Using Josephson Junctions to Simulate Neural Network Behaviors

Yen-Hsiang Chang
Email:alexalex881108@gmail.com

Abstract

In this project, we use computer programming language "python" to write programs to obtain numerical solutions of non-ordinary differential equations. Also, we reproduce the results in (*1*). We first simulate a single neuron, and simulate connecting neurons. And we summarize some content in (*1*) to compare with our results.

1 Introduction

In order to realize the operation of neural networks systems, some projects (*2–4*) have used Large-scale digital simulation to demonstrate neural networks system. Although it is effective to use parallel computing, the simulation time is still a huge problem to include biologically realistic features since this simulation method would cause discontinuous voltage and discontinuous current. Another method is analog simulation by using very-large-scale-integrated (VLSI) circuits. Using VLSI circuits to mimic neurons and synapses could improve in realism and speed, but complexity and power consumption are still big issues. In recent years, scientists attempt to use superconducting circuits to simulate neural networks systems. This method not only make the voltage and current in the circuits into continuous, but also conquer the issues about computational time and power consumption. One example is *Josephson junction neurons - JJ Neurons* (*5*). JJ-Neurons can be used to demonstrate biological characteristics of neurons and be realized in parallel computing. Thus, we can use JJ-Neurons to research into problems about huge and complex neural network systems.

In this project, we aim to discuss some issues by using computer programming language "python"

- Solve the numerical solutions of JJ-Neurons circuits differential equations and choose appropriate parameters for every electronic components.
- Simulating the response about JJ-Neurons after current pulses.
- Simulating neurons characteristics included the action potential, firing threshold and refractory period.

2 Preliminaries

2.1 Josephson junction

Josephson junctions (*5*) consist of two superconductors separated by a thin insulating barrier. Electrons in each superconductor are described by a coherent wave function with definite phase. The phase difference is the so-called Josephson phase ϕ, which controls all of the electrical properties of the junction such as the voltage and current. Current can flow through the device without creating a voltage, which is called supercurrent. We can use Josephson junctions to mimic ion channels in neurons.

2.2 Euler Method

Euler method (*6*) is a first-order numerical procedure for solving ordinary differential equations with a given initial value. Suppose that the initial value is to be solved on the interval $[x_0, X_M]$. We can divide the interval by the **mesh-points** $x_n = x_0 + nh, n = 0, ..., N$, where $h = (X_M - x_0)/N$ and N is a positive integer. The positive real number *h* is called the **step size**. For each **n**, we seek a numerical approximation y_n to $y(x_n)$, the value of the analytical solution at the mesh point x_n. Given that $y(x_0) = y_0$ is known, suppose that we have already solve y_n up to some n. $0 \le n \le N-1$, we can define

$$y_{n+1} = y_n + hf(x_n, y_n), \qquad n = 0, ..., N-1$$

. Taking in succession $n = 0, 1, ..., N-1$, one step at a time, the approximate values y_n at the mesh points x_n can be easily solved. However, the disadvantages of Euler method are that the local error is proportional to the square step size, and the global error is proportional to the step size. Thus, Euler method is often used to be the basis to construct more complex methods.

2.3 Runge-Kutta Methods

Runge-Kutta methods (*6,7*) aim to achieve higher accuracy by sacrificing the efficiency of Euler method through re-evaluating $f(\cdot,\cdot)$ at points intermediate between $(x_n, y(x_n))$ and $(x_{n+1}, y(x_{n+1}))$. The general *R*-stage Runge-Kutta family is defined by

$$\begin{aligned}
y_{n+1} &= y_n + h\Phi(x_n, y_n; h), \\
\Phi(x, y; h) &= \sum_{r=1}^{R} c_r k_r, \\
k_1 &= f(x, y) \\
k_r &= f(x + ha_r, y + \sum_{s=1}^{r-1} b_{rs} k_s), \quad r = 2, ..., R, \\
a_r &= \sum_{s=1}^{r-1} b_{rs}, \quad r = 2, ..., R.
\end{aligned}$$

Take **Four-stage Runge-Kutta methods** for example. For $R = 4$, an analogous argument leads to a two-parameter family of four stage Runge-Kutta methods of order four. A particularly popular example from this family is:

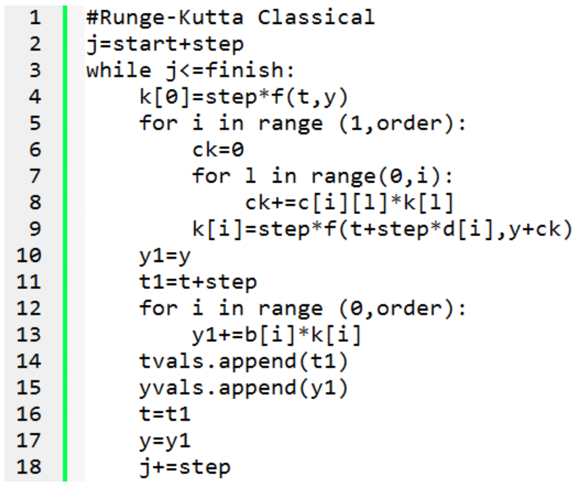
$$y_{n+1} = y_n + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4),$$

where

$$
\begin{aligned}
k_1 &= f(x_n, y_n),\\
k_2 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_1),\\
k_3 &= f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hk_2),\\
k_4 &= f(x_n + h, y_n + hk_3).
\end{aligned}
$$

Here k_2 and k_3 represent approximations to the derivative $y'(\cdot)$ at points on the solution curve, intermediate between $(x_n, y(x_n))$ and $(x_{n+1}, y(x_{n+1}))$, and $\Phi(x_n, y_n; h)$ is a weighted average of the k_i, $i = 1, ..., 4$.

Figure 1 is my main python code of Runge-Kutta methods.



```
#Runge-Kutta Classical
j=start+step
while j<=finish:
    k[0]=step*f(t,y)
    for i in range (1,order):
        ck=0
        for l in range(0,i):
            ck+=c[i][l]*k[l]
        k[i]=step*f(t+step*d[i],y+ck)
    y1=y
    t1=t+step
    for i in range (0,order):
        y1+=b[i]*k[i]
    tvals.append(t1)
    yvals.append(y1)
    t=t1
    y=y1
    j+=step
```

Figure 1: Main code of Runge-Kutta method

2.4 Single Neuron Model

We use Josephson junctions to simulate neurons, and the supercurrent flow through the device could be increased to the junction's critical current I_0 above which a voltage develops given by $V = (\Phi_0/2\pi)d\phi/dt$, where t is time, $\Phi_0 = h/e$ is the flux quantum, h is Planck's constant and

e is the electron charge. The current through a junction i depends on the phase ϕ through the relation (*8*)

$$i = \ddot{\phi} + \Gamma\dot{\phi} + \sin\phi \tag{1}$$

The dot notation refers to differentiation with resepct to normalization time τ, where $\tau^2 = t^2\Phi_0 C/2\pi I_0$, with C the capacitance of the junction. The normalized damping parameter is $\Gamma^2 = \Phi_0/2\pi I_0 R^2 C$, where R is the resistance of the junction. The critical current , conductance ($1/R$) and capacitance of a junction are assumed to scale linearly with the cross-sectional area (A) of the junction.

Equation 1 displays a useful analogy between the dynamics of a junction and that of a pendulum, allowing us to describe the action potential-like pulse generated by the circuit. Equation 1 is exactly the equation of motion for a damped and driven pendulum. For appropriate parameters, and time dependent torque i, the pendulum will whirl over just once and settle back to a tilted state. In a Josephson junction, the phase whirling over just once creates a magnetic flux pulse, called a single-flux quantum pulse (SFQ pulse) (*9*). This pulse has a similar shape and area each time it is stimulated. Pulses of this type form the action potentials in our neuron model.

The JJ Neuron circuit shown in Figure 2 connects two Josephson junctions in a superconducting loop. The structure is a simplified superconducting digital component from RSFQ logic circuitry called a "dc-to-SFQ converter" (*10*). We call the two junctions the pulse junction and the control junction, denoted by subscripts p and c. And we define some notations. Two incoming currents (normalized to I_{0p}) are called the input current $i_i n$ and the bias current i_b which provides energy to the circuit. The signal to the synapse is the voltage across the pulse junction $v_p = \dot{\phi}_p$, through the magnetic flux through the loop could be used for an inductive synapse circuit. The figure shows a model synaptic circuit connected above the pulse junction and thus driven by v_p. The circuit parameters are the indicated branch inductances L_s, L_p, and L_c which we scale by theri sum L_{total} to obtain Λ_s, Λ_p and Λ_c. Using current conservation and fluxoid quantization (*8*), we can obtain two equations of motion,

$$\ddot{\phi}_p + \Gamma\dot{\phi}_p + \sin\phi_p = i_p = -\lambda(\phi_c + \phi_p) + \Lambda_s i_{in} + (1 - \Lambda_p)i_b. \tag{2}$$

$$\eta[\ddot{\phi}_c + \Gamma\dot{\phi}_c + \sin(\phi_c)] = i_c = -\lambda(\phi_c + \phi_p) + \Lambda_s i_{in} - \Lambda_p i_b \tag{3}$$

$\lambda = \Phi_0/2\pi L_{total} I_{0p}$ is the coupling parameter and $\eta = A_c/A_p$ is the geometric parameter. Γ,

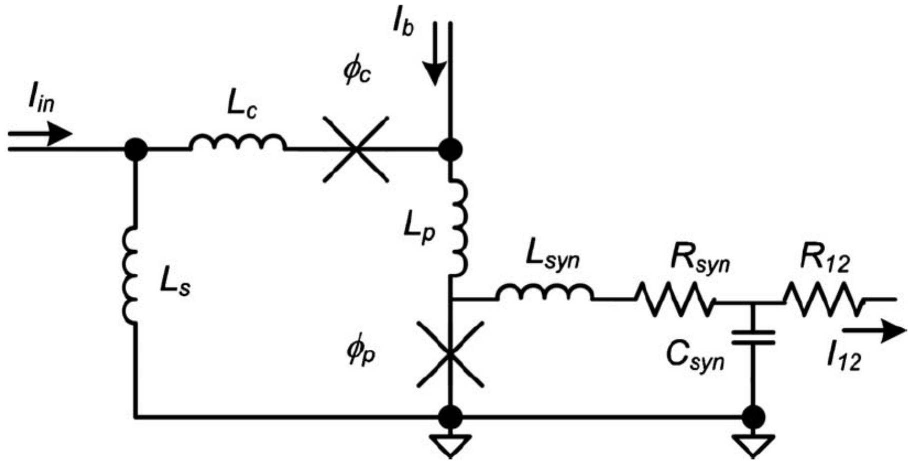


Figure 2: Main code of Runge-Kutta method

λ, Λ_s, i_{in}, Λ_p, i_b and η are all the parameters relate to the circuit and we can modify them. To make the comparison of the circuitry more clear, I set the parameters to the values suggested in (*1*), $\Lambda_c = 0, \Lambda_s = \Lambda_p = 0.5$, and $\eta = 1$ in JJ Neuron. In order to analysis the current in JJ Neuron circuit, we need to solve the circuit equations. However, the circuit equations of JJ Neurons are non-linear simultaneous differential equations, so we need to use Runge-Kutta method to obtain the numerical solutions, and I implement Runge-Kutta method by computer programming language ”python”.

3 Research Flowchar

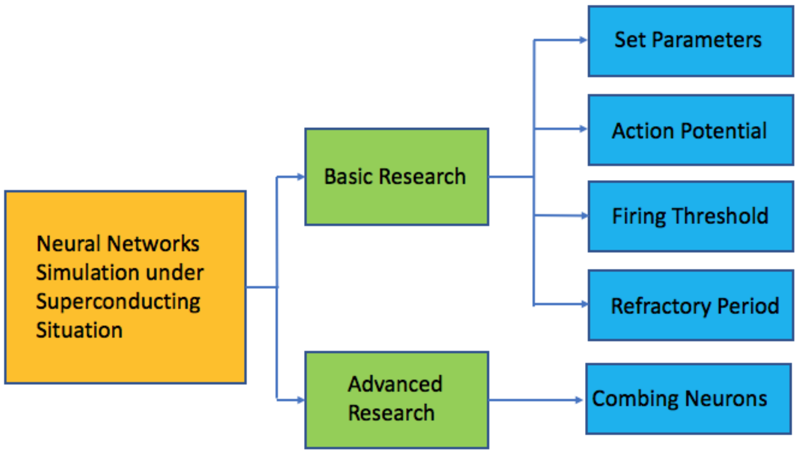


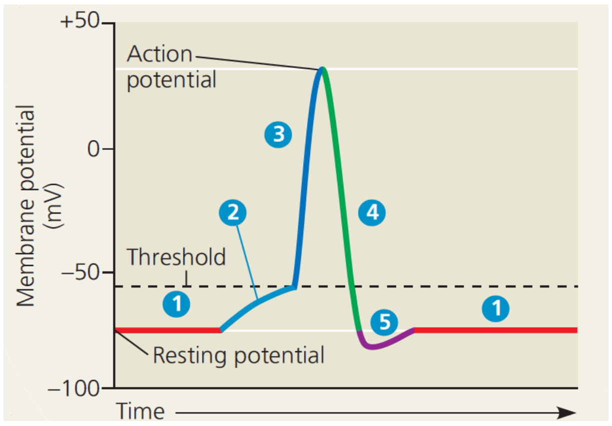
Figure 3: Research flow chart

4 Result and Discussion

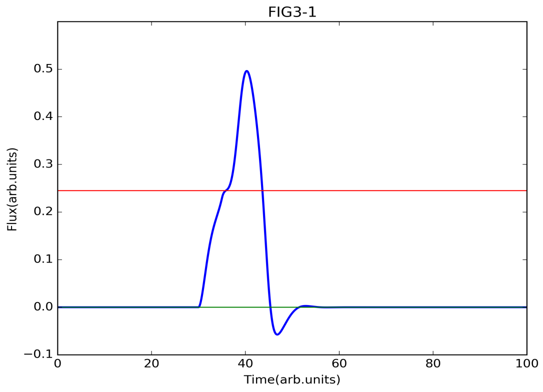
4.1 Single Neuron Model Characteristics

4.1.1 Action Potential

Action potential (AP) is a spike or pulse in voltage across the neuron membrane as the neuron fires. AP firing rates and interspike interval distributions are used to encode information in the brain (*11*). APs require at least two independent dynamic variables, generally two voltage-gated ionic currents such as Na^+ and K^+. The inward Na^+ current produces the rising phase of the pulse, while the outward K^+ current restores the membrane to its resting potential. (*12*). These currents have different time scales, resulting in the characteristics shape of the action potential as Figure 4a. Figure 4a is the typical AP, and Figure 4b is the AP python simulation implemented by myself. According to these two figures, we can see that the numerical solution of JJ Neuron circuit equations has demonstrated the characteristics of a single neuron. That is to say, we can map the different sections in the AP python simulation (Figure 4b) with the sections in actual AP (Figure 4a) with high correspondence.



(a) Action Potential (from (*13*))

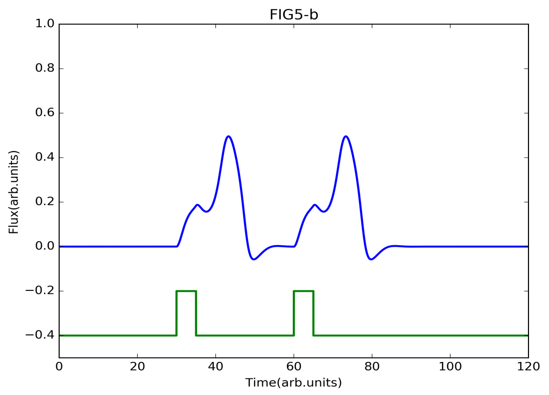


(b) Action Potential Python Simulation

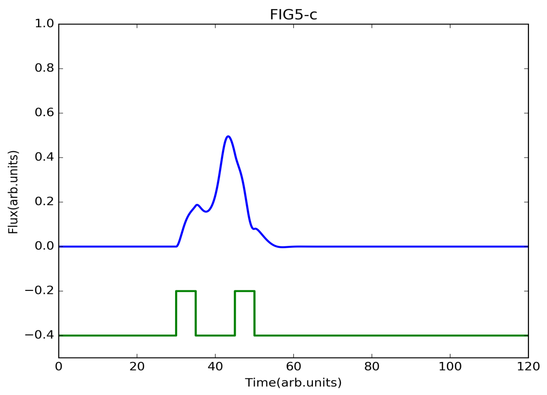
Figure 4: Caption for this figure with two images

4.1.2 Refractory Period

Figure 5b demonstrates the refractory period of the JJ Neuron. Refractory period has many different precise definitions in the literature. One universal notion about the refractory period is that it has two major impacts on the neuron (*14*). First, it ensures that propagation only proceeds in one direction at a time in the axon. Our spatially homogeneous model does not address this aspect. Second, the refractory period provides an upper limit on the frequency of a neuron's response. In Figure 5, we can see that current stimuli which are too closely spaced fail to produce a second AP, and the threshold time space is called refractory period. The most important thing is that the refractory period is not explicitly hard-coded into the model, but appears naturally as a result of differences in the time scales of the activating and restoring processes as for biological neurons.



(a)



(b)

Figure 5: Refractory Period Demonstration

4.1.3 Firing Threshold

Figure 6 demonstrates the firing threshold. It shows the response to a brief current pulse of increasing strength along with an inset showing the threshold dependence of voltage peak on stimulus strength. When stimuli exactly exceed a threshold value, neuron can exactly produce a complete action potential, and this threshold value is called ”firing threshold”. In Figure 6, we can see the corresponding membrane voltage to the firing threshold is the purple line. Stimuli below threshold evoke a small response, while above the threshold full APs occur. The inset graph shows how peak response depends on the strength of the input pulse.

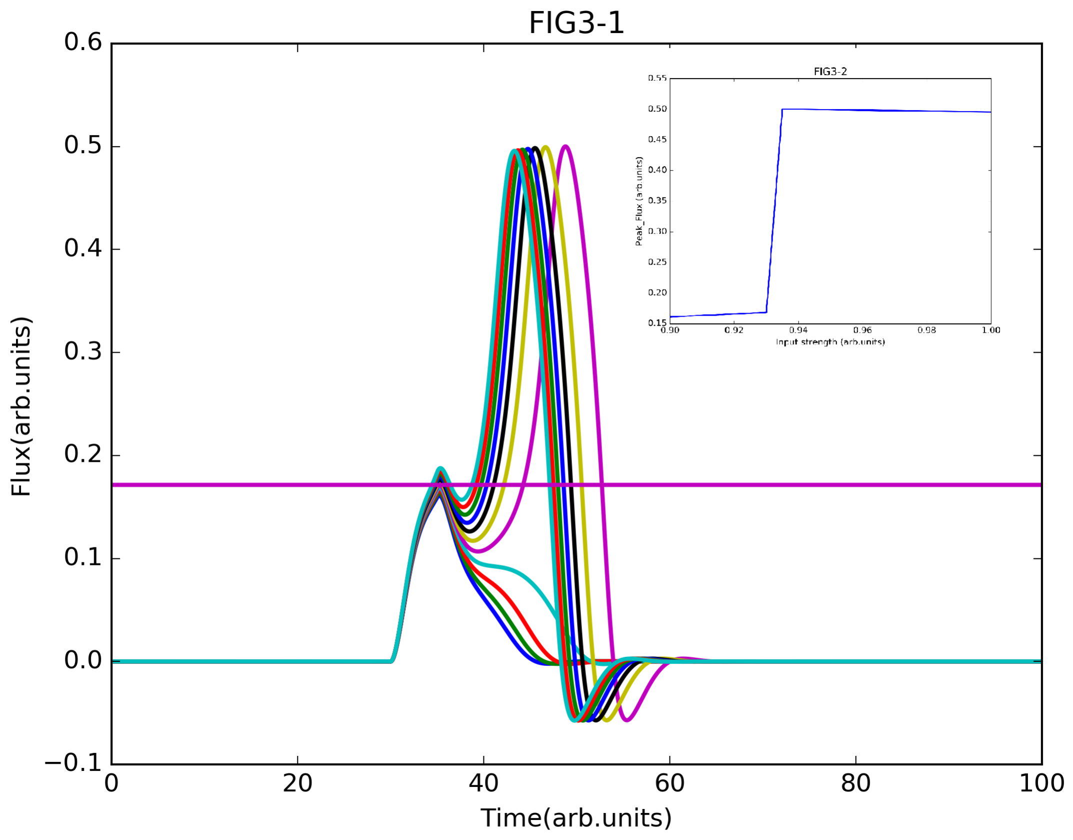


Figure 6: Firing Threshold

4.2 Connecting Neurons

Synaptic models connect individual neuron models to form a network. Synapses control communication, signal transfers, and timing (*15*). They can operate by means of direct electrical connections (electrical "gap junction" synapses) or chemical intermediates called neurotransmitters (chemical synapse). We focus on chemical synapses here. Chemical synapses can be excitatory or inhibitory, moving the postsynapse neuron closer to or farther from threshold, respectively. In chemical synapses, an AP causes the release of neurotransmitter molecules which diffuse across the synaptic cleft, bind to receptors on the postsynaptic membrane, and induce input currents in the postsynaptic neuron. The net effect on AP transmission across the synapse

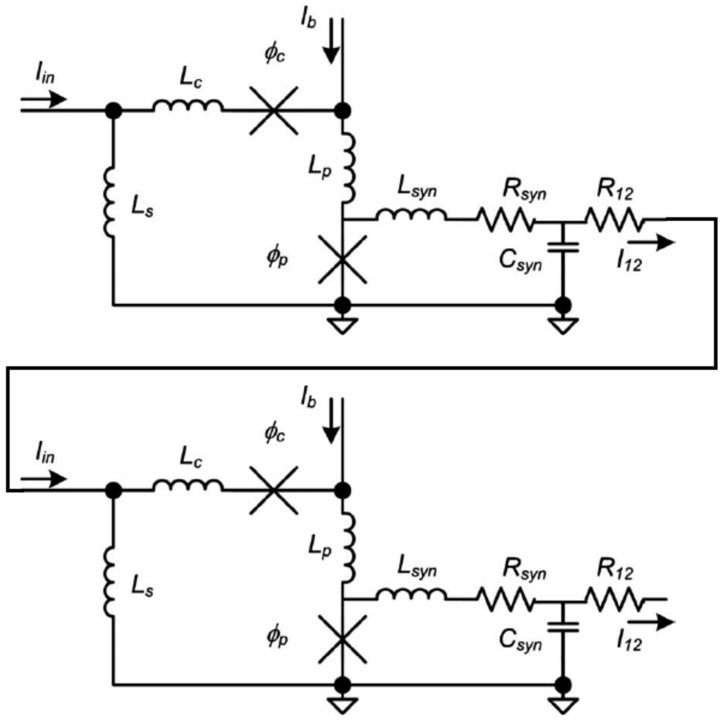


Figure 7: Connecting Two Neurons

is a delay in transmission and the spreading of the pulse. We model the synaptic process using a resonant circuit attached to the pulse junction, shown in the right hand side of Figure 2. The output is taken across the capacitor and sent through a resistor to the input of a postsynaptic neuron. If the bias current applied to the JJ Neuron is positive (negative) with respect to ground, then the synapse is excitatory (inhibitory). Besides the equation 2 and 3, we derive two more equations as

$$\frac{1}{\Omega_0^2}\ddot{v_{out}} + \frac{Q}{\Omega_0}\dot{v_{out}} + v_{out} = v_p - \frac{Q\Omega_0\Lambda_{syn}}{\lambda}i_{12} - \frac{\Lambda_{syn}}{\lambda}\dot{i_{12}} \tag{4}$$

$$\frac{\Lambda_{syn}(1-\Lambda_{syn})}{\lambda}\dot{i_{12}} + \frac{r_{12}}{\Gamma}i_{12} = v_out - \Lambda_{syn}(v_{c2} + v_{p2}) \tag{5}$$

where $\Omega_0 = \tau/\sqrt{L_{syn}C_{syn}}$ is the resonant frequency, $Q = \Omega_0 R_{syn}C_{syn}/\tau$ is the quality factor, $\Lambda_{syn} = L_{syn}/L_{total}$ is the synaptic coupling and r_{12} normalized by R_{0p} is denoted as the resistive coupling to the next neuron. v_{p2} and v_{c2} represent the voltage across the pulse and control junction of the next neuron, respectively.

In addition, the synapse circuit's back-action on the JJ Neuron circuit creates two additional terms on the right side of Equation 2: $-i_{12} - \lambda v_{out}/(\Lambda_{syn}\omega_0^2)$. If we want to get more accurate solutions, we need to add the two additional terms back to the differential equations. However, for the simple demonstrations of coupling behavior shown here, this back-action was not strong enough to warrant buffer junction.

We demonstrate synaptic behavior for inhibitory and excitatory coupling in a two-neuron system Figure 7

4.2.1 Excitatory Synaptic Coupling

In Figure 8, one JJ neuron in blue (neuron 1) drives another in green (neuron 2). The red line denoted the input current scale after shifted vertically, and the input current at $t = 0$ is zero, and the system is initially quiet. We show excitatory coupling from neuron 1 to neuron 2. Neuron 1 receives an external stimulus, and its output drives neuron 2, which responds by firing at the same rate but out of phase with neuron 1. We can see that this model simulates the depolarization of excitatory synaptic coupling very well.

4.2.2 Inhibitory Synaptic Coupling

In Figure 9, we show inhibitory coupling from neuron 1 (blue line) to neuron 2 (green line). Neuron 2 is configured to fire repetitively by increasing its bias current. When neuron 1 is not firing, this external stimulus causes neuron 2 to fire repeatedly. However, when neuron 1 is stimulated (lower red line), it inhibits the firing of neuron 2 even though neuron 2 continues to receive the external stimulus.

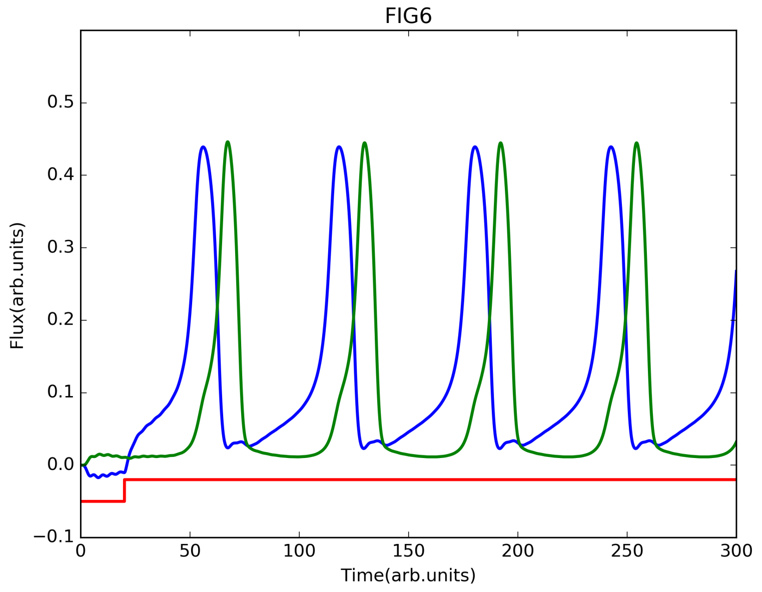


Figure 8: Excitatory Synaptic Coupling

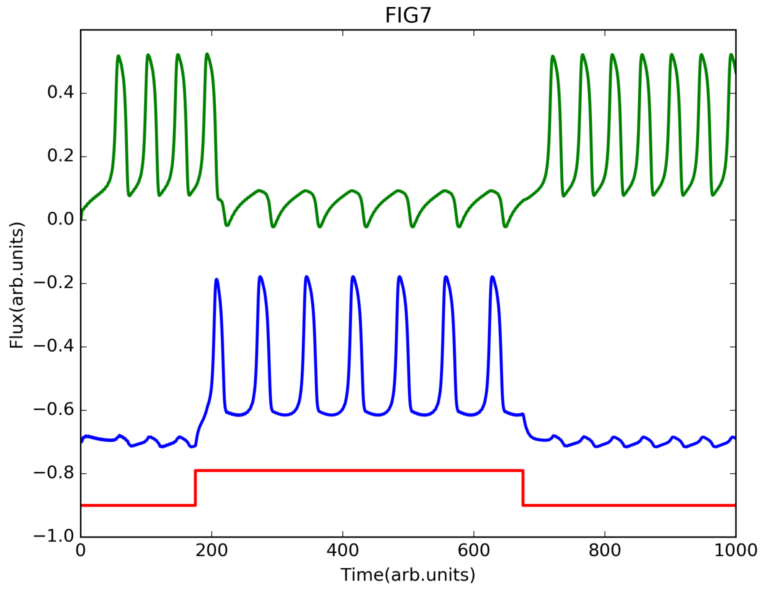


Figure 9: Inhibitory Synaptic Coupling

5 Conclusion

In this project, we use computer to simulate the behaviors of neurons, solve numerical solutions of circuit equation and observe the According to the above results, JJ Neurons have outstanding ability to simulate neural network, demonstrate characteristics of neurons very well, and change the simulation method from digital simulation to non-digital simulation. Real neurons can be divided into three different modes by its dynamic behaviors. Thus, we have to design circuits to simulate three different types of neurons, and also use JJ neurons to simulate real signal transmission behaviors in biological neuron networks. In the future, we also want to combine JJ neurons with memory to simulate signal computation in human brain.

References and Notes

1. P. Crotty, D. Schult, K. Segall, *Physical Review E* **82**, 011914 (2010).
2. H. Markram, *Nature Reviews Neuroscience* **7**, 153 (2006).
3. J. G. King, *et al.*, *Frontiers in neuroinformatics* **3** (2009).
4. E. M. Izhikevich, *IEEE transactions on neural networks* **15**, 1063 (2004).
5. B. D. Josephson, *Physics letters* **1**, 251 (1962).
6. E. Süli (2014).
7. E. Hairer, C. Lubich .
8. T. P. Orlando, K. A. Delin, *Foundation of applied superconductivity* (Addison-Wesley, 1991).
9. K. Likharev, V. Semenov, *IEEE Transactions on Applied Superconductivity* **50** (1991).
10. P. Bunyk, K. Likharev, D. Zinoviev, *International journal of high speed electronics and systems* **11**, 257 (2001).
11. F. Rieke, *Spikes: exploring the neural code* (MIT press, 1999).
12. C. Vandenberg, F. Bezanilla, *Biophysical journal* **60**, 1511 (1991).
13. J. B. Reece, *et al.*, *Campbell biology* (Pearson Boston, 2014).
14. R. A. Barker, S. Barasi, M. J. Neal, *Neuroscience at a Glance* (Wiley-Blackwell, 2003).
15. E. R. Kandel, *et al.*, *Principles of neural science*, vol. 4 (McGraw-hill New York, 2000).