Computational Model of a Two-Cell Half Center Oscillator

Austin Firestone

The University of Missouri

CMP_SC-4590: Computational Neuroscience

Modeling Project

Professor Satish Nair

12/5/2023

Computationally Modeling a Two-Cell Half Center Oscillator

Introduction

In this modeling project the author explored the workings of Central Pattern Generators (CPGs). To accomplish this, he created a biologically plausible computational model of a two-cell half center oscillator (HCO), one of the simplest CPGs.

Function

Central Pattern Generators are seen all throughout biology due to the necessity of their function. They give the nervous system the ability to produce a rhythmic output without rhythmic inputs. In the case of a two-cell HCO, upon activation by excitatory current, the first cell will fire, then the second will fire, then the first cell, and so on. They will keep alternating their bursts until they receive an inhibitory current or until they stop on their own through adaptation. Oftentimes, only one excitatory input is needed to start the chain reaction and one inhibitory input to end the chain reactions. Furthermore, in some cases the inhibitory input will not even be necessary as the cells are adaptive in nature and will cut off their rhythmic bursting after an allotted period of time. The biological applications of a two-cell HCO are endless. Usually, they are used as building blocks for more complex CPGs that contribute to functions such as walking, breathing, flying, and swimming of vertebrates (Marder E, 2001). A common example of this two-cell HCO can be found in the hearts of a leech (Calabrese et al., 1989). Leeches have bilaterally paired hearts that run along the length of their body. The rhythm of these multiple hearts is moderated by inhibitory connections, essentially forming a two-cell HCO. In Figure two, one can see a simplified model of a two-cell HCO including its expected output.

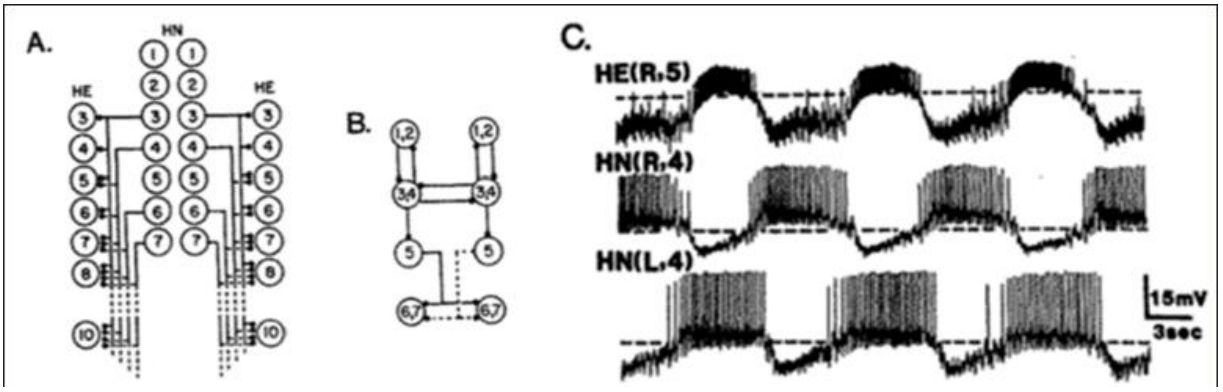

Fig. 1. Shows the synaptic connectivity of heart (HE) and motor (HN) neurons in a leech. Circles are neurons, lines are axons, and dotted lines are inhibitory connections. A and B show the circuit model for the neurons and their connections in a leech. C shows the output of these half center oscillators. (Calabrese et al., 1989)

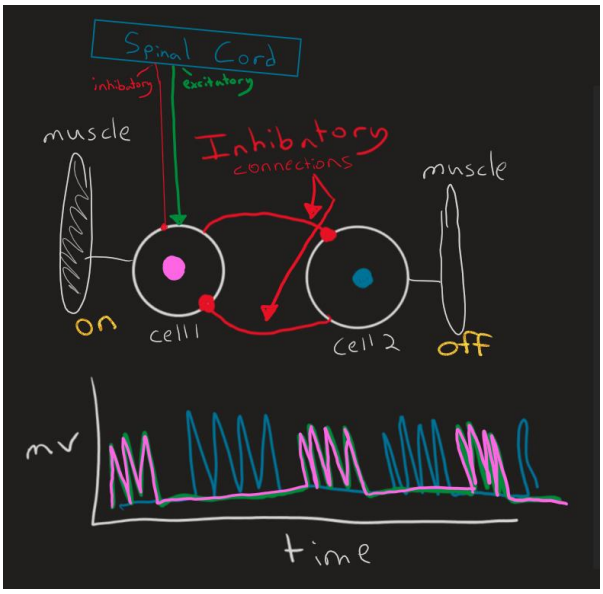

Fig. 2. Depiction of a two-cell half center oscillator including its inputs (from the spinal cord) and its outputs (to the muscles).

Biology

The "hardware" nature requires to achieve the effect described above is simply two rebound bursting neurons with mutual inhibitory connections, input pathways from the spinal cord, and output pathways to the muscle. Rebound bursting neurons are neurons that burst after a period of inhibitory stimulation ends. They accomplish this through an H gate. Furthermore, the inhibitory synapses connecting these two neurons ensure only one fires at any given moment and also play a role in the "rebound" effect. To start the oscillations all that is needed is an excitatory input from the spinal cord which will cause the first cell to fire. Then, after it is done bursting, the second cell will "rebound" and begin bursting. This process will repeat back in forth until the pair receives an inhibitory input from the spinal cord, stopping the oscillations.

Model

The author created the computational model of the two-cell half center oscillator described above using python and NEURON, a library for simulating neurons and networks of neurons. This library will allow the author to insert neurons, set conductance values for the channels and synaptic connections of those neurons, apply excitatory and inhibitory current injections, and measure the outputs of these systems. There are three laws that define how the connections between these components operate. The first is Ohm's law which describes the flow of electricity with respect to current, voltage, and resistance. The second is the Capacitor Charging Equation which describes the relationship between capacitance, charge, and voltage. This is important because the cell membrane of neurons is essentially a biological capacitor. The last is Kirchhoff's Law or the Conservation of Charge. Kirchhoff's and Ohms law play a key role in defining the nature of current injection, synapses, and channels. One can see the role these laws play in the circuit diagram below of a simple neuron with a passive channel. This schematic

is the foundation of the much more complicated system needed to create a two-cell half center oscillator.

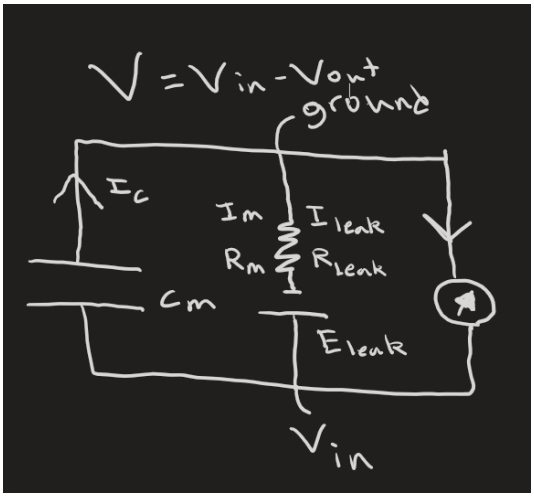

Fig. 3. Schematic of a neuron with a passive channel

Math

Upon combining the three laws above one can derive the general first order differential equation that describes the change in voltage's relationship with capacitance and current injection(A). Furthermore, using the gate probability of each channel, one can create the differential equation for the probability of that channel being open at any given voltage(B). The equation for modeling synaptic connections can be derived using a similar method(C). Neuron performs all these calculations on the backend through mod files and outputs the current given from the channel or synapse.

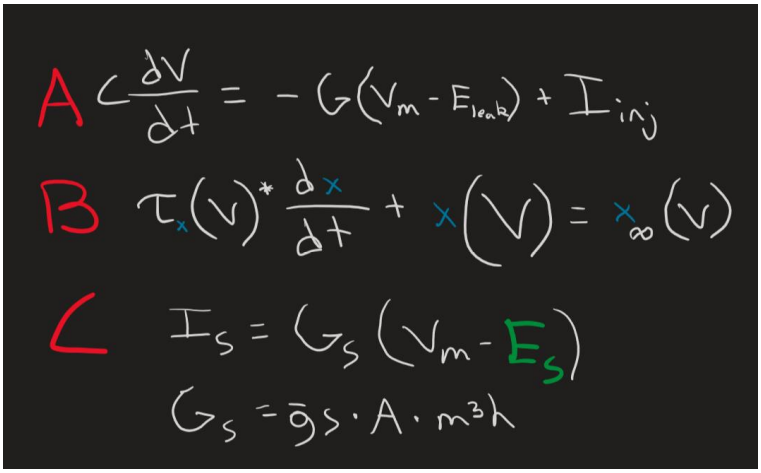

Fig. 4. A: ODE for a neuron. Add another -G(Vm-E) for each additional channel. B: ODE for a channel's activation. The variable X is replaced with the channel's name. C: An equation

describing a synapses Current given it's Conductance(from probability) and Voltage. Es determines whether the synapse is inhibitory or excitatory.

Objectives

In this modeling project the author sought to computationally model a two-cell half center oscillator. This is defined by two rebound bursting cells connected to each other by inhibitory connections. When stimulated by a small current injection, the first cell will fire, then the second, then the first, and so on. This will continue until one of the cells receives an inhibitory current from a control center. This control center is likely the spine.

Method

The method utilized by the author for building a two-cell half center oscillator began with first modeling one of the simplest neurons, a spiking neuron. This model only requires one passive channel(leak) and two active channels(Na + K). From here, the author increased the model's complexity little by little until the model progressed to the desired end result. This allowed the author to develop a better understanding of the firing patterns within the model and gave him the ability to accurately fine tune the parameters in a step-by-step manner. A flow chart describing the evolution of his model can be seen in Figure 5 below.

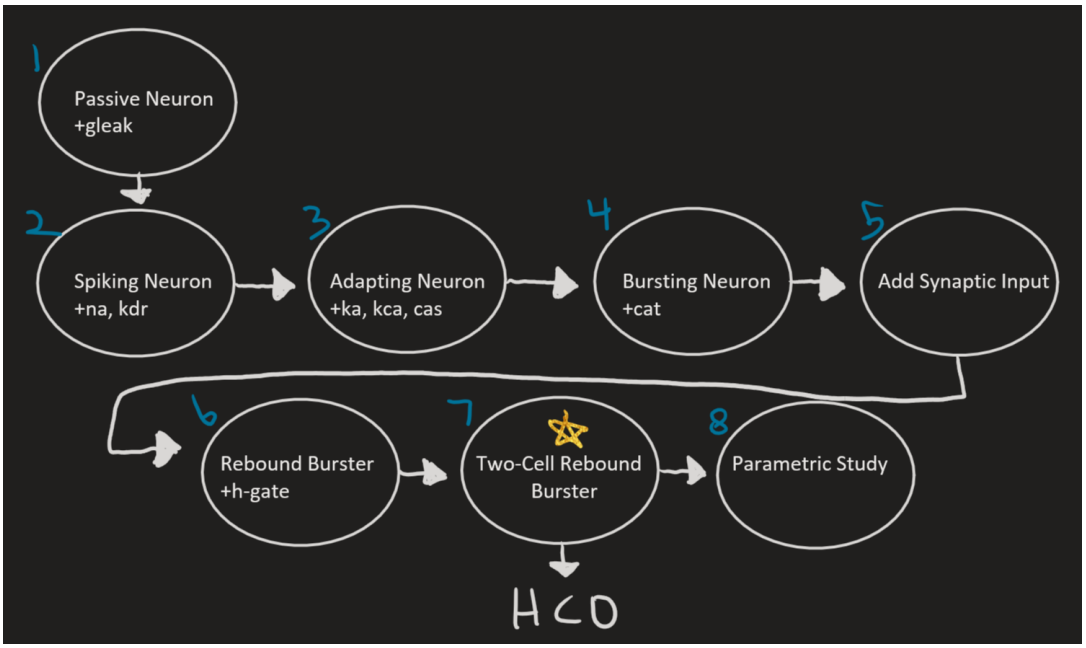

Fig. 5. Flow chart showing the evolution of the model. Below each stage one can see the channels added to achieve it. The star represents the successful implementation of a two-cell half center oscillator. Note that at stage 5 current injection was replaced by synaptic input for cellular stimulation.

A detailed description of each stage and the channels used will now be discussed.

Stage 1

The author started by inserting a simple neuron into his code. It is only defined by its passive leak channel and current injection. The passive channel acts as a resistor and the cell membrane as a capacitor. All that can be observed is the effect of Ohm's law when the current injection is increased or decreased.

Fig. 6. Create a single neuron.

Stage 2

Next the author progressed to modeling a spiking neuron. All that is needed to achieve this effect is the addition of a depolarizing channel, sodium, and a repolarizing channel, potassium. When current injection increase the membrane voltage past the threshold(Ohm's law), the sodium channel will open then close, followed by the potassium channel opening and closing. This causes an action potential to occur. To increase the frequency of these action potentials, all that's required is an increase in the amplitude of the current injection. The parameters used for this stage of the model can be seen in Figure 6.

```
Parameters:
gbar_leak = 3e-05
gbar_na = 0.1
gbar_kdr = 0.1
```

```
# Current injection parameters
ccl_param = {
    'delay': 200,    # start time (ms)
    'dur': 800,    # duration (ms)
    'amp': 0.7  # amplitude (nA)
```

Fig. 6. gbar_na is the sodium channel. gbar_kdr is the potassium channel. Note: the gbar_leak parameter is listed as "none" in the code but is assigned this small default value by Neuron.

Stage 3

The next stage involved creating an adaptive neuron. The adaptive neuron halts its action potentials after a given amount of time. It works by adding in three channels. Ka, Kca, and Cas. Cas is a calcium channel that allows calcium to slowly build up with each action potential, Kca activates when enough calcium has built up and repolarizes the cell, stopping the action potentials. This allows for an adaptive response. The Ka channel is more rapid and helps control the timing of action potentials. To adjust the strength of the adaptive response all one needs to do is adjust the value of Kca to change the calcium threshold or adjust the value of Cas to increase or decrease the rate of calcium build up.

```
# Biophysical parameters
parameters = {
    # Conductance of channels (siemens/cm2)
    'gbar_leak': None,
    'gbar_na': .1,    # (.1~.5)
    'gbar_kdr': .1,    # (.1~.5)
    'gbar_ka': 0.1,    # (.1~.5)
    'gbar_kca': 0.015,    # (.01~.05)
    'gbar_cas': 0.001,    # (.001~.01)
```

Fig. 7. Parameters for an adaptive neuron.

Stage 4

Now the author transitions to creating an adaptive bursting cell. He accomplishes this by adding a Cat channel, which allows the cell to bring in more calcium with each spike, lowering the action potential's refractory period. It does this by using calcium to keep the membrane threshold at an excitable state where sodium channels can easily open. This process is terminated when the calcium levels reach a certain threshold and the Kca channel opens repolarizing the cell, giving the neuron its adaptive nature. In the code, a base Cat channel was added, and the current injection was increased to 2nA.

```
# Biophysical parameters
parameters = {
    # Conductance of channels (siemens/cm2)
    'gbar_leak': None,
    'gbar_na': .1,    # (.1~.5)
    'gbar_kdr': .1,    # (.1~.5)
    'gbar_ka': 0.1,    # (.1~.5)
    'gbar_kca': 0.015,    # (.01~.05)
    'gbar_cas': 0.001,    # (.001~.01)
    'gbar_cat': 0.005,    # (.005~.01)
    'gbar_hyper': 0,    # (.0001~.0003)
    # Ca pool parameters
    'tauca_capool': None,    # decay time constant
    'fca_capool': None    # ca influx factor that goes to ca pool
} # parameters set to "None" are to use default values defined in the template

# Current injection parameters
ccl_param = {
    'delay': 200,    # start time (ms)
    'dur': 800,    # duration (ms)
    'amp': 2   # amplitude (nA) (set to 0 to disable current injection)
}
```

Fig. 8. Parameters for adaptive burster neuron.

Stage 5

The author now switches from stimulating his cell with current injection to synaptic input. The main difference from the synaptic model is that the current injection is no longer coming from a direct source(current clamp) but from other neurons/synapses. When switching to this model in Collab the author noticed the neuron did not spike or burst, all that could be observed were small rhythmic increases in the membrane potential that corresponded to the synapses interval(time between spikes). He assumes the neurons docile nature is due to the synapse's small conductance value. A model with several synapses could likely display the same effects as a current injection of 2nA(value used in burster). Furthermore, the synaptic channels

reversal potential is what decides whether the synapse is excitatory or inhibitory. The final model should use an inhibitory channel so the two neurons can oscillate(alternate spiking).

Stage 6

In this stage the author created a single cell rebound burster. This was tricky because he had to utilize the H-gate. The h-gate allows slow depolarization of the cell while the hyperpolarizing inhibitory signal from the other neuron is active and is stopping the rebound burster from spiking. This slow depolarization eventually leads to a reversal and is why two neurons of this configuration can "oscillate". The h-gate cannot act alone though, it needs the hyperpolarizing current from the other neuron as well as the correct calcium concentrations to help it get the cell past the spiking threshold(Cas) and then burst(Cat). For his model the author added a h-gate and slightly increased the conductance of Cas(to help with the reversal). Increase Cas too much and the neuron will endogenously burst, which one does not want. Put Cas too low and the H-gate won't be enough to rebound the membrane potential and cause a burst. Get it just right and the Cas channel will work with the h-gate to create a rebounding effect. Furthermore, the author manually adjusted the spike number and interval values from the synaptic input to mimic an inhibitory bursting signal(It is a single cell model). His parameters can be found in Figure 9.

```
# Biophysical parameters
parameters = {
    # Conductance of channels (siemens/cm2)
    'gbar_leak': None,
    'gbar_na': .1,    # (.1~.5)
    'gbar_kdr': .1,    # (.1~.5)
    'gbar_ka': 0.1,    # (.1~.5)
    'gbar_kca': 0.015,    # (.01~.05)
    'gbar_cas': 0.003,    # (.001~.01)
    'gbar_cat': 0.005,    # (.005~.01)
    'gbar_hyper': 0.0001,    # (.0001~.0003)
```

```
# Synaptic input parameters
syn_param = {
    'weight': 1,    # synaptic weight (set to 0 to disable, 1 to enable synapse)
    'start': 200.,    # start time of first spike (ms)
    'interval': 50.,    # time between spikes (ms)
    'number': 5,    # number of spikes
    'esyn': -80,    # synaptic channels reversal potential (mV) (set to -80)
    'gmax': 40e-3,    # synaptic channels maximum conductance (uS) (default: 40e-3)
    'tau1': 10,    # rise time (ms) (default: 10)
    'tau2': 20    # decay time (ms) (default: 20)
}
```

Fig. 9. Parameters for a single cell rebound burster.

Stage 7

In stage 7 the author builds his final model of the two-cell half center oscillator. To build the model, he first duplicated the rebound burster cell described above using the same parameters found in Figure 9. Then he connected the two neurons via an inhibitory synapse with the parameters seen in Figure 10. Finally, he applied a quick current injection(50ms) to one of the rebound bursters to simulate the initial excitatory current that kick starts the oscillations. The parameters can be found in Figure 11.

```
# Synapse parameters
syn_param = {
    'esyn': -80,    # synaptic channels reversal potential (mV)
    'gmax': 40e-3,    # synaptic channels maximum conductance (uS) (default: 40e-3)
    'tau1': 10,    # rise time (ms) (default: 10)
    'tau2': 20    # decay time (ms) (default: 20)
}
```

Fig. 10. Synaptic parameters for the two-cell half center oscillator. Note: It is important for the reversal potential to be low(-80mV) to create an inhibitory effect.

```
# Current injection parameters
ccl_param = {
    'delay': 200,    # start time (ms) 100
    'dur': 50,   # duration (ms) 10
    'amp': 2    # amplitude (nA) (set to 0 to disable current injection)
}
```

Fig 11. Parameters for simulating an initial excitatory current (from the spinal cord).

In stage 7, the only parameter the author had to change was the duration of current injection. There was a much larger number in the previous stages(800ms) because the focus was more on bursting and inhibitory synapses rather than oscillations. Now with two neurons, a short period of current injection is all that's required, just enough to kick start the process. To utilize a shorter duration of current injection or weaker amplitude, all that is necessary(within reason) is to strengthen the synapse conductance(gmax) of the two neurons and the oscillator will work just the same. Furthermore, there are no new channels required to make this transition from a single cell rebound burster to a 2 cell HCO. All that's required is a second single cell rebound burster and a strong enough synapse connecting the two. Now when one neuron fires, it inhibits the other while its bursting. And when it stops bursting, the H-gate causes the rebound effect in the other neuron, starting the bursting process for it. This process continues repeatedly until an external signal(inhibitory) stops the oscillation.

Results and Discussion

Results

The first meaningful results were found in Stage 6 upon the creation of the rebound burster. Figure 12 shows the outcome of the parameters described. One can observe a period of inhibitory current, marked by a dip in the membrane voltage, followed by a rebound burst shortly after. Furthermore, the importance of the H-gate's slow depolarization of the cell can be seen in Figure 13 where the H-gate has been removed. One can observe a period of inhibitory current followed by no rebound bursting behavior. Finally, in Figure 14, one can see the reliance rebound bursters have on synaptic input. The synaptic input is turned off and as a result the rebound bursting neuron shows no activity.

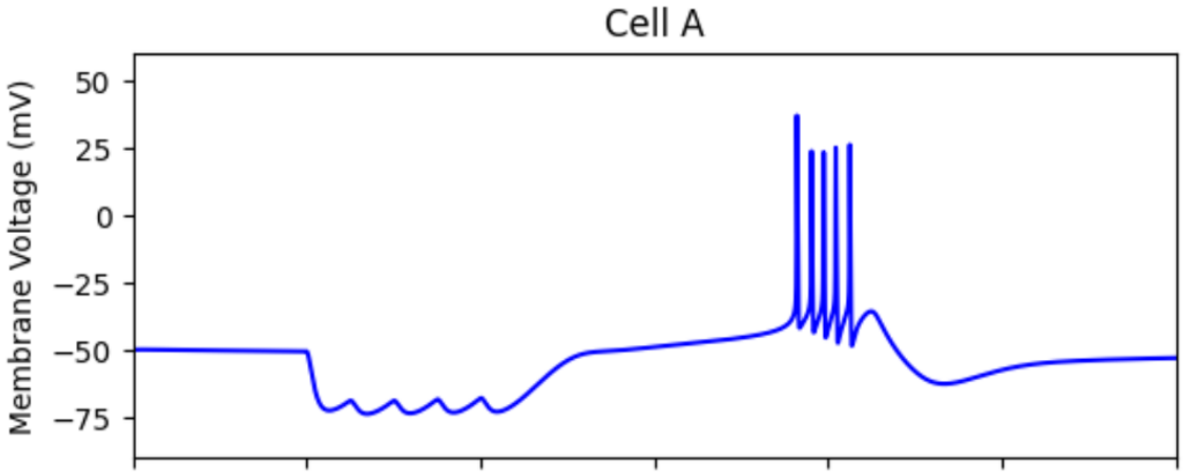

Fig. 12. Functional rebound bursting neuron.

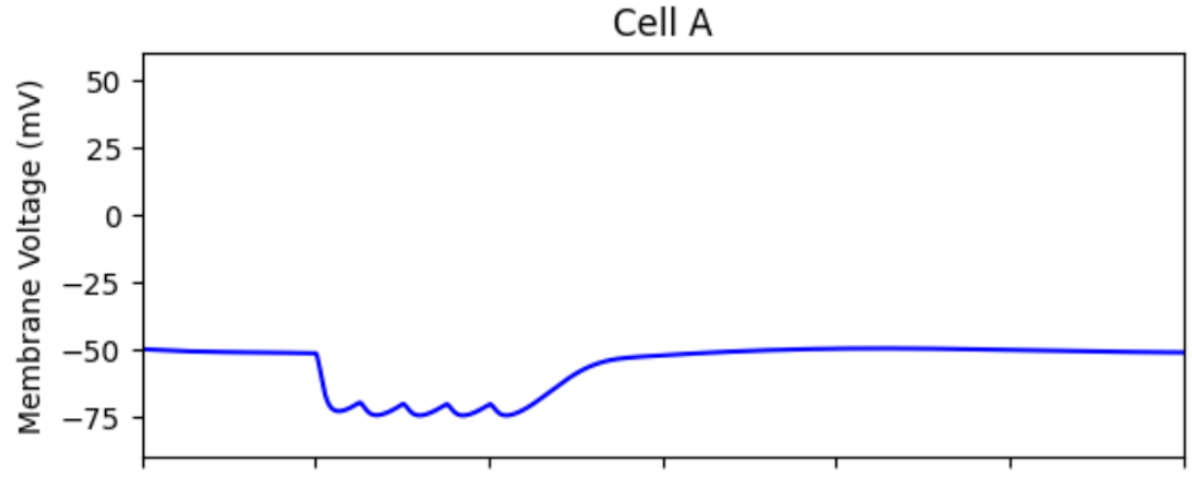

Fig. 13. Rebound burster with removed H-gate.

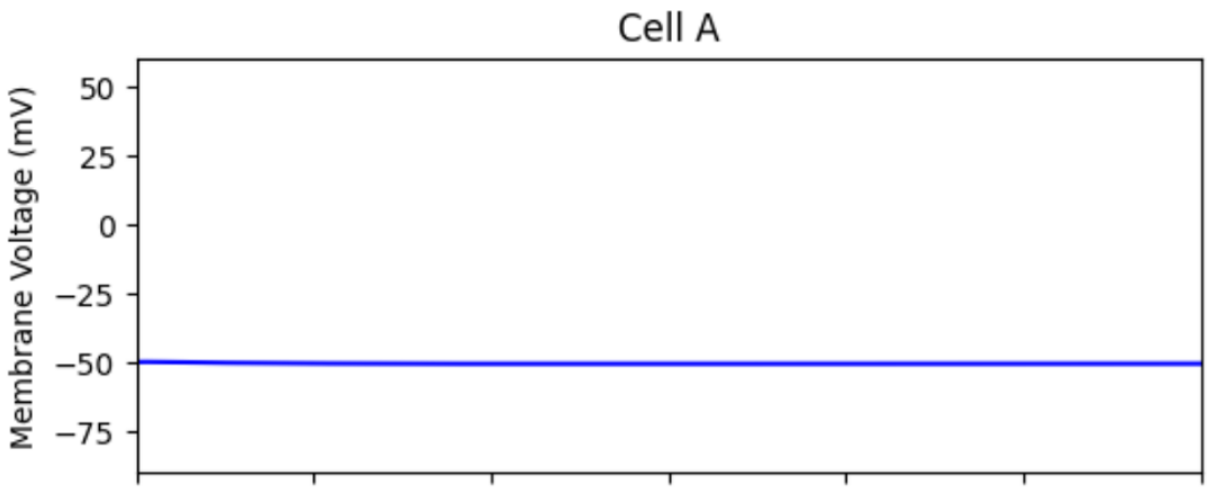

Fig. 14. Rebound burster with no synaptic input.

As described in the Method section above(Stage 7), all that's required to move from a rebound burster to a two-cell half center oscillator is the presence of two rebound bursters connected by inhibitory synapses and a short excitatory current injection to start the oscillations.

The results from stage 7 were promising. One can observe the oscillations of bursts in the two-cell half center oscillator. The depicted graph can be seen below in Figure 15.

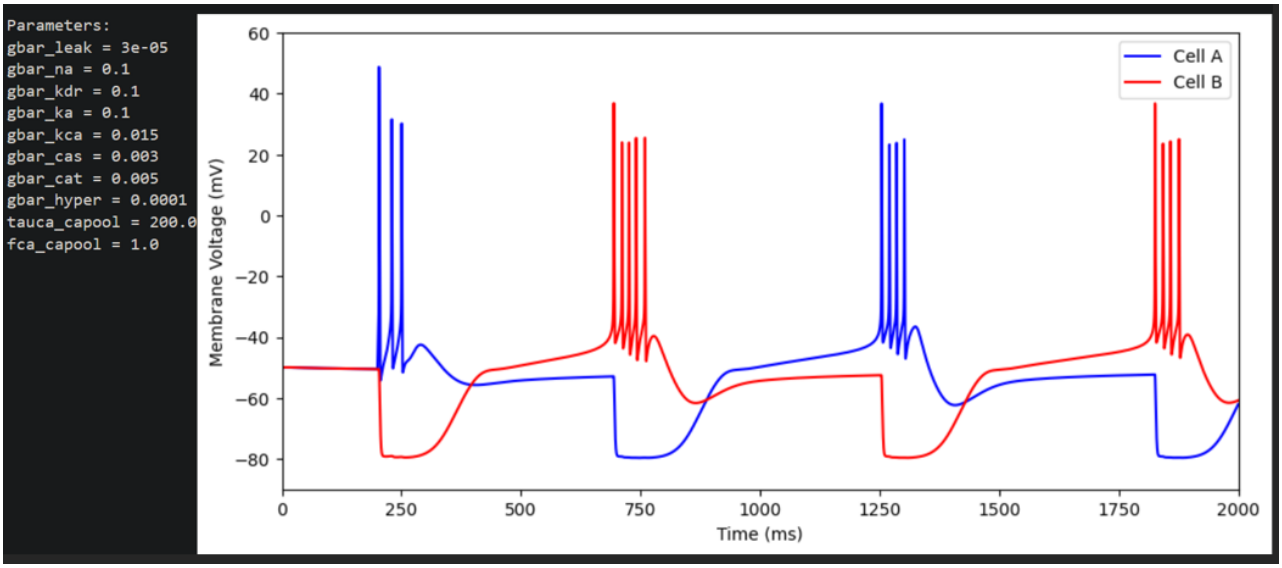

Fig. 15. Results from the two-cell half center oscillator (comprised of two rebound bursting neurons).

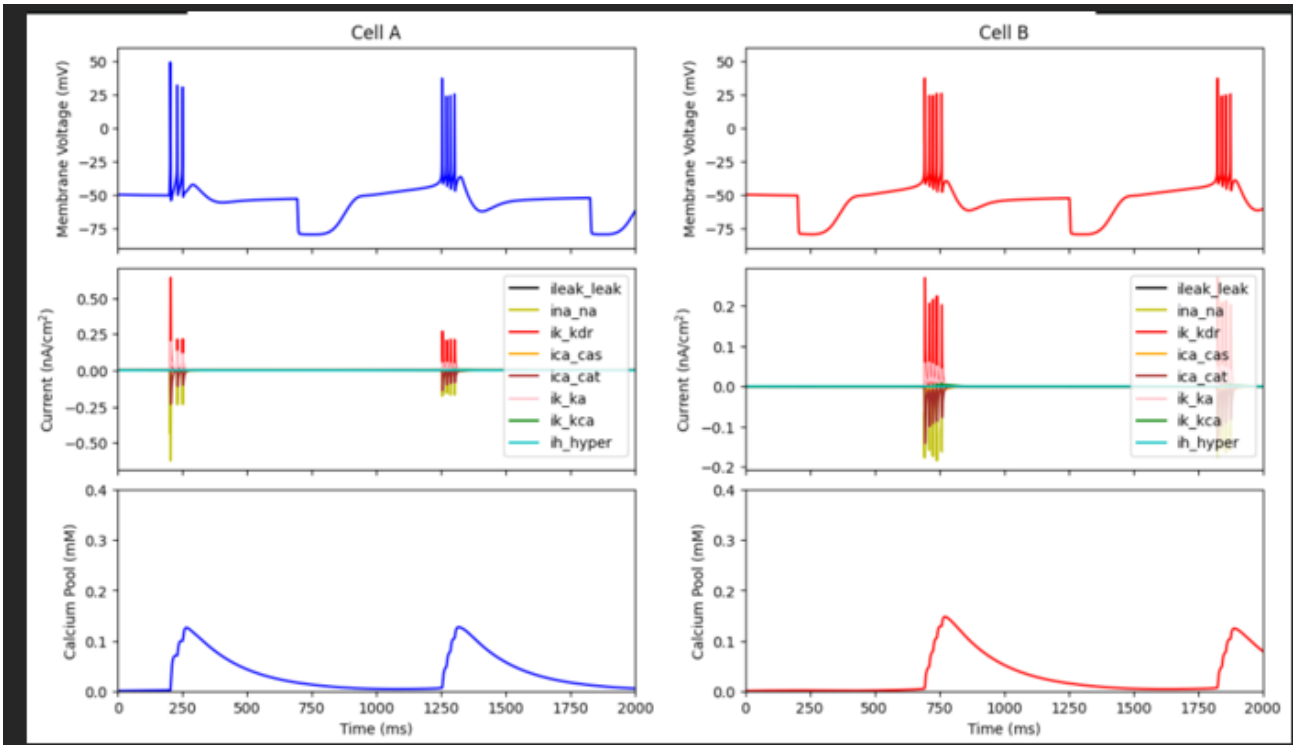

Fig. 16. Data on the two rebound bursting neurons comprising the two-cell HCO. From top to bottom one can see membrane voltage, current(from each channel), and calcium pool concentration for both neurons.

Parametric Study

The first variable examined was the relationship between current injection and the output of the two-cell HCO. The results showed that the model was very competent at predicting the current(from the spinal cord) injection's ability to trigger the bursting effect needed for functional oscillations. At the duration of 50ms the minimum amplitude required kick start the two-cell HCO was 0.45nA. The results from the injection of 0.44nA can be seen in Figure 17. Furthermore, the author found that weak current injections could be still overcome the bursting threshold if they were held for a longer duration. In Figure 19 one can see a successful two-cell HCO with a current injection of 0.1nA over 100ms.

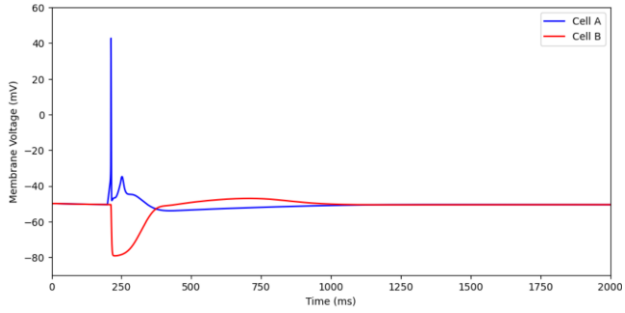

Fig. 17. Response of a two-cell HCO with 50ms of 0.44nA current injection

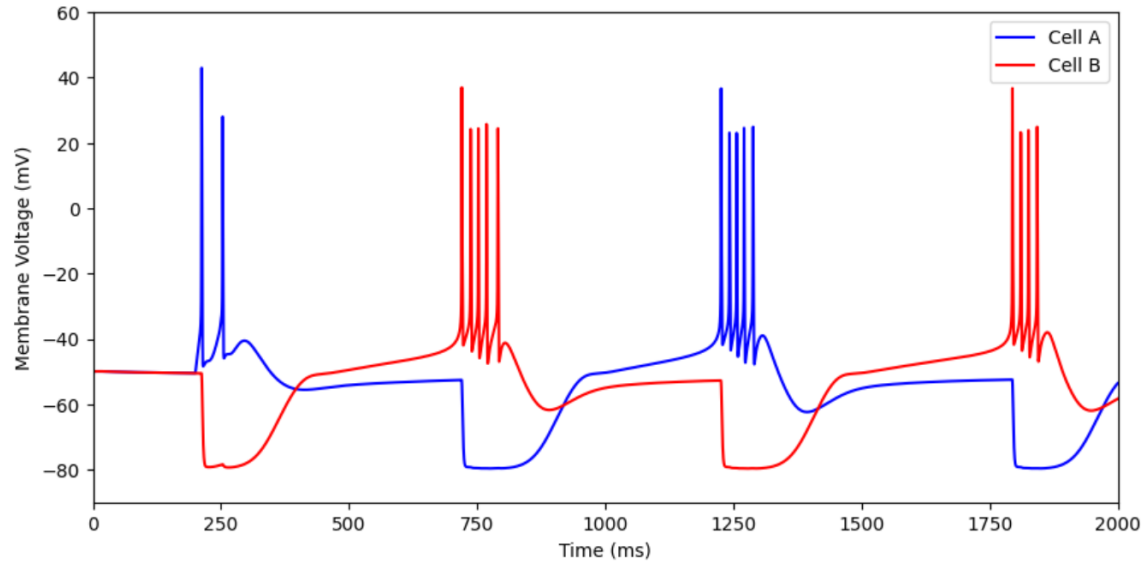

Fig. 18. Response of a two-cell HCO with 50ms of 0.45nA current injection

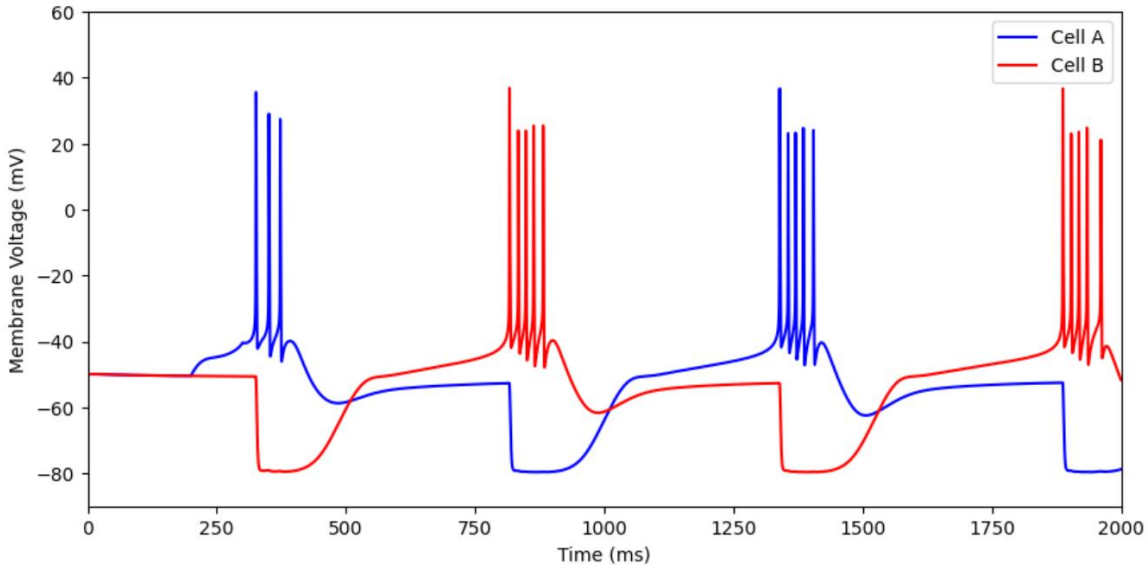

Fig. 19. Response of a two-cell HCO with 100ms of 0.1nA current injection

The second variable examined was the synaptic reversal potential's role on the output of the two-cell half center oscillator. When increasing the reversal potential to -70mV, no oscillations were observed. However, when decreasing the reversal potential to -100mV, the oscillations occur at a higher frequency. The author believes that this occurs because the reversal potential is directly related to the Nernst potential of Cl- in inhibitory synapses. Decreases in Es results in more Cl- ions entering the cell, resulting in a stronger inhibition that shortens the "rebound" time. Increasing Es results in fewer Cl- ions entering the cell, weakening the inhibitory current and

stopping the “rebound” effect from happening. One can observe this in the graphs below by looking at the magnitude of valleys with respect to voltage.

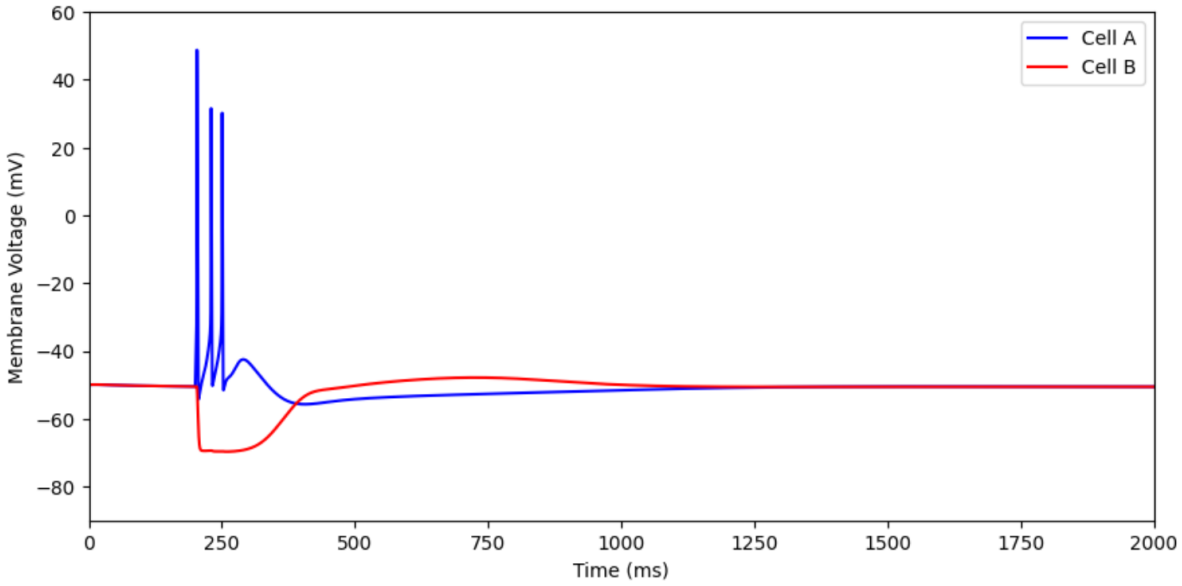

Fig. 20. Two-cell HCO with a synaptic reversal potential of -70mV

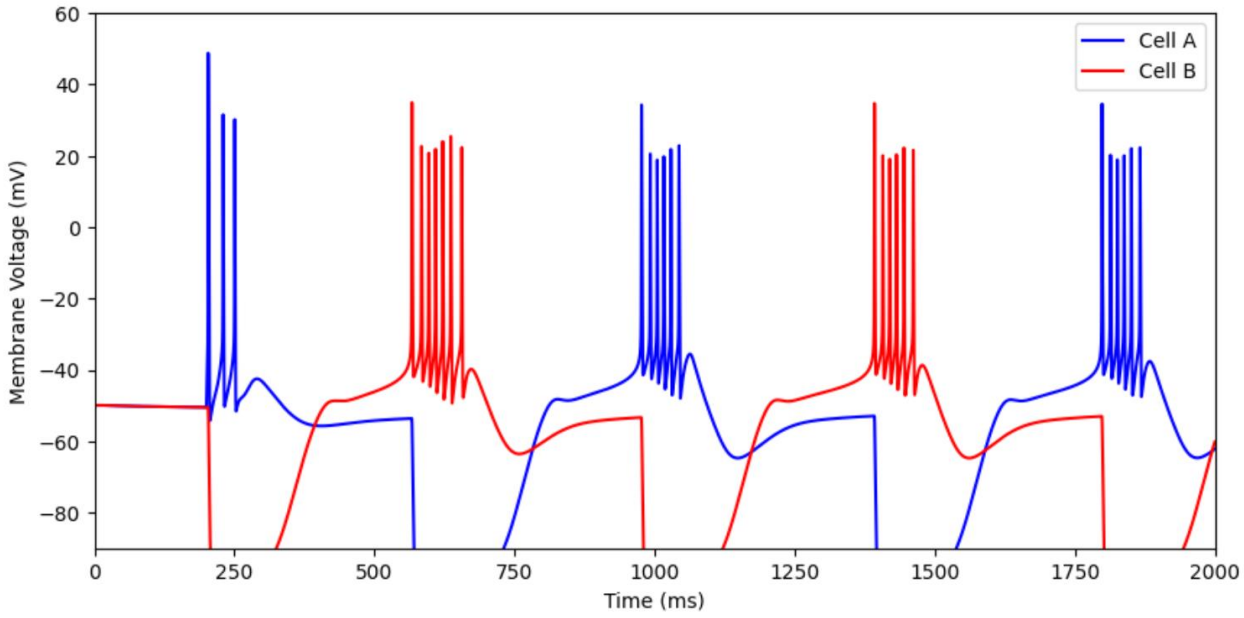

Fig. 21. Two-cell HCO with a synaptic reversal potential of -100mV

The third variable examined was the Cas channel. Changing its value showed the model’s predictive ability on how calcium concentrations influence a two-cell half center oscillator. Increases in conductance led to increases in burst frequency as well as increases in action potential frequency within the bursts. The author believed this occurred because of the nature of

the Cas channel. Increasing its conductance allows more calcium to enter the cell with every action potential, shortening the time it takes Kca to activate and lowering the action potentials refractory period, allowing for more action potentials per burst.

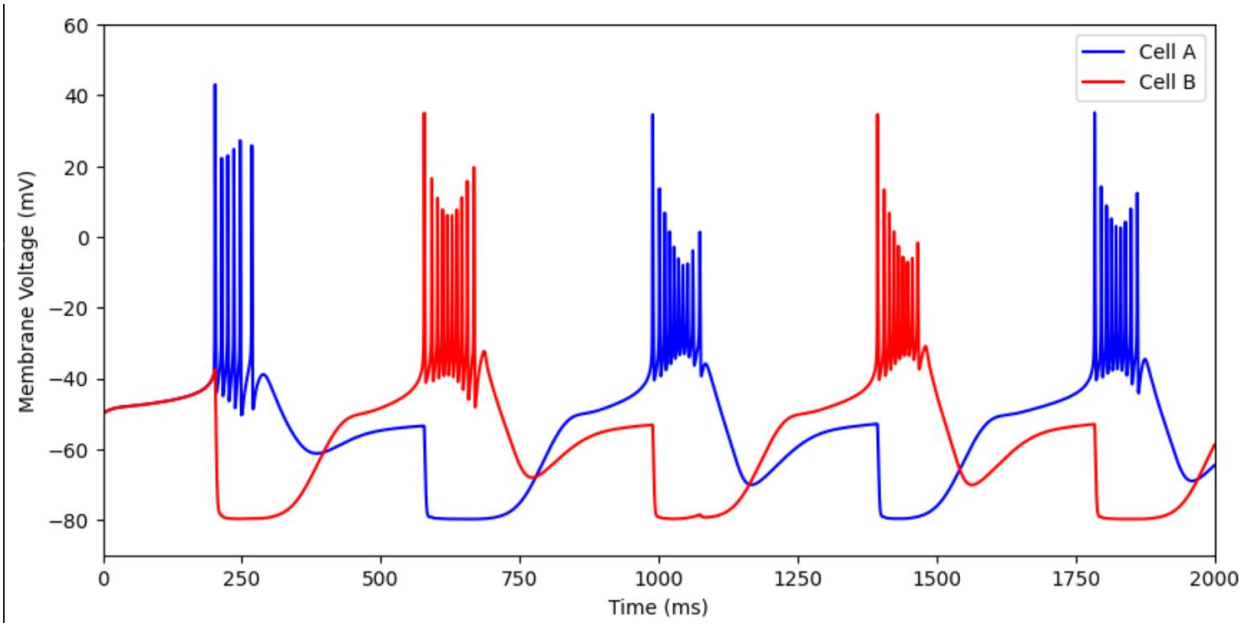

Fig. 22. Two-cell HCO with a doubled conductance value in its Cas channel (0.006 siemens/cm^2)

The final variable examined was the H-gate. Increase in its conductance increased the frequency of the oscillations between neurons. The author believes this demonstrates the H-gates essential role in the rebound function that both neurons possess.

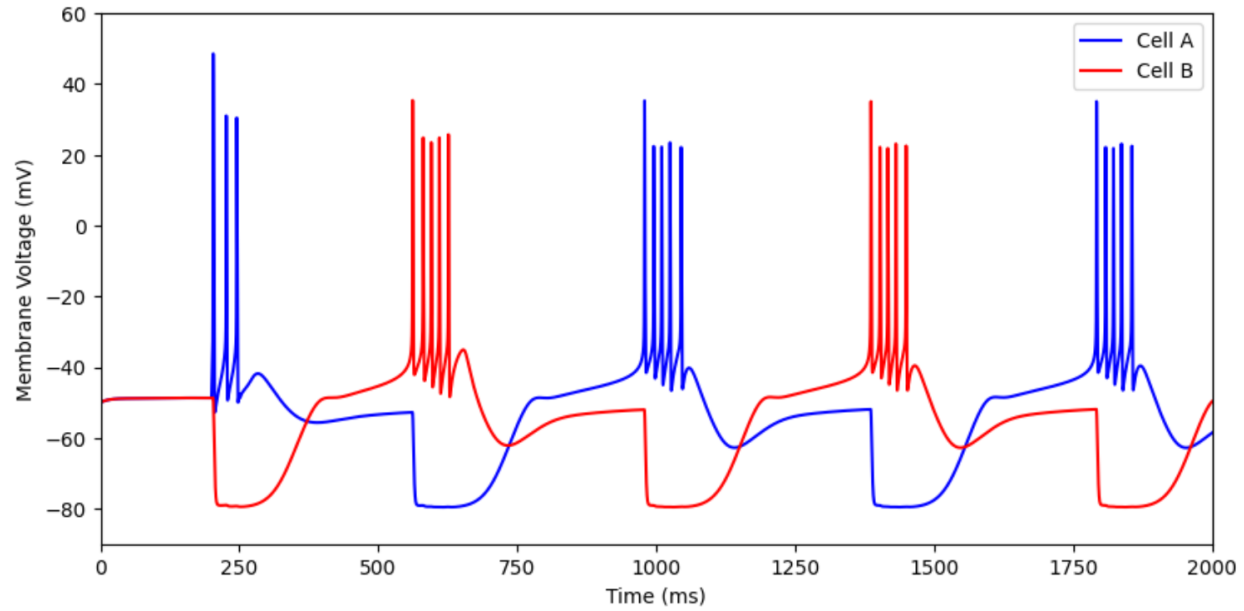

Fig. 23. Two-cell HCO with a tripled conductance value in its H-gate (0.0003 siemens/cm^2)

Discussion

The key aspects of the author's model are the two rebound bursting cells, their inhibitory synapses, and the initial excitatory current injection.

Rebound Bursting Neuron. One assumption made during the creation of the rebound bursting cells was that a neuron had only 8 types of channels. In reality, neurons have many more, however, they still demonstrate results similar to the model suggesting that maybe biologically driven rebound bursters only have 8 essential channels. One problem encountered during the creation of the rebound neurons was preventing the creation of endogenous bursters. An endogenous burster bursts on its own, without a "rebound" or initial current injection. The author had to find the "goldilocks" number for the Cas channel that allowed it to work together with the H-gate to induce the rebound effect, but also not be high enough that the neuron bursts on its own. Biologically, the results make sense. Research establishes biological outputs that are very similar to computational ones seen in this paper (Sharp et al., 1996).

Inhibitory Synapses. One assumption made during the creation of the inhibitory synapses was the values for tau, or the time constants. The author set tau1, defining the rise time, as 10ms and tau2, defining the decay time, as 20ms. Changing these synaptic values can greatly change the behavior of the two-cell half center oscillator. One challenge faced during the creation of the inhibitory synapses was interpreting the value of gmax and its effect on the output of the two-cell HCO model. It appears to be a sensitive metric with predictable changes only happening near the default value (40e-3 uS). Biologically, the inhibitory synapses are very close to reality. Their reversal potential is set at -80mV, a value close to the Nernst potential of Cl-, the ion primarily responsible for inhibitory currents.

Current Injection. An assumption made during the initial excitatory current injection (that signaled the two-cell HCO model to start) was the values for amplitude and duration of the current injection. The author assumed the values to be 2nA and 50ms. A problem encountered pertained to exactly this as the author did not have accurate data on what the parameters for an excitatory current from the spinal cord would be. To solve this, he engaged in a parametric study and found that the values for amplitude and duration were not significant in affecting the output of the two-cell HCO as long as changes in one led to an inversely proportional change in another. For more details on this refer to the parametric study. Biologically, the current injection is plausible because the "kick start" mechanism for CPG often comes from the spinal cord.

Conclusions and Impact of Research

In summary, a two-cell half center oscillator can be modeled in python with only a handful of channels and two inhibitory synapses. The model successfully predicts the outcome of a small excitatory current injection that, if past the threshold, will start the domino and display the same oscillating bursts one would see in biological two-cell half center oscillators. Limitations to this computational model include the assumptions described in the Discussion section as well as the model's simplicity. One can see from the inhibition based neural

oscillations observed in the anatomy of leeches that even simple organisms, much less humans, have central pattern generators that are significantly more complex than the two-cell model shown in this paper (Calabrese et al., 1989).

The first conclusion derived from this computational model was that every parameter included in the rebound burster, inhibitory connections, and current injection had the power to crucially change the outcome or halt the outputs of the two-cell HCO.

The second conclusion was derived from the parametric study and found that only a short excitatory current was required to induce the oscillating bursts seen in the two-cell HCO model. Furthermore, it was found that as long as changes in one metric of the injection was met with an inverse change in the second metric (i.e. amplitude decrease paired with duration increase), the two-cell HCO model was hardly affected.

The third conclusion was that the inhibitory synapses had to have a threshold value of inhibition to induce the rebound effect. Increasing the reversal potential of these synapses by just 10mV led to drastic changes in the model's output. Also, once past the threshold value, continued decreases in reversal potential led to increases in burst frequency.

The fourth conclusion found was that calcium concentrations play a critical role in determining burst frequency and action potential frequency within each burst. This can be seen in the exploration of the Cas channel in the parametric study.

The fifth conclusion was that the H-gate is the primary inducer of the rebound effect in rebound bursting neurons. The slow depolarization despite inhibition is the driving force behind the oscillations seen in the two-cell HCO model.

The impact of this research is significant because despite the proposed model being a simple two cell application, it displays the workings of many neurological functions seen in nature. For instance, the alternating flexing and contracting of a fish's muscles on the side of its body is what propels it through the water. Furthermore, the rhythmic opening and closing (flexing and contracting) of heart valves is what pumps blood through mammal's bodies. This model can be extrapolated and used to better understand how these functions arise. Also, this model could possibly be used as a building block in a much larger model that seeks to better explain the functions described above, and others. This is already being done by several researchers (Doloc-Mihu and Calabrese, 2011). The author is optimistic that the mastery of simple models like the one shown in this paper can be used as a foundation to help researchers explore the depths of neuroscience and better understand the inner workings of Central Pattern Generators.

Appendix

The code used to create this model can be found below.

Build Cell Template:

```
from neuron import h
import matplotlib.pyplot as plt

class HCOcell(object):
    def __init__(self):
        self.create_sections()
        self.define_geometry()
        self.define_biophysics()
        self.setup_record()

    def create_sections(self):
        """Create the sections of the cell."""
        self.soma = h.Section(name='soma',cell=self)

    def define_geometry(self):
        """Set the geometry of the cell."""
        self.soma.nseg = 1 # create only one segment in the soma
        # gives area of .314e-3 cm^2
        self.soma.L = 1000 # (microns)
        self.soma.diam = 9.99593 # (microns)

    def define_biophysics(self):
        """Initialize the membrane properties of the cell."""
        self.soma.cm = 1 # Membrane capacitance (microF/cm2)

        self.soma.insert('leak')
        self.soma.eleak = -50 # (mV)

        self.soma.insert('na')
        self.soma.ena = 50

        self.soma.insert('kdr')
        self.soma.ek = -80
```

#use the neuron package to create a neuron and add channels

```
        self.soma.insert('capool')
        self.soma.cao = 3 # (mM)
        self.soma.cai = 50e-6 # (mM)

        self.soma.insert('cas')
        self.soma.insert('ka')
        self.soma.insert('kca')
        self.soma.insert('cat')
        self.soma.insert('hyper')
        self.soma.eh = -20

        self.default_parameters = {
            'gbar_leak': .03e-3, # (siemens/cm2)
            'gbar_na': .1,
            'gbar_kdr': .1,
            'gbar_ka': .1,
            'gbar_kca': .01,
            'gbar_cas': .001,
            'gbar_cat': .005,
            'gbar_hyper': .0001,
            'tauca_capool': self.soma.tauca_capool,
            'fca_capool': self.soma.fca_capool
        }
        self.set_biophysics(**self.default_parameters)

    def set_biophysics(self,**attributes):
        """Change the membrane properties of the cell."""
        for param,value in attributes.items():
            if value is not None:
                setattr(self.soma,param,value)
            elif param in self.default_parameters.keys():
                setattr(self.soma,param,self.default_parameters[param])

    def get_biophysics(self,**attributes):
        """Get the membrane properties of the cell."""
        for param in attributes.keys():
            attributes[param] = getattr(self.soma,param)
        return attributes
```

#set default values for parameters

```
def setup_record(self):
    """Set up the vectors for recording variables."""
    self.t = h.Vector()
    self.t.record(h._ref_t)
    self.vars = ['ileak_leak','ina_na','ik_kdr',
                 'ica_cas','ica_cat','ik_ka','ik_kca',
                 'ih_hyper','v','cai']
    self.clrs = ['k','y','r','orange','brown','pink','g','c']
    self.record = {}
    for v in self.vars:
        vec = h.Vector()
        vec.record(getattr(self.soma(.5),'_ref_'+v))
        self.record[v] = vec

def plot_vars(self,cellid=0,figsize=None):
    """Plot recorded variables."""
    cellname = 'Cell B' if cellid>0 else 'Cell A'
    clr = 'r' if cellid>0 else 'b'
    t = self.t
    fig = plt.figure(figsize=figsize)
    axs = fig.subplots(3,1,sharex=True,gridspec_kw={'hspace':0.1})
    axs[0].set_title(cellname)
    axs[0].plot(t,self.record['v'],clr)
    axs[0].set_ylim(-90,60)
    axs[0].set_ylabel('Membrane Voltage (mV)')
    axs[2].plot(t,self.record['cai'],clr)
    axs[2].set_ylim(0,0.4)
    axs[2].set_ylabel('Calcium Pool (mM)')
    for i,v in enumerate(self.vars[:-2]):
        if getattr(self.soma,'gbar_'+v.split('_')[-1])>0:
            axs[1].plot(t,self.record[v],color=self.clrs[i],label=v)
    axs[1].legend(loc=1)
    axs[1].set_ylabel('Current (nA/cm$^2$)')
    axs[2].set_xlim(t[0],t[-1])
    axs[2].set_xlabel('Time (ms)')
    return fig, axs
```

#define the variables to record and plot the results for a single neuron

Build the Two-Cell HCO:

```
!pip install neuron
```

#install neuron package

```
import os

if os.path.basename(os.getcwd())!='HCO-Model-Project-Colab':
  !git clone https://github.com/chenziao/HCO-Model-Project-Colab.git
  %cd HCO-Model-Project-Colab/
else:
  print('Mod files have already been downloaded.')
%ls
```

#import mod files that describe channel activity

```
import os
print(os.system('nrnivmodl')) # compile modfiles. Return 0 for success, 1 for failure.
```

#compile mod files

```
from neuron import h
from HCO_cell_template import HCOcell
import matplotlib.pyplot as plt
import ipywidgets as widgets

h.load_file('stdrun.hoc')

h.dt = 0.025 # time step (resolution) of the simulation in ms
h.v_init= -50 # initial membrane potential in mV

# create two cells from HCO_cell_template
cellA = HCOcell()
cellB = HCOcell()
cells = [cellA,cellB] # put the two cells in a list

# create a current clamp to cellA
ccl = h.IClamp(cellA.soma(.5))

# create two synapses connecting the two cells
syn = [] # make empty list for synapse object
nc = [] # make empty list for NetCon object
for cell in cells:
    syn.append(h.inhsyn(.5,sec=cell.soma)) # create a synapse into each cell
for i,cell in enumerate(cells):
    nc.append(h.NetCon(cell.soma(.5)._ref_v,syn[1-i],0,0,10,sec=cell.soma)) # a NEURON object that connects a source to a target
```

#import the cell template created above, duplicate it, connect the two neurons with two synapses, give one neuron current injection.

```
h.tstop = 2000 # Simulation time (ms)

# Biophysical parameters
parameters = {
    # Conductance of channels (siemens/cm2)
    'gbar_leak': None,
    'gbar_na': .1,    # (.1~.5)
    'gbar_kdr': .1,    # (.1~.5)
    'gbar_ka': .1,    # (.1~.5)
    'gbar_kca': .015,    # (.01~.05)
    'gbar_cas': .003,    # (.001~.01)
    'gbar_cat': .005,    # (.005~.01)
    'gbar_hyper': .0003,    # (.0001~.0003)
    #Ca pool parameters
    'tauca_capool': None,    # decay time constant
    'fca_capool': None    # ca influx factor that goes to ca pool
} # parameters set to "None" are to use default values defined in the template

# Current injection parameters
ccl_param = {
    'delay': 200,    # start time (ms) 100
    'dur': 50,   # duration (ms) 10
    'amp': 2    # amplitude (nA) (set to 0 to disable current injection) 1
}

# Synapse parameters
syn_param = {
    'esyn': -70,    # synaptic channels reversal potential (mV)
    'gmax': 40e-3,    # synaptic channels maximum conductance (uS) (default: 40e-3)
    'tau1': 10,    # rise time (ms) (default: 10)
    'tau2': 20    # decay time (ms) (default: 20)
}


# Setup parameters (Do Not need to change the code below)
for p,v in ccl_param.items():
    setattr(ccl,p,v)
for cell,s in zip(cells,syn):
    cell.set_biophysics(**parameters)
    for p,v in syn_param.items():
        setattr(s,p,v)

# Run simulation
_ = h.run()
```

#set parameters and run simulation

```
out  = [widgets.Output() for _ in range(4)]

# Print parameters
with out[0]:
    print('Parameters:')
    for p,v in cellA.get_biophysics(**parameters).items():
        print('{} = {}'.format(p,v))
# Plot results
with out[1]:
    plt.figure(figsize=(10,4.8))
    for i,cell in enumerate(cells):
        cellname = 'Cell B' if i>0 else 'Cell A'
        clr = 'r' if i>0 else 'b'
        plt.plot(cell.t,cell.record['v'],clr,label=cellname)
    plt.xlim(cellA.t[0],cellA.t[-1])
    plt.ylim(-90,60)
    plt.legend(loc=1)
    plt.xlabel('Time (ms)')
    plt.ylabel('Membrane Voltage (mV)')
    plt.show()
for i,cell in enumerate(cells):
    with out[i+2]:
        fig,axs = cell.plot_vars(cellid=i,figsize=(6.4,8))
        plt.show()

widgets.VBox([widgets.HBox([out[0],out[1]]),widgets.HBox([out[2],out[3]])])
```

#plot results

References

Calabrese, Ronald, Angstadt, J. D., & Arbas, E. A. (1989). A Neural Oscillator Based on Reciprocal Inhibition. *Perspectives in Neural Systems and Behavior*, 33–50.

Doloc-Mihu, A., & Calabrese, R. L. (2011). A database of computational models of a half-center oscillator for analyzing how neuronal parameters influence network activity. *Journal of Biological Physics, 37*(3), 263–283. https://doi.org/10.1007/s10867-011-9215-y

Marder E, Bucher D. Central pattern generators and the control of rhythmic movements. Curr Biol. 2001 Nov 27;11(23):R986-96. doi: 10.1016/s0960-9822(01)00581-4. PMID: 11728329.

Sharp AA, Skinner FK, Marder E. Mechanisms of oscillation in dynamic clamp constructed two-cell half-center circuits. J Neurophysiol. 1996 Aug;76(2):867-83. doi: 10.1152/jn.1996.76.2.867. PMID: 8871205.