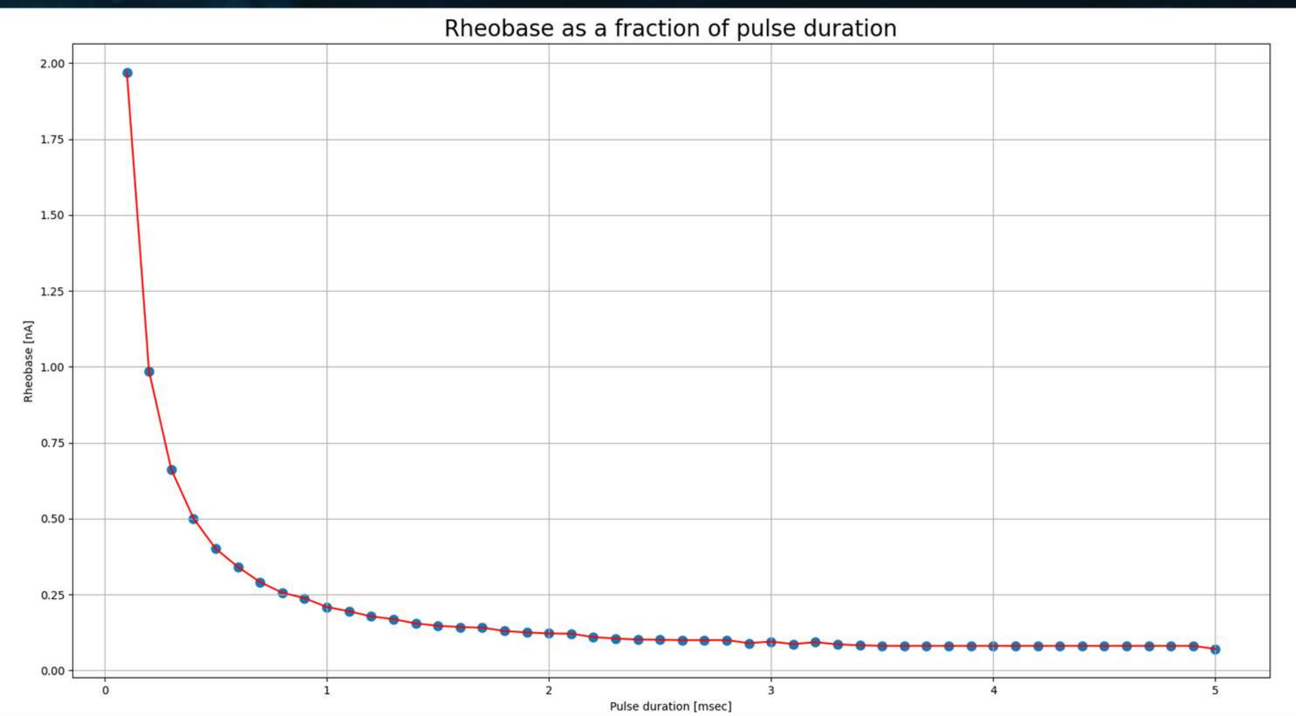
A little zoom in on the interesting areas for your pleasure

Simulation Results and Analysis: Rheobase and Pulse Duration

The rheobase is the minimum current required to elicit an action potential in a neuron. It is a key parameter in understanding the excitability of the neuron.

The simulation results showed that the rheobase decreases as the duration of the stimulus increases.

Simulation Results and Analysis: Rheobase as a Function of Pulse Duration

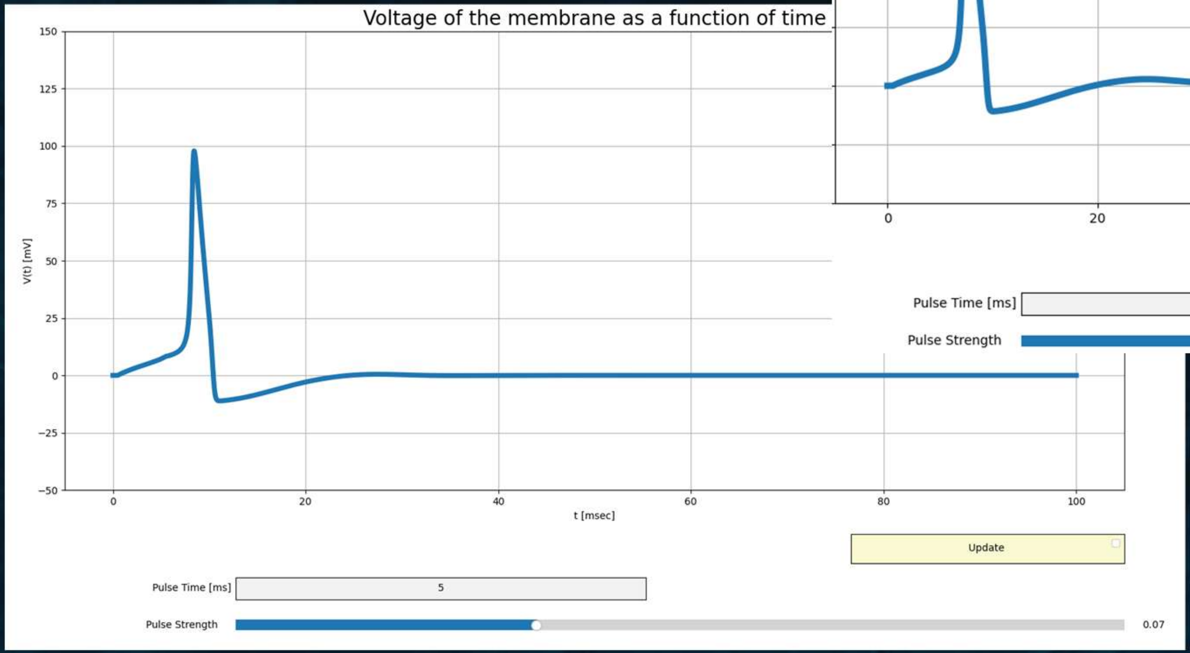


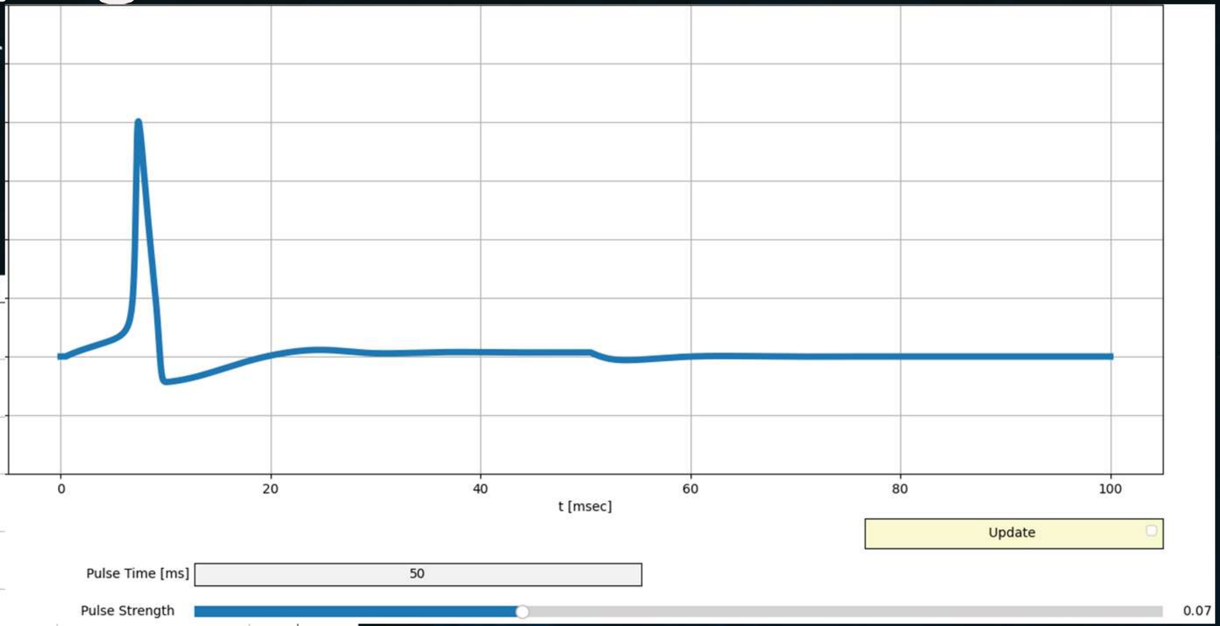
Here we can see a graph of the rheobase as a function of pulse duration that was made by a LOT of measurements of the minimum current that triggered an action potential with each duration (on a scale of 0.1msec from 0.1 to 5 [msec]).

Simulation Results and Analysis: The action potential, again

In here we'll continue to explore the action potential behavior when the duration of the pulse is changed.

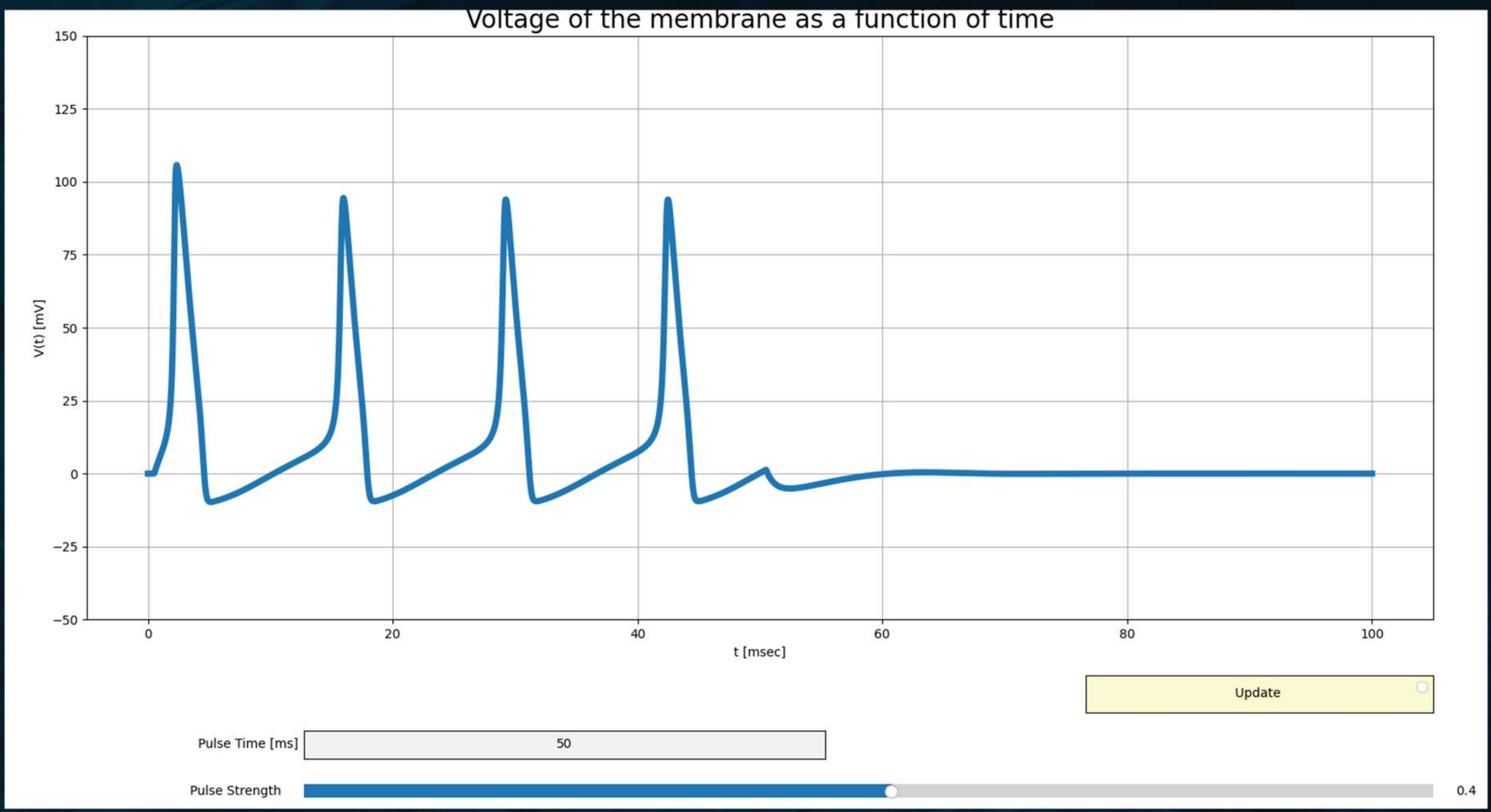
Here (down) is the last action potential that was generated in the previous graph, a current of around 0.07nA for 5msec.





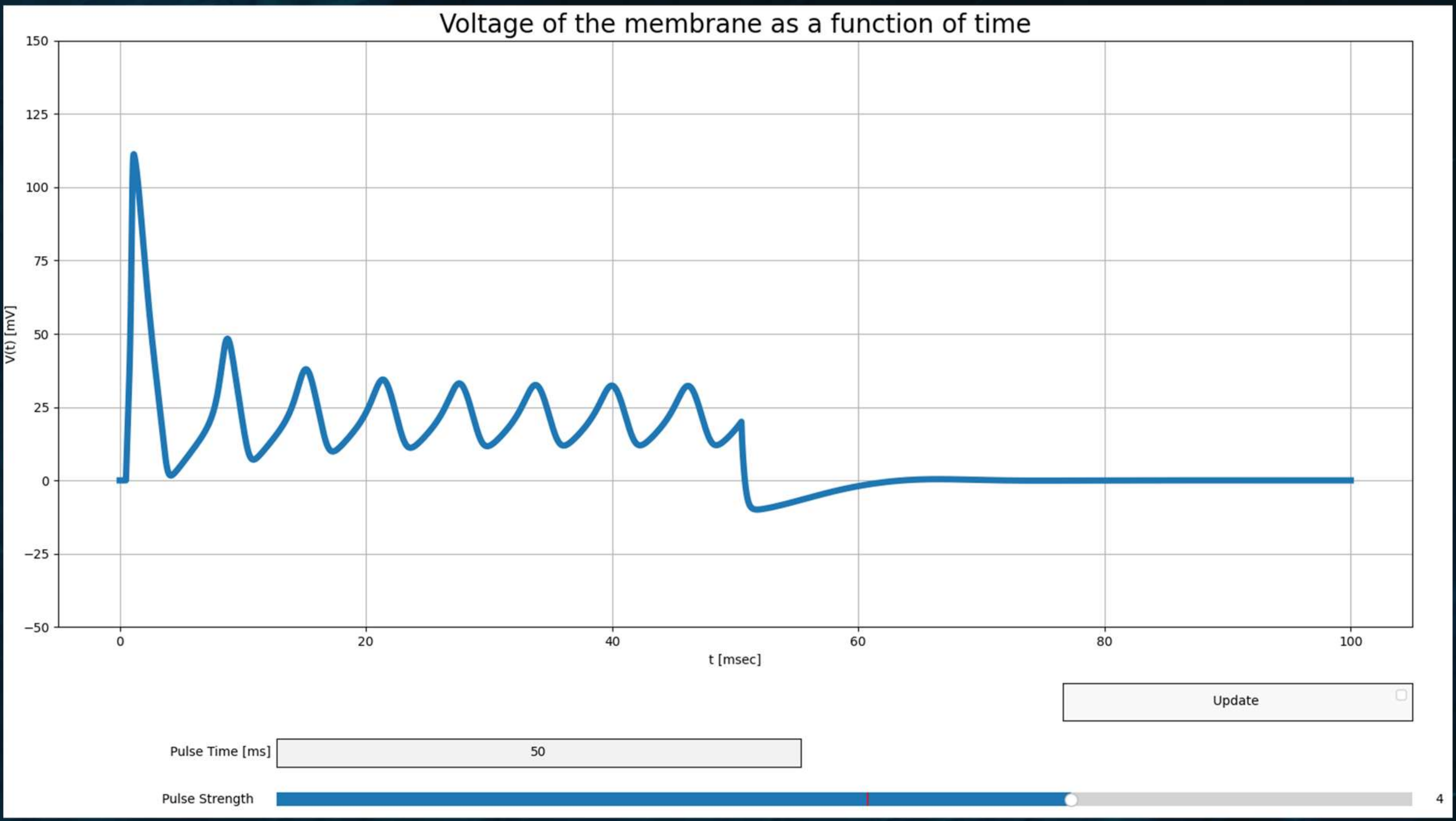
If we just increase the duration, it looks almost the same that's why we'll continue next with "cranking it up".

Simulation Results and Analysis: The action potential, Burst-fire



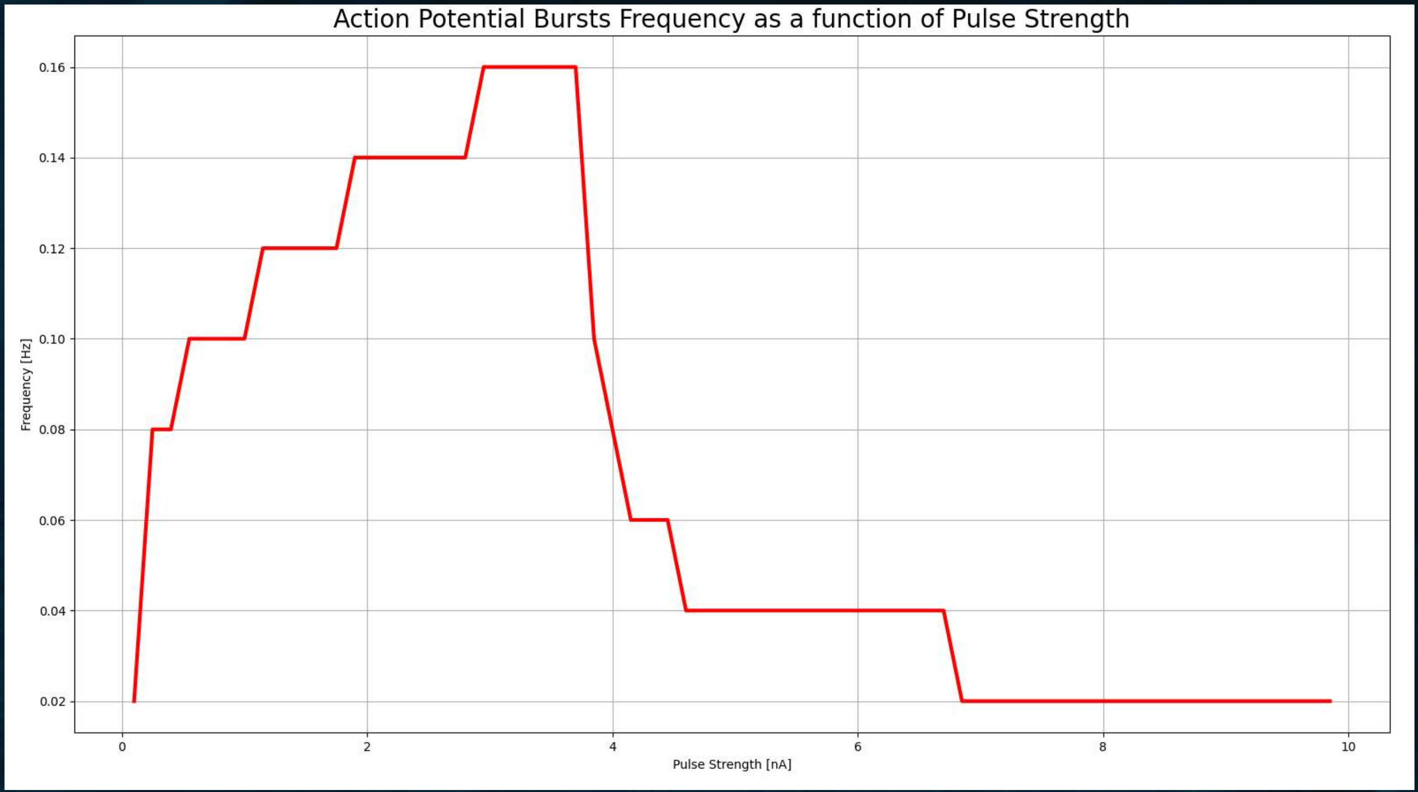
Here I updated the pulse to be as the original 0.4nA that we used earlier and something interesting is happening, we can see that there are bursts of action potentials

Simulation Results and Analysis: The action potential, Burst-fire



And when we increase the pulse amplitude from 0.4 to 4, we see that there's more peaks and the height of each peak is lower, so the voltage threshold changes with time when a strong current is injected.

Simulation Results and Analysis:
The action potential, Burst-fire Frequency as a Function of Pulse Strength



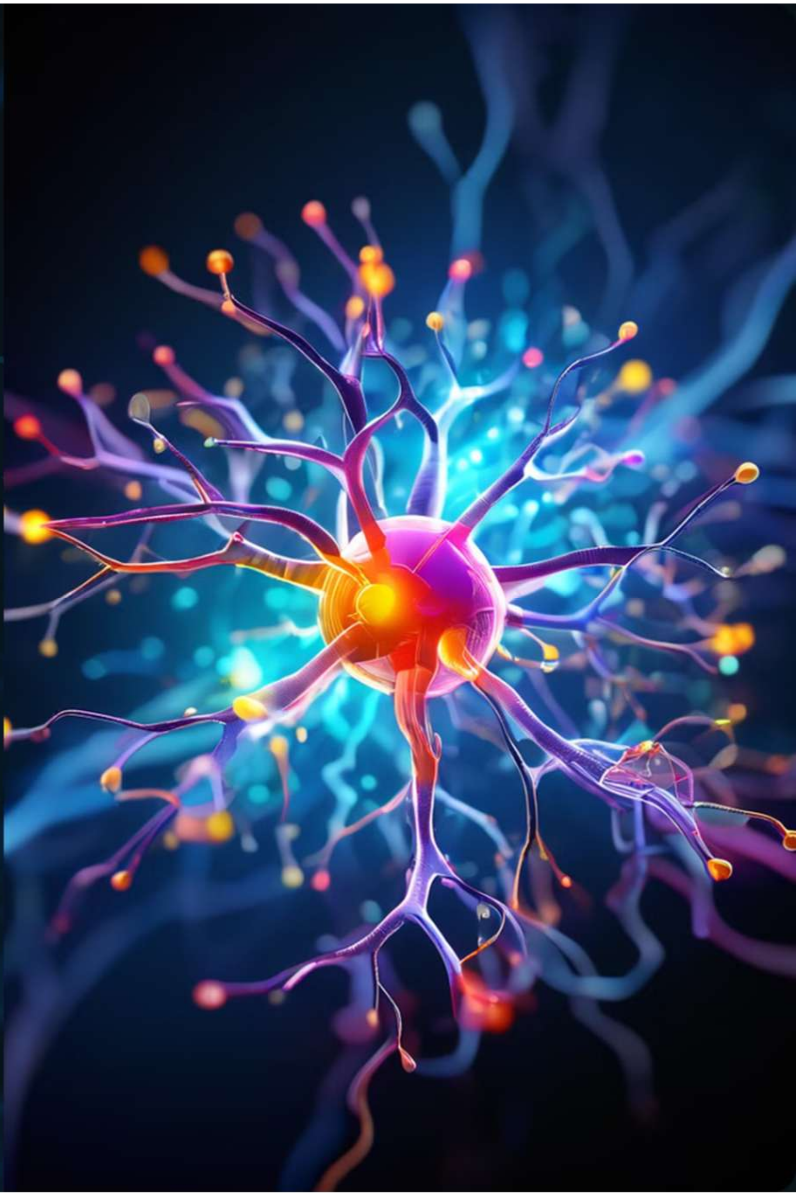
Here I stayed with the 50msec I used before, and we can see here that each point represents the frequency fire-rate of the neuron when it is injected with different pulse strength

Simulation Results and Analysis: Refractory Period

The refractory period is the time after an action potential during which is less likely to fire another action potential.

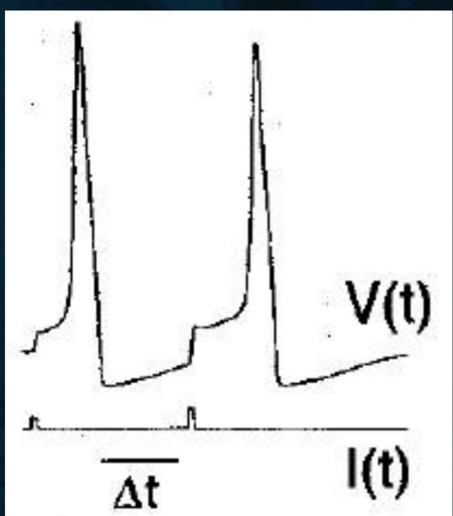
This is an important mechanism for regulating neuronal firing rate and in order to prove that there is a refractory period I'll assume that there is no refractory period and run some tests in the attempt find a contradiction.

So, if I assume there is no refractory period then the neuron should fire again after the first action potential just as likely every time I try to inject two currents.

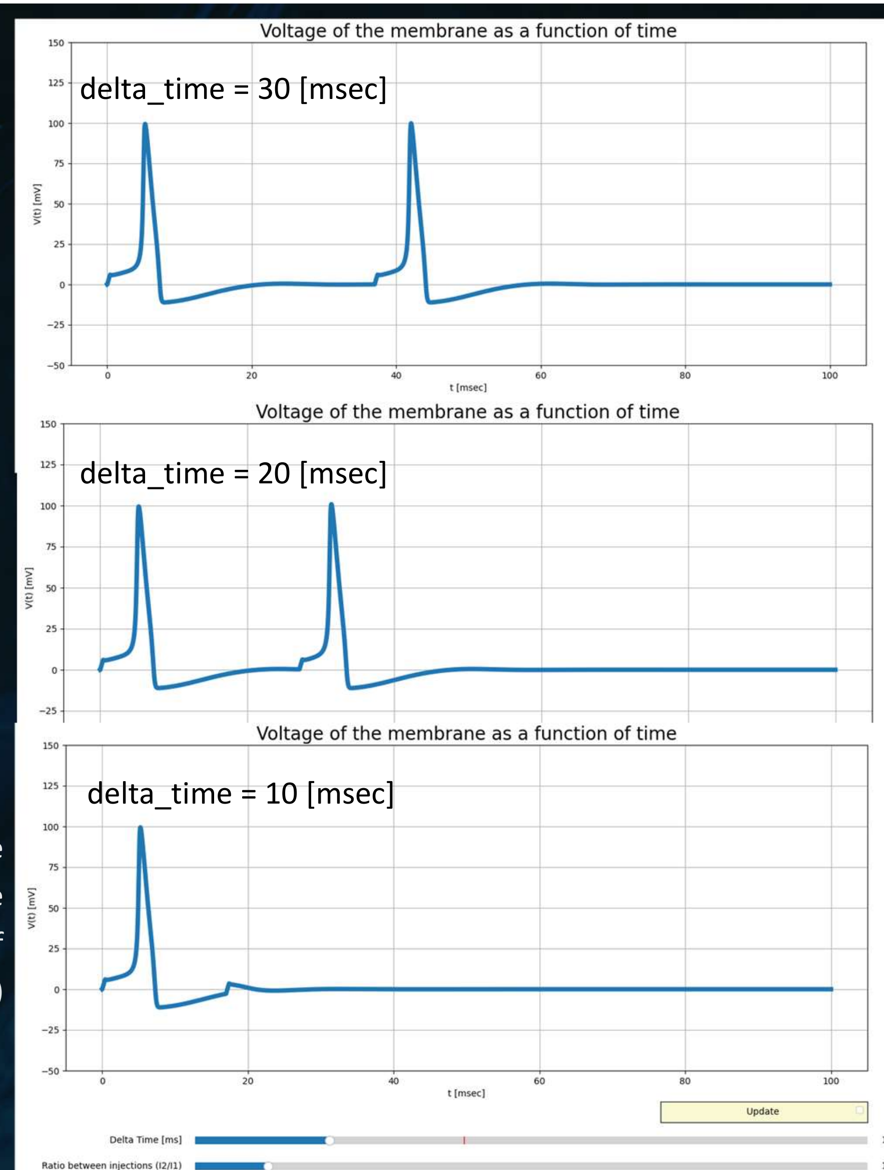


Simulation Results and Analysis: Refractory Period

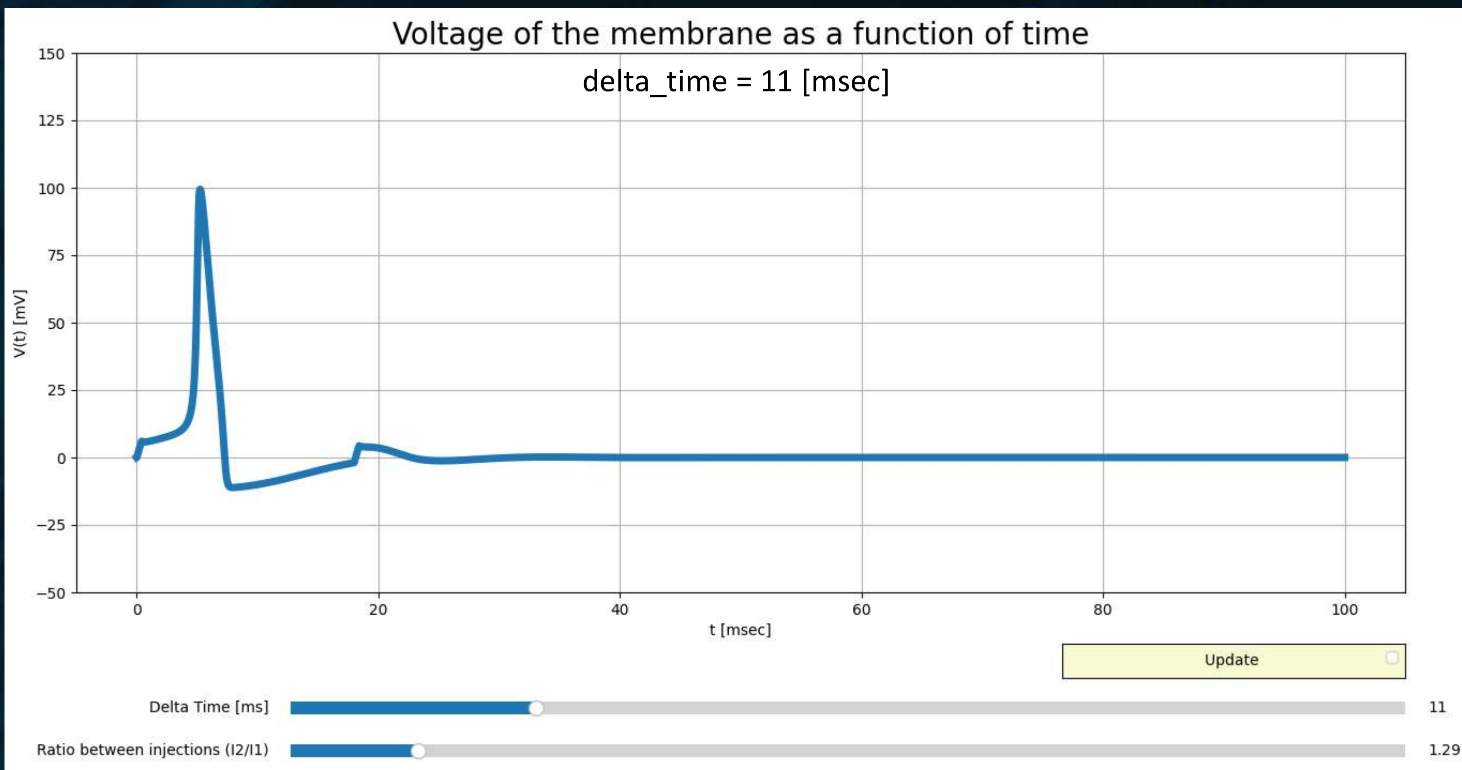
I'll start by injecting a current to the membrane, twice with some time between the injections. Koch called the time that passed from the moment when the neuron first repolarized to 0mv, until the second injection, delta time .



We can see that we get a contradiction because the "distance" (delta_time) between the injections, ultimately affects the generation of the second spike (lower right graph)

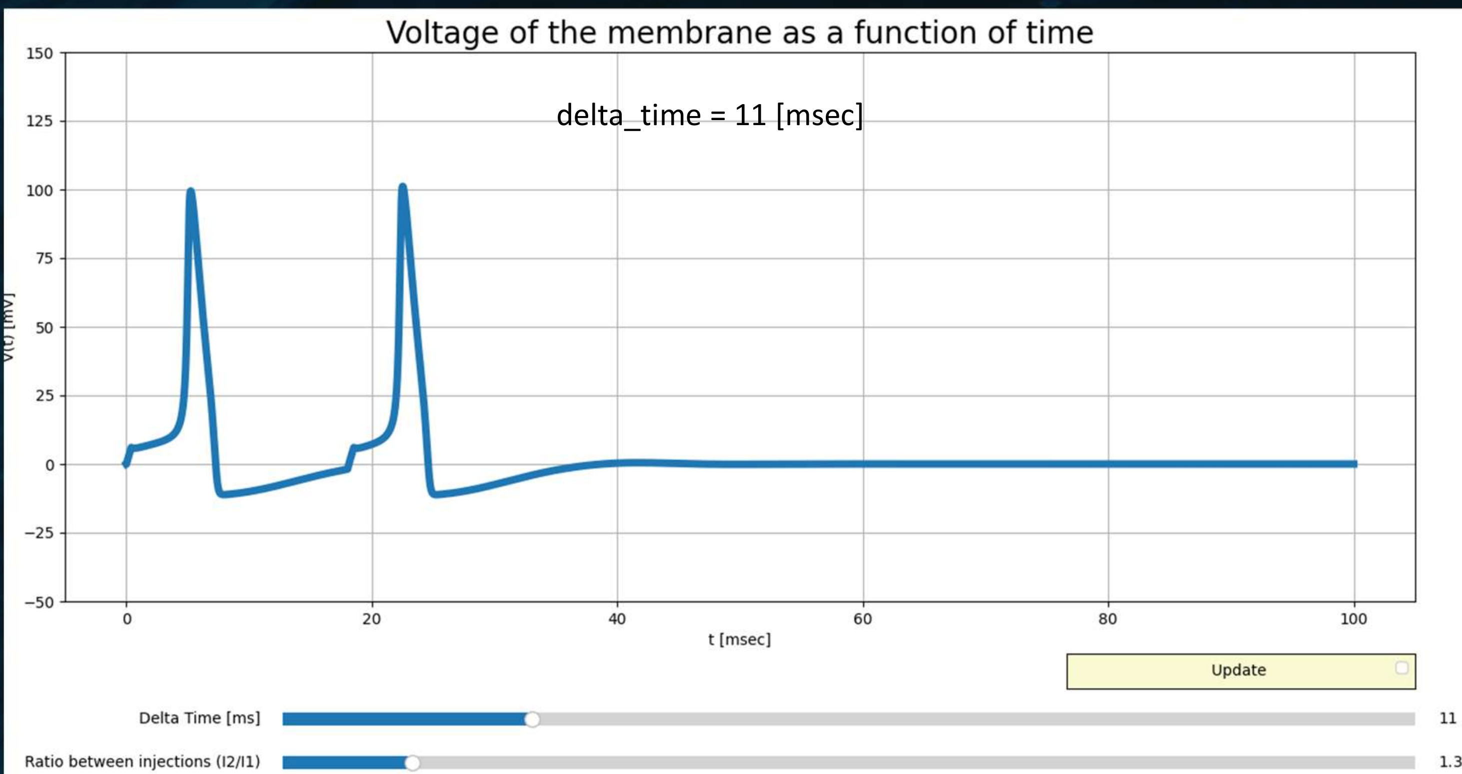


Simulation Results and Analysis: Relative Refractory Period



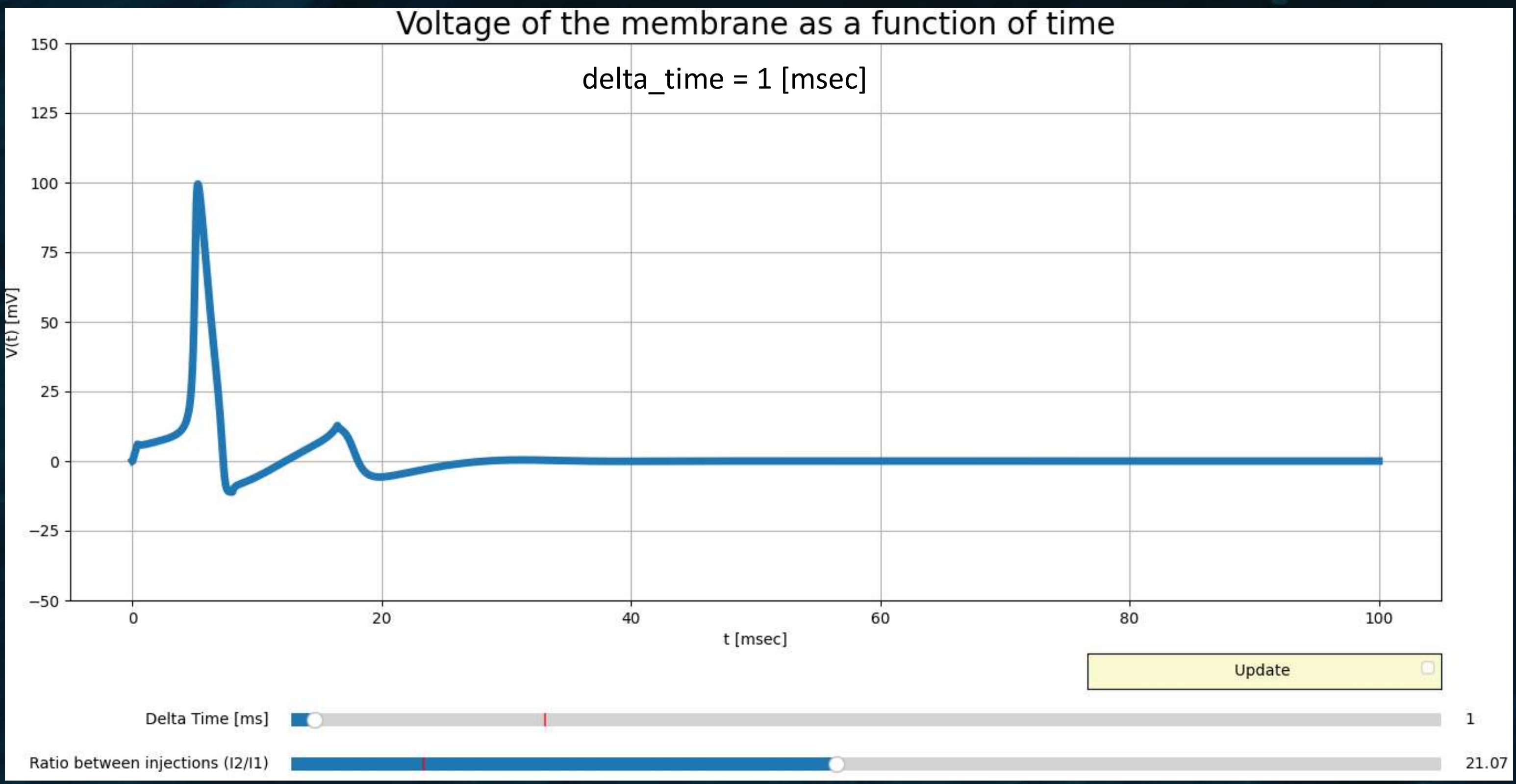
Here I am looking for the relative refractory period by going to the values provided by Koch. He says that the relative refractory period is around delta=11ms and the ratio between the spikes is spoused to be small, he had 15% difference, as you can see here my model is at 29% already and still no second spike.

Simulation Results and Analysis: Relative Refractory Period



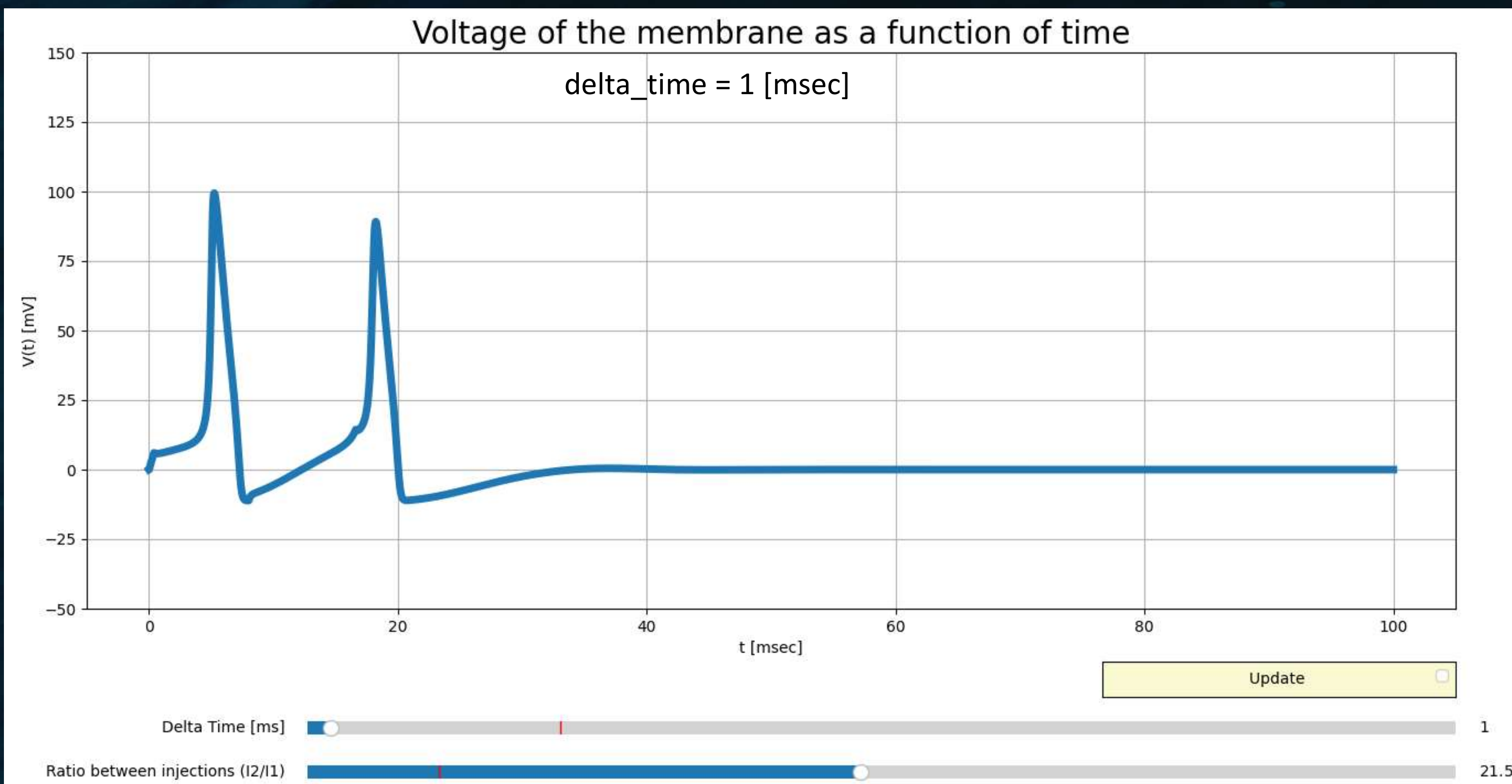
But at 30% there is another spike. This is the relative refractory when I tried to inject the same after 11ms I got no second spike when the second current became little bigger (by a third more) the action potential was generated.

Simulation Results and Analysis: Absolute Refractory Period



Now to look for the refractory period, which is supposed to accrue on low delta time, so I to be 1ms and I raised the current a lot but still no

Simulation Results and Analysis: Refractory Period



And as you can see, when the current ratio is above 21.5 then the second spike returns :)

This is the absolute refractory period because in order to induce a second spike, the second current had to be larger by 20500% which is nowhere near physiological values.

Conclusion and Comparison to Original Model

The recreation of the Hodgkin–Huxley model in Python successfully the original model's behavior. The simulation results demonstrate the model's accuracy in capturing the dynamics of action potential generation and propagation.

Although I tried to go as close as possible to the original experiments, I had some deviations (that probably came from uncounted variables of temperature and neuron sample size) , these are the main ones I noticed:

- The original pulse current that generated the first action potential was lower than mine by a factor of 33.2 (mine=13.28[nA], original=0.4[nA]).

- The original ratio that generated the second action potential was supposed to be around 15%, but I got 30%.

Thank you for reading, bey :)