An Exploration of the FitzHugh-Nagumo Model

Authors: Brian Frost-LaPlante, George Ho and David Kim
Advisor: Stanislav Mintchev
Submitted as a project report for Ma 402 at The Cooper Union

December 13, 2017

An Overview of The Model

The FitzHugh-Nagumo model (hereafter FHN) is a commonly used mathematical model for the membrane potential of an axon. It is determined in its general form by a system of two ordinary differential equations. These equations are, in the field of biology, represented in the following way:

$$\dot{V} = f(V) - W + I \tag{1}$$

$$\tau\dot{W} = V - aW + b \tag{2}$$

where $f(V)$ is a cubic polynomial and I, a, b and τ are constants. For convenience, we take $\epsilon = 1/\tau$. Physically, V represents the membrane voltage of a cell, W is a linear recovery variable and I is an external stimulus current. In the language of geometric singular perturbation theory, one says that V is a *fast variable* and W a *slow variable.* The critical manifold for this system is described by

$$M_0 = \{(V, W)|f(V) - W + I = 0\} \tag{3}$$

and for τ sufficiently large (equivalently, for ϵ sufficiently small), the dynamics of the system are characterized by a manifold that is diffeomorphic to M_0 and locally flow-invariant. We will operate under the assumption that the experimentally determined value of τ, which is approximately 12.5, is sufficiently large. We wish to consider the initial value problem where at time $t = 0$ we have $(V, W) = (v_0, w_0)$. To qualitatively determine the dynamics of this system, we look at the *nullclines*, or the curves determined by setting the time derivatives of V and W equal to 0. From Equations 1 and 2, one can see that these nullclines are of the form:

$$N_c = \{(V, W)|W = f(V) + I\} \tag{4}$$

$$N_l = \{(V, W)|aW = V + b\} \tag{5}$$

Geometrically, the curve determined by N_c is that of a cubic polynomial, shifted vertically by I. The curve determined by N_c is simply a line, with a and b determining the slope and intercept. For all values of these constants, there exists at least one point of interception between the nullclines. We limit our exploration to the cases in which there is exactly one point of interception (as FitzHugh did in his initial proposition of FHN) which we will refer to as the *equilibrium point.* We expect that, for all initial v_0 and w_0, (V, W) will eventually settle to this rest point as time goes on. As a final note, the realization of this system we will look at most frequently is one in which $f(V) = V - V^3/3$, $\tau = 12.5$, $a = 0.8$ and $b = 0.7$. These values are near those experimentally determined for the giant axons of a squid. As a result, we refer to this as the *squid realization.* Figure 1 shows the graphs of the nullclines for the squid realization in the V-W plane, for various values of I.

Mathematical Foundations

We look to obtain some formal mathematical understanding of this dynamical system so as to aid our ability to verify and interpret our numerical simulations. We pay specific attention to the system's equilibrium point, which lies at the intersection of the cubic and linear nullclines. This point, at which $\dot{V} = \dot{W} = 0 \ \forall t$, plays an important role in the theory of dynamical systems; as will be shown later in this section, analysis of the system's Jacobian matrix at this point will characterize the behavior of the system in neighborhoods of this point.
It is first, however, important to note that such a point exists and is unique for all realizations of FHN. The W-nullcline for this system is always a straight line, while the V-nullcline is always the graph of a cubic polynomial, so to show the existence of an equilibrium point, one can show that any given line and graph of a cubic polynomial in two-dimensional real space necessarily intersect. Take $p(V)$ and $q(V)$ to be polynomials with real coefficients of orders 1 and 3, respectively. The graphs of these polynomials intercept at v where $p(v) = q(v)$, should such a v exist, or identically at the roots of $q(V) - p(V)$. We know that $q(V) - p(V)$ is a polynomial with real coefficients of order 3, and as cubic polynomials are bounded neither below nor above and are continuous on $\mathbb{R}$, there exists a $v \in \mathbb{R}$ at which $q(v) - p(v) = 0$ by the intermediate value theorem. This shows that any non-horizontal, non-vertical line will intercept the graph of any cubic polynomial at least once. Any horizontal line will intercept the graph of a cubic, as a cubic is onto, and any vertical line will do so as the domain of a cubic is $\mathbb{R}$. Thereby the cubic and linear nullclines are guaranteed to have at least one intersection point for any given parameters a, b, I, τ and cubic $f(V)$. FHN specifically limits its considerations to cases in which the nullclines intersect only once, so the uniqueness of this equilibrium point is simply a hypothesis of the model.
We look now to characterize the attractiveness of this equilibrium point, but to do so, we must first look at what is known as the *linearization* of the system near this point. Let (v, w) be the coordinates of the unique equilibrium point for some realization of FHN. The system's linearization near this point is given by

$$\begin{pmatrix} \dot{V} \\ \dot{W} \end{pmatrix} = \mathbf{J} \begin{pmatrix} V - v \\ W - w \end{pmatrix} \tag{6}$$

Where $\mathbf{J}$ is the system's Jacobian evaluated at the equilibrium point, given by

$$\mathbf{J} = \begin{pmatrix} f'(v) & -1 \\ \epsilon & -a\epsilon \end{pmatrix} \tag{7}$$

where $\epsilon = 1/\tau$. Our reason for looking at this system, and more specifically, at $\mathbf{J}$, is to hopefully apply the Hartman-Grobman theorem, which states that the behavior of the linearization of a system in a sufficiently small neighborhood of a *hyperbolic* equilibrium point is qualitatively the same as that of the original system itself. Hyperbolicity of an equilibrium point is characterized by all eigenvalues of the Jacobian matrix evaluated at that point having nonzero real part. We will now analyze the hyperbolicity of the fixed points of an FHN realization.
Solving for the eigenvalues of $\mathbf{J}$ yields

$$\lambda_{1,2} = \frac{f'(v) - a\epsilon \pm \sqrt{a\epsilon - f'(v) + 4a\epsilon - 4\epsilon}}{2} \tag{8}$$

It is difficult using this expression alone to determine hyperbolicity of a general FHN realization's equilibrium point. This noted, for sufficiently small ϵ, the eigenvalues are approximately

$\frac{f'(v) \pm \sqrt{-f'(v)}}{2}$, which provides an intuition for where the hyperbolic fixed points are in terms only of the slope of the cubic nullcline. Significantly more quantitative results can be shown when given a specific realization of FHN.

One very simple example is any realization in which the W-nullcline is vertical, i.e. $a = 0$. Equation 8 is heavily simplified in this case, and values of v for which the equilibrium point is nonhyperbolic are clearly those at which the derivative of f is 0, i.e. the two local extrema of the cubic. The sign of the real parts of the eigenvalues are also determined entirely by the tonicity of f at the equilibrium point.

As a less simple example, the squid realization with $I = 0$ has its equilibrium point at $v \approx -1.2$, and its Jacobian here has eigenvalues of, approximately, -0.119 and -0.3838. Both of these values are negative and real, so this equilibrium point is hyperbolic. The approximate expression for the eigenvalues above would lead us to believe that if we chose nonzero I so that the equilibrium point were at a position where $f'(v) > 0$, the eigenvalues would be complex with positive real part. It also hints that one could find a value of I for which the real part of the eigenvalues of the Jacobian at the equilibrium point are identically zero near the extrema of the cubic nullcline. We attempt to do so here, thus determining all values of I for which this system has a hyperbolic fixed point.

Plugging in the constants for the squid realization into Equation 8 with $f'(v)$ kept arbitrary, we have

$$\lambda_{1,2} = \frac{f'(v) - 0.064 \pm \sqrt{(-0.644)f'(v) - 0.256}}{2} \tag{9}$$

The real part of one of these eigenvalues is zero in one of two cases; either the term in the square root is non-positive and $f'(v) = 0.064$, or the term in the square root is positive and the numerator of the expression is identically zero. The latter case reduces to a problem of solving a quadratic in $f'(v)$ which has only complex roots. The details are spared here, but as the derivative of this real-valued cubic polynomial will never be complex, this case will never occur for any value of I. The first case, however, can occur, as when $f'(v) = 0.064$, the expression within the square root is negative and thus the real part of the eigenvalue is zero. As $f'(v) = 1 - v^2$, we know this occurs at equilibria for which $v = \pm\sqrt{0.936}$. Note that our expectation from the approximate eigenvalue formula was that these values of v would be near the extrema of the cubic nullcline, and they in fact are; the extrema lie at ± 1. To find the values of I for which this can occur, one simply sets the expressions for the nullclines equal to one another with these values of v and arbitrary I. The expression is

$$\frac{v + 0.7}{0.8} = v - \frac{v^3}{3} + I \tag{10}$$

which yields approximate I values of 0.935 and 1.42. For all other values of I, the equilibrium point of the squid realization is hyperbolic. This is substantial as I can take any real value with only two exceptions and the Hartman-Grobman theorem can be applied. One can follow a similar procedure for any given FHN realization to determine which I values produce non-hyperbolic equilibria. Note that there are at most six values of I that can force this; two from the situation in which the term prior to the square root is zero and the term within the square root is negative, and up to four in the case of the entire expression being zero, as this reduces to a quadratic in $f'(v)$, from which the problem becomes solving a quadratic in v.

Once we have that our system's equilibrium point is hyperbolic, we are able to apply the Hartman-Grobman theorem, which, as noted above, allows us to look only at the dynamics of the linearized system to determine the qualitative behavior of the system in some neighborhood of the fixed point.

This makes the task of analyzing the stability of the system near the equilibrium point much simpler. Specifically, we need only to continue looking at the real parts of the eigenvalues of $\mathbf{J}$. The eigenvectors associated with these eigenvalues generate invariant subspaces of $\mathbb{R}^2$ which are referred to as *stable subspaces* in the case that the real parts of the associated eigenvalues are negative or *unstable subspaces* in the case that the real parts of the associated eigenvalue is positive. It is known for a linear dynamical system that, should a solution have initial conditions within the stable subspace, that solution will converge in time to the equilibrium point. Similarly, if a solution has initial conditions inside an unstable subspace, it will certainly not converge to any point, and will in fact be unbounded. These behaviors should be observed for the unlinearized system as well in some neighborhood about the equilibrium point, by the Hartman-Grobman theorem. Most interestingly, if $\mathbf{J}$ has only eigenvalues with negative real parts, any solution that gets close to, in a qualitative sense, the equilibrium point will necessarily converge to it. This was shown to be the case for the squid realization with $I = 0$ above.

There are, however, situations in which both of the eigenvalues of $\mathbf{J}$ have positive real part, in which case, any solution getting sufficiently close to the equilibrium point will eventually necessarily leave that neighborhood. In the case that the eigenvalues have real part of different sign, the behavior will differ dependent on which subspace the point "arrives in" when it becomes sufficiently close. This explanation is far from rigorous, but provides some insight in what we expect to see when we run simulations of an FHN realization.

It is important to note how regions of the cubic nullcline correspond to the eigenvalues of $\mathbf{J}$ associated with the fixed points there. For the squid realization, we found that the eigenvalues associated with the fixed points at $v = \pm\sqrt{0.936}$ had zero real part, and it is not too difficult to show that all equilibrium points satisfying $v < -\sqrt{0.936}$ or $v > \sqrt{0.936}$ have Jacobian eigenvalues with negative real part (using Equation 9). This means that all I corresponding to equilibrium points with v outside of the interval $[-\sqrt{0.936}, \sqrt{0.936}]$ lead to realizations with attractive equilibria. This interval of equilibria, as shown earlier, corresponds to I values approximately between 0.935 and 1.42. For values of I corresponding to v within the open interval $(-\sqrt{0.936}, \sqrt{0.936})$, it is the case that both eigenvalues have positive real part and the solution of the system will not be able to get arbitrarily close to the equilibrium point, as entering a sufficiently small neighborhood around it constitutes entering an unstable subspace. We will use these facts to come up with some expectations for our simulation in the next section.

Although examination of the eigenvalues of the Jacobian has helped us to determine the behavior of the system within a neighborhood of the equilibrium point, the information is not particularly useful in cases where we cannot assume that the solution ever lies sufficiently close to the equilibrium point. The question of when this will or will not happen should be addressed in a mathematically rigorous way, but simply to provide reason to believe this will often happen, we appeal to geometric singular perturbation theory; the trajectory of a solution, for τ sufficiently large, ought to resemble closely the critical manifold M_0 for some periods of time. As the equilibrium point lies on this manifold necessarily (this manifold is the cubic nullcline), it is reasonable to expect the solution to get close to this point.

Expectations of the Simulation

Here we list several qualitative tests that we may apply to our simulator, to provide quick and simple sanity checks. These expectations are validated using our simulator, and the results are shown in Figure 2.

1. With an attractive fixed point and ϵ sufficiently small, the solution trajectory should converge to the fixed point, regardless of the initial position. In the case of the squid realization, this corresponds to fixed points with V-coordinate outside the interval $[-\sqrt{0.936}, \sqrt{0.936}]$, or in a very loose sense, "most" fixed points on decreasing sections of the cubic nullcline.

2. With a repulsive fixed point and ϵ sufficiently small, the solution trajectory should not converge to the fixed point, regardless of the initial condition. In the case of the squid realization, this corresponds to fixed points with V-coordinate inside the interval $(-\sqrt{0.936}, \sqrt{0.936})$, or in a very loose sense, "most" fixed points on the middle, increasing branch of the cubic nullcline.

3. For items 1 and 2, take ϵ larger than before. As ϵ grows, we expect the solution trajectory to adhere less and less to the cubic nullcline.

4. For a non-hyperbolic fixed point (of which there are at most 6), the Hartman-Grobman theorem is inconclusive. The solution trajectory will therefore not necessarily get arbitrarily close to the fixed point.

Networks of FitzHugh-Nagumo Neurons

The Single Neuron 'Network'

We now look to focus our exploration to a specific problem; that of communication between FHN neurons receiving and transmitting electrical pulses. A simple and instructive preliminary is that of a single FHN neuron whose equilibrium point lies on the left branch of the cubic nullcline. From this point onwards we no longer consider the excitation current I as a constant parameter but rather as a function of time, which can be thought of as an input to the system that is this single neuron. Unlike previous explorations where I was kept constant but initial conditions were varied, for the most part we consider the initial state to be equilibrium with differences in behavior entirely caused by the input function $I(t)$. For the purposes of this exploration, we limit our scope to only "step-like" $I(t)$; that is, $I(t)$ that are piecewise constant. We consider the voltage V to be the system's output, and look to study the relationship between these input and output variables in time.
The ultimate goal of any FHN neural network (regardless of its length) is to achieve perfect reconstruction of the input signal $I(t)$, which, assuming step-like $I(t)$, is to determine the values of all breakpoints and amplitudes of the piecewise constant function.
For the sake of simplicity, as an extremely general discussion would be incredibly complex, we look at a very specific set of input currents; specifically we look at those that can be represented as a linear combination of unit step functions shifted in time. We look first at the response of a single neuron to a unit step, a boxcar function, and then a squarewave to illustrate how one can determine certain characteristics of the input current given only the output voltage.

Unit Step Impulse

Consider an FHN neuron left at rest ($I = 0$) for a long time so that it has reached its equilibrium point, which is then suddenly excited by a current of magnitude I_0, beginning at some time t_0. Mathematically this is representable by $I(t) = I_0 u(t - t_0)$. In terms of the system's nullclines, this manifests in the cubic shifting vertically, a corresponding relocation of the fixed point, and the system being forced out of equilibrium.
We claim that so long as a) the system is not excited so much that its new equilibrium point is unstable (which is easily quantifiable given the development above), and b) the W nullcline is not vertical, then the system will eventually settle to the new equilibrium point, and perfect reconstruction of the input current signal is possible.
Once the impulsive excitation begins, the system can be thought of as an example of the situation already considered, in which I is constant and the initial conditions are not equilibrium. When one observes this new equilibrium voltage (assuming that this equilibrium still lies on the left branch of the cubic nullcline), the value of I_0 can be determined as the unique value that causes the nullclines to meet at that V-coordinate (assuming that the W nullcline is not vertical). It will also be clear at what time V is forced out of its initial equilibrium state, and thus t_0 can be determined as well. That is to say, if one knows to expect a unit step-like input, one can uniquely determine the input given only the output voltage after the system has been allowed to settle to its new equilibrium.

Boxcar Impulse

This perfect reconstruction, however, is not possible for more complicated input signals. Suppose, for example, that a single FHN neuron at equilibrium is excited by an I_0-amplitude current at time t_0, but that the excitation only has duration T. Mathematically, this can be represented by $I(t) = I_0[u(t-t_0)-u(t-t_0-T)]$, a boxcar function. It is useful to consider the solution's trajectory in phase space as the input current changes. Initially, the system is in equilibrium, and is jolted out of equilibrium instantaneously at time t_0. Time $t_0 + T$ could happen at one of many points in the solution's trajectory. It could occur:

a) once the solution has reached equilibrium.

b) while the solution is slowly following the cubic nullcline on either side, or

c) in a fast horizontal motion states in which the solution is not near the cubic nullcline,

Each of these possibilities will allow for a different amount of information to be concluded about the input signal given only the output voltage.

a) In the case that time $t_0 + T$ occurs when the system has reached its equilibrium, the value of I_0 can be determined from the system's second equilibrium position (as was done in the case of the unit step). Both times at which V is forced from its equilibrium position will be marked by sharp changes from equilibrium, so t_0 and T will be determinable as well. This is to say, if one expects a boxcar input current of sufficiently long duration, one can determine entirely the input current signal from the output voltage.

b) In the case that time $t_0 + T$ occurs when the solution is near the cubic nullcline, a sharp change in the voltage will be seen and thus t_0 and T can be determined. However, the value of I_0 will not be obvious, as the equilibrium point has not yet been reached. This means that if one expects a boxcar function input, one may be able to retrieve all time data but not the amplitude of the input if T is within a particular range. This range depends on the initial nullclines and I_0.

c) In the case that time $t_0 + T$ occurs during one of the intermediate portions of the system's trajectory, the voltage will not make a sharp change and thus the value of T will not be determinable. I_0 will not be either, but t_0 still will be, meaning that for a boxcar input it is always possible to determine the starting time t_0 from the output alone, but for some T, it is not possible to determine anything else specifically. For instance, for sufficiently small T this will be the case. Note that one can gain some information about T; it is necessarily less than the time it takes the system to settle back to its initial equilibrium.

Square Wave and Other Step-Like Impulses

It is not challenging to then apply the concept of the boxcar response to a square wave response, with a given duty cycle/period and amplitude. For a long enough period and certain duty cycles, one would expect the system to always reach its equilibrium in each cycle and all information about the input could be determined from the output. This can conceptually be generalized to other linear combinations of shifted unit steps. The major important concept is that the output will always reflect when the system has been excited from equilibrium, and can reflect amplitudes and durations of input step-like signals if T is sufficiently large.

The responses of a single FHN neuron following the squid realization to three different stimulus currents were simulated, and the resulting phase-space trajectories and voltage curves were plotted in Figure 3.

The Dual Neuron Unidirectional Network

With this qualitative understanding of a single neuron's voltage behavior in response to a time-dependent input current, it is natural to consider this neuron as a communicator. Suppose instead that there are two FHN neurons, one of which is a transmitter and the other of which is a receiver. The transmitter is independent, and works exactly as described above; it responds to some time-dependent current and is entirely independent of the second neuron. The receiver cannot directly see the input current, but instead is excited by $I_r(t) = \gamma(V_t(t) - V_r(t))$, where γ is some constant of proportionality. That is to say that the receiver's excitation current is proportional to the difference in voltage between the two neurons. Mathematically this network is described by a system of four differential equations as follows:

$$\begin{aligned}
\frac{dV_1}{dt} &= f(V_1) - W_1 + I \\
\frac{dW_1}{dt} &= \epsilon(V_1 - aW_1 + b) \\
\frac{dV_2}{dt} &= f(V_2) - W_2 + \gamma(V_1 - V_2) \\
\frac{dW_2}{dt} &= \epsilon(V_2 - aW_2 + b)
\end{aligned}$$

where V_1 and W_1 are the voltage and recovery variables for the first (transmitting) neuron, V_2 and W_2 are the voltage and recovery variables for the second (receiving) neuron, and I is now a time-dependent impulse current.

We are interested in seeing what information we can gleam about the initial input current given only the receiver's voltage. Once again, we assume only linear combinations of shifted unit steps as inputs, and we assume that the nullclines are such that the equilibria are on the left branches of the cubic. We also, for simplicity, assume that both neurons havet the same nullclines when unexcited so that independently they each behave the same way.

Because the communication is unidirectional (i.e. the transmitter neuron is independent with respect to the receiver), we can consider the transmitter to behave exactly as described in the above section. As the receiver neuron is only dependent on the voltage of the transmitter, any information about the input signal that was indeterminable at the output of the transmitter will necessarily also be indeterminable at the output of this receiver. What information, however, can be determined from the output of the receiver?

To answer this question, we must first answer a more fundamental question; after an excitation has occurred and the transmitter neuron has reached its equilibrium, will the receiver neuron reach its equilibrium or may it exhibit unstable behavior? We phrase this question in the following way: is this system reliable? We take reliability to be the condition that, given an input current $I(t)$ that is eventually constant, the receiving neuron eventually equilibrates to a fixed point, regardless of the exact input current.

Reliability will depend on the input signal as well as the initial nullclines, but if a system is not reliable, it is not interesting to consider a communication scheme, as little information could be gleamed from the output of the second neuron.

To show that this system can be reliable, assume that the transmitter neuron has reached its equilibrium, (v_0, w_0) at time $t = t_0$. Then, for $t > t_0$, behavior of the receiver neuron is described by a pair of differential equations as follows:

$$\begin{aligned}\frac{dV_2}{dt} &= V_2 - V_2^3/3 - W_2 + \gamma(v_0 - V_2) = (1-\gamma)V_2 - V_2^3/3 - W_2 + \gamma v_0 \\ \frac{dW_2}{dt} &= \epsilon(V_2 - aW_2 + b)\end{aligned}$$

In other words, for $t > t_0$, (V_2, W_2) is described by the single neuron model above with initial condition $(v, w) = (V_2(t_0), W_2(t_0))$, where $V_2(t_0), W_2(t_0)$ are state vectors of V_2 and W_2 at $t = t_0$ obtained from the original dual neuron unidirectional network.
Consider γ values that place the equilibrium point of the receiver neuron on the left branch of the V_2-nullcline. As we've stated earlier in this report, for all initial conditions of a neuron with equilibrium point on the left branch of the cubic nullcline, the receiver neuron will reach its equilibrium. It is important to note that, while the $f(V)$ is no longer the same as in our previous discussion (i.e. the coefficient of the linear V term is now $1-\gamma$), the curvature of $f(V)$ still satisfies the prescriptions of the FHN model. Therefore, for appropriate γ values, the system is reliable. We are then concerned with the conditions on γ such that this is the case.
The V_2-nullcline is given by $f_1(V_2) = (1-\gamma)V_2 - V_2^3/3 + \gamma v_0$, and the left branch of the nullcline is given by $V_2 < -\sqrt{1-\gamma}$ (based on $f_1'(V_2) = (1-\gamma) - V_2^2$). $\gamma \leq 1$ is required because $f_1'(V_2) < 0$ $\forall$ V_2 if $\gamma > 1$. Since the fixed point is by definition the point at which $\dot{V}_2 = \dot{W}_2 = 0$, it satisfies:

$$\begin{aligned}&(1-\gamma)V_2 - V_2^3/3 - W_2 + \gamma v_0 = 0 \\ &V_2 - aW_2 + b = 0\end{aligned}$$

By expressing W_2 in terms of V_2 and substituting, one obtains the following equation:

$$g(V_2) = (1 - \gamma - \frac{1}{a})V_2 - \frac{V_2^3}{3} + (\gamma v_0 - \frac{b}{a}) = 0 \tag{11}$$

Since $g'(V_2) < (1-\gamma-1/a) - (1-\gamma) = -1/a < 0$ for $V_2 < -\sqrt{1-\gamma}$, if the fixed point is on the left branch of the cubic nullcline, then $g(-\sqrt{1-\gamma}) < 0$. If $1-\gamma-1/a > 0$, the local maximum of $g(V_2)$ at $V_2 = \sqrt{1-\gamma-1/a}$ must be negative to ensure uniqueness of the fixed point on left branch. Putting it all together, the receiver neuron has a unique fixed point on the left branch of the V_2-nullcline, i.e. $V_2 < -\sqrt{1-\gamma}$, if $g(V_2)$ meets the following conditions:

i) if $1-\gamma-1/a > 0$

$$\begin{aligned}g(-\sqrt{1-\gamma}) &< 0 \\ g(\sqrt{1-\gamma-1/a}) &< 0\end{aligned}$$

ii) if $1-\gamma-1/a \leq 0$

$$g(-\sqrt{1-\gamma}) < 0$$

For the squid realization ($a = 0.8$ and $b = 0.7$) with $\gamma = 1$, $g(V_2) = -V_2/0.8 - V_2^3/3 + (v_0 - 7/8) = 0$. Since $1-\gamma-1/a = 1-1-1/0.8 = -1/0.8 \leq 0$ and $g(-\sqrt{1-\gamma}) = g(0) = v_0 - 7/8 < 0$, the corresponding dual neuron unidirectional network is reliable. ($v_0 - 7/8 < 0$ since $v_0 < 0$ as the fixed

point of the transmitter neuron is assumed to be on the left branch of V_1-nullcline)
So with the guarantee that our system is reliable for step-like inputs, we can reasonably discuss the information retrievable from the receiver neuron's voltage. Note that at time t_0 when the transmitter cell is excited, the excitation current for the receiver immediately becomes nonzero; in other words, the transmitter and receiver are initially excited at the same time. Thereby the initial excitation time is still always retrievable in a two-neuron network.
The receiver neuron cannot settle back to its original equilibrium state unless the transmitter has done so first, thus the second cell must take more time to equilibrate to a point at which I_0 and T are retrievable from the output. This is to say that although voltage from a single neuron may be able to provide information about an input boxcar signal's amplitude and duration, this duration may be too short for a receiver cell to have also equilibrated in the same amount of time. This can be phrased in the following way; adding more neurons to a unidirectional communication network lowers the temporal resolution on the communication scheme. Unsurprisingly this means that square waves must have lower frequencies to be recoverable by a two-neuron network than by a single neuron.
The responses of a two-FHN-neuron unidirectional network (with each neuron following the squid realization) to three different stimulus currents were simulated, and the resulting phase-space trajectories and voltage curves were plotted in Figure 4.
Although we do not simulate this situation in which there are more than two FHN neurons communicating unidirectionally, one can see that for any number of neurons an excitation in one leads to in excitation in the others, so t_0 will always be recoverable. Each additional neuron added to the system decreases the temporal resolution in the manner described above, so a system with an extremely large number of neurons unidirectionally communicating would potentially only be useful for largely spaced electrical impulses.

An Alternative Communication Method

Until now, we have worked under the assumption that while at rest and while excited, the FHN neurons in question had equilibria on the left branch of the cubic nullcline. A very similar and symmetrical case can be made for points on the right branch of the cubic nullcline, however, we have avoided unstable equilibrium points on the center branch entirely. One could, however, consider a network where the rest equilibrium point is on the center branch of the cubic nullcline. Should the W-nullcline be non-vertical, an excitation could position the excited equilibrium on one of the left or right branches. In phase space, the solution's trajectory would be periodic motion around the rest point when unexcited, but then it would tend to rest at the excited equilibrium. One could then determine whether or not the system had been excited by whether the voltage was constant or oscillatory.
This scheme still requires a settling time and thus has imperfect temporal resolution as the previously mentioned system did. It is also sensitive to magnitude of input in that if an input current is too small, the equilibrium point will remain in the center branch and periodic motion in phase space will still be observed. This, however, is a problem observable in the first scheme proposed as will, as it is always the case for non-vertical W-nullclines that a specific magnitude of excitation currents force the equilibrium point into an unstable regime. This scheme was not simulated, but is simply mentioned for the sake of interest and completeness.

A Brief Explanation of the Program

The program used to simulate and visualize FHNs was written in Python in a Jupyter notebook environment, making use of the NumPy and matplotlib libraries.
The program is capable of modelling a single FitHugh Nagumo neuron, as well as unidirectionally connected, dual-neuron networks. It is fairly simple to adapt the code to model unidirectionally-connected chains of n neurons.
To model a single neuron, the program takes values of the initial point (V_0, W_0), and FHN parameters a, b, I, and τ as inputs, and approximates the solutions of the coupled system of the corresponding realization of the FHN model, plotting the solution's trajectory in phase space. It is possible to pass in either scalar values of I, or an array of pre-computed values, in order to achieve a constant impulse current, or a time-varying current.
To model a unidirectionally connected two-neuron network, the program takes values of the initial point (V_0, W_0), and FHN parameters a, b, I, and τ as inputs, and approximates the solutions of the coupled system of the corresponding realization of the FHN model, plotting the solution's trajectory in phase space. It is possible to pass in either scalar values of I, or an array of pre-computed values, in order to achieve a constant impulse current, or a time-varying current.
A straightforward Runge-Kutta 4 solver was implemented to approximate the solutions to the coupled system of differential equations, and simple static graphs are produced. The nullclines are plotted with dashed lines, and the solution trajectory is plotted with a solid line and changing colors: as such, the time dependence of the solution trajectory can be visualized.
The source code is included in the appendix of this document.

Figures

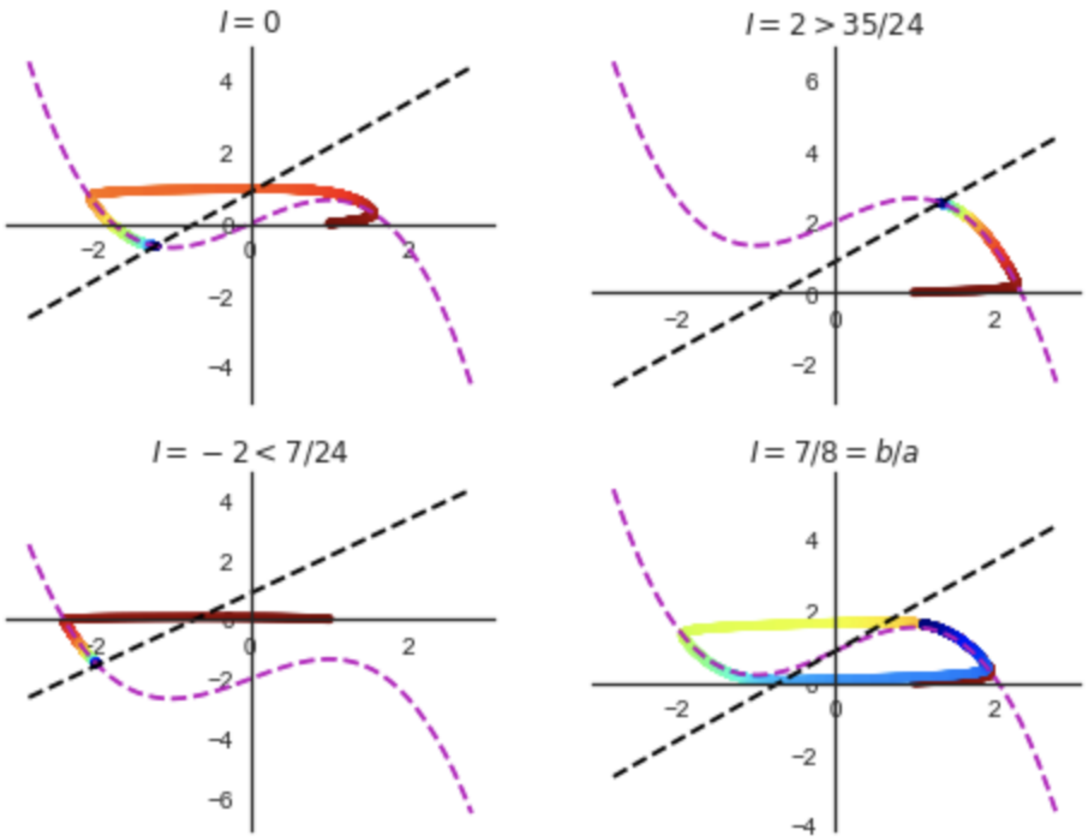

Figure 1: Nullclines for the squid realization, with $(V_0, W_0) = (1, 0.01)$ and varying values of I

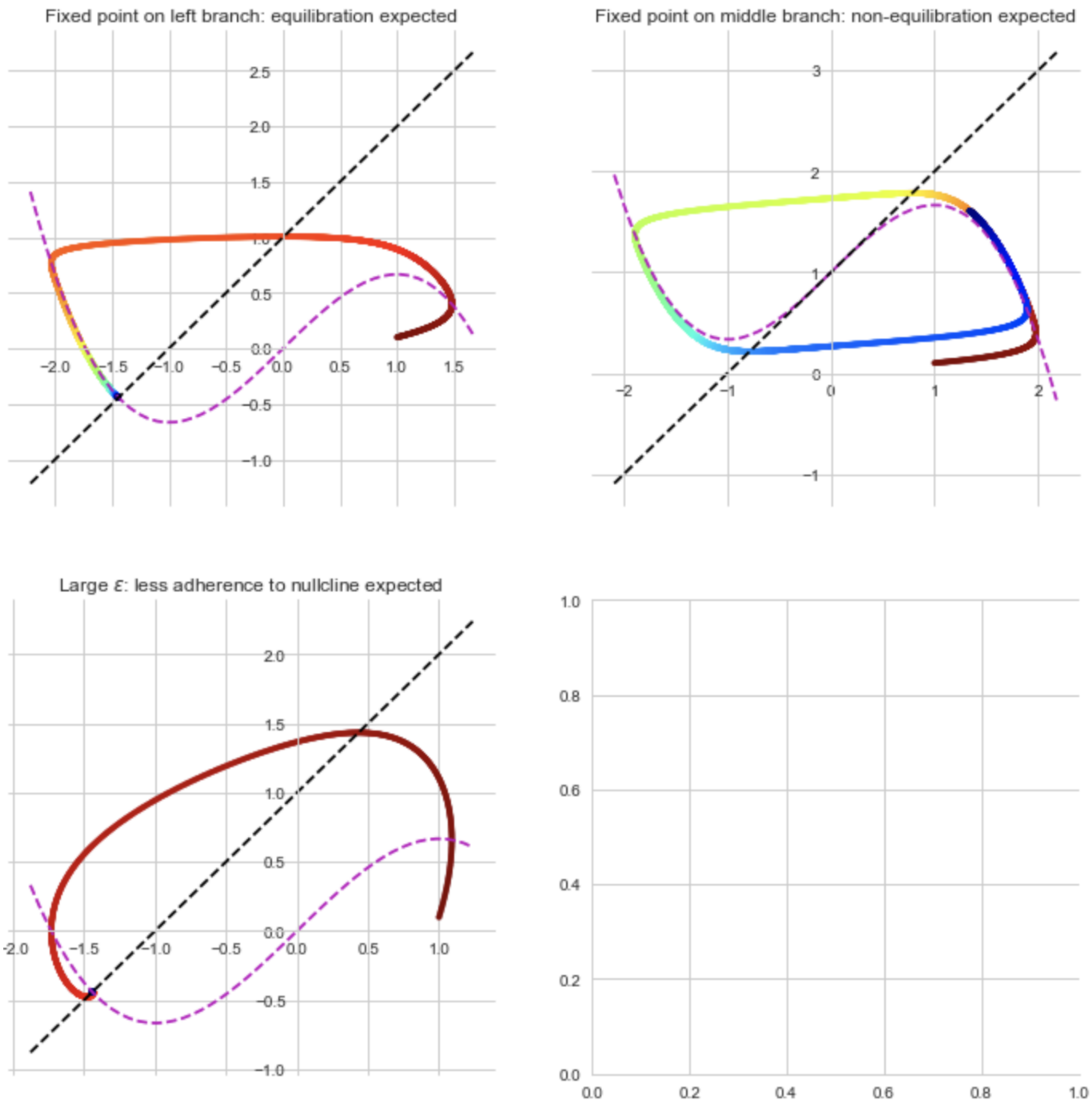

Figure 2: Numerical simulation of expectations of simulations

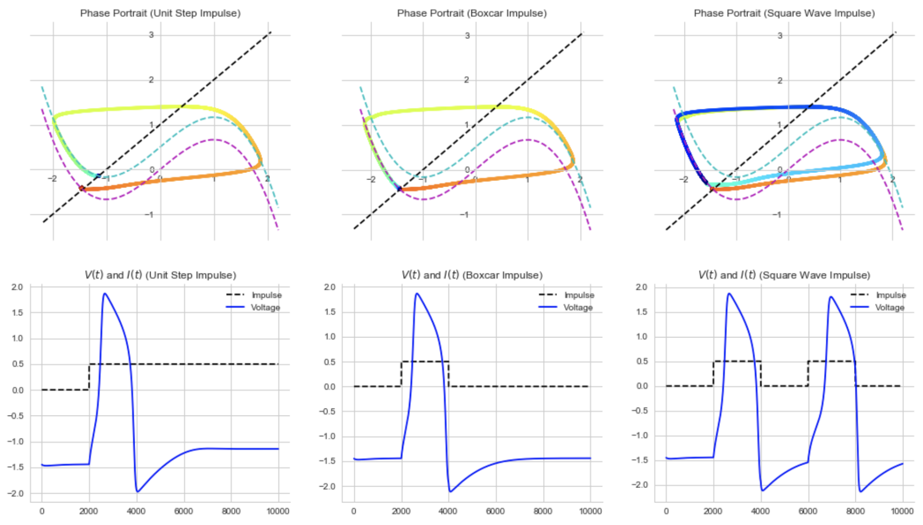

Figure 3: Phase portrait and output voltage for a single neuron, for various input currents

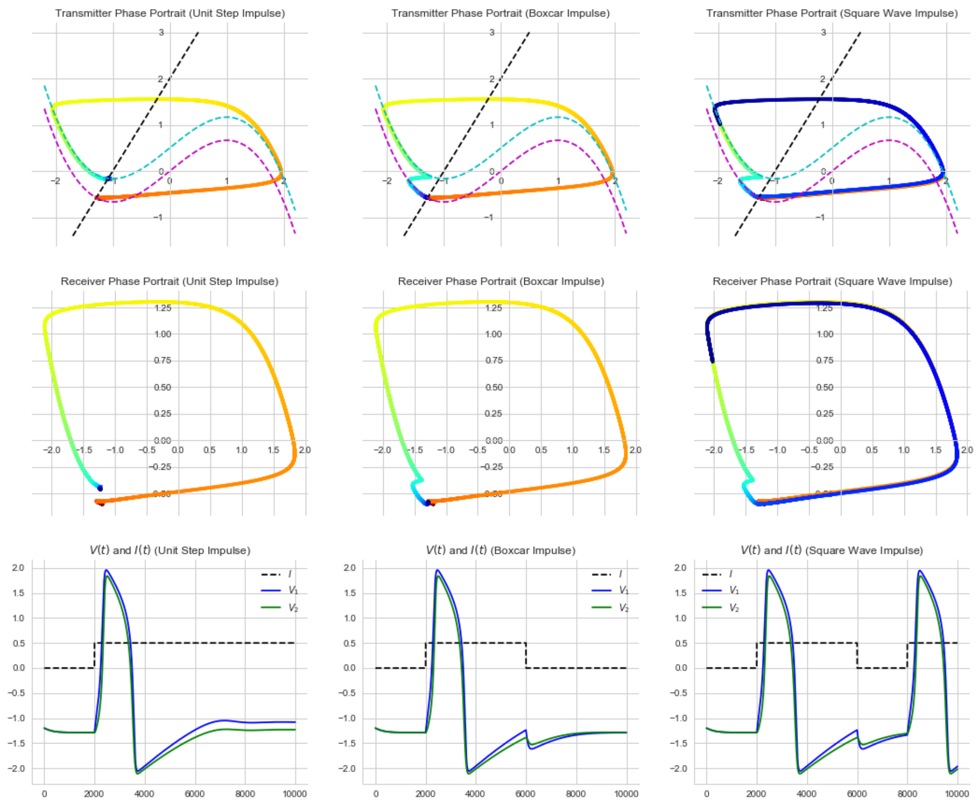

Figure 4: Phase portraits and output voltages for both neurons in a unidirectional neuron network, for various input currents

Appendix - Code

```
import numpy as np
import numbers
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns
sns.set(style='whitegrid')


# Derivatives (RHS of differential equations)

def g1(V, W, I):
    return V - (1/3)*V**3 - W + I


def g2(V, W, tau, a, b):
    return (1/tau)*(V - a*W + b)


def rk4_xmtr(V0, W0, h, N, tau, a_arr, b_arr, I_arr):
    '''
    Runge-Kutta solver for transmitting neuron

    Parameters
    ----------
    (V0, W0) = initial point in phase space
    h = step size
    N = number of steps to simulate for
    tau = time scale constant
    a, b, I = parameters of FHN

    I_arr = impulse current in transmitting neuron
    '''

    V_arr = np.zeros(N, dtype=np.float128)
    W_arr = np.zeros(N, dtype=np.float128)

    V_arr[0] = V0
    W_arr[0] = W0

    for i in range(N-1):
        k0 = h*g1(V_arr[i], W_arr[i], I_arr[i])
        l0 = h*g2(V_arr[i], W_arr[i], tau, a_arr[i], b_arr[i])
        k1 = h*g1(V_arr[i] + 0.5*k0, W_arr[i] + 0.5*l0, I_arr[i])
        l1 = h*g2(V_arr[i] + 0.5*k0, W_arr[i] + 0.5*l0, tau, a_arr[i], b_arr[i])
        k2 = h*g1(V_arr[i] + 0.5*k1, W_arr[i] + 0.5*l1, I_arr[i])
        l2 = h*g2(V_arr[i] + 0.5*k1, W_arr[i] + 0.5*l1, tau, a_arr[i], b_arr[i])
        k3 = h*g1(V_arr[i] + k2, W_arr[i] + l2, I_arr[i])
        l3 = h*g2(V_arr[i] + k2, W_arr[i] + l2, tau, a_arr[i], b_arr[i])

        V_arr[i+1] = V_arr[i] + (1/6)*(k0 + 2*k1 + 2*k2 + k3)
        W_arr[i+1] = W_arr[i] + (1/6)*(l0 + 2*l1 + 2*l2 + l3)

    return V_arr, W_arr


def rk4_rcvr(V0, W0, h, N, tau, a_arr, b_arr, gamma, V1_arr):
    '''
    Runge-Kutta solver for the receiving neuron

    Parameters
    ----------
    (V0, W0) = initial point in phase space
    h = step size
    N = number of steps to simulate for
    tau = time scale constant
    a, b, I = parameters of FHN

    gamma = proportionality constant of
    V1_arr = voltage of the transmitting neuron
    '''

    V_arr = np.zeros(N, dtype=np.float128)
    W_arr = np.zeros(N, dtype=np.float128)

    V_arr[0] = V0
    W_arr[0] = W0

    for i in range(N-1):
        k0 = h*g1(V_arr[i], W_arr[i], gamma*(V1_arr[i]-V_arr[i]))
        l0 = h*g2(V_arr[i], W_arr[i], tau, a_arr[i], b_arr[i])
        k1 = h*g1(V_arr[i] + 0.5*k0, W_arr[i] + 0.5*l0, gamma*(V1_arr[i]-V_arr[i]))
        l1 = h*g2(V_arr[i] + 0.5*k0, W_arr[i] + 0.5*l0, tau, a_arr[i], b_arr[i])
        k2 = h*g1(V_arr[i] + 0.5*k1, W_arr[i] + 0.5*l1, gamma*(V1_arr[i]-V_arr[i]))
        l2 = h*g2(V_arr[i] + 0.5*k1, W_arr[i] + 0.5*l1, tau, a_arr[i], b_arr[i])
        k3 = h*g1(V_arr[i] + k2, W_arr[i] + l2, gamma*(V1_arr[i]-V_arr[i]))
        l3 = h*g2(V_arr[i] + k2, W_arr[i] + l2, tau, a_arr[i], b_arr[i])

        V_arr[i+1] = V_arr[i] + (1/6)*(k0 + 2*k1 + 2*k2 + k3)
        W_arr[i+1] = W_arr[i] + (1/6)*(l0 + 2*l1 + 2*l2 + l3)
    return V_arr, W_arr


def single_fhn(V0, W0, h, N, tau=None, a=None, b=None, I=None, ax=None):
    '''
    Plots a phase portrait of the FitzHugh-Nagumo neuron

    Parameters
    ----------
    (V0, W0) = initial point in phase space
    h = step size
    N = number of steps to simulate for
    tau = time scale constant; defaults to 12.5 ("sufficiently large")
    a, b, I = parameters of FHN; default to 1, 1, 0
    '''

    if tau is None:
        tau = 12.5
    if a is None:
        a = 1
    if isinstance(a, numbers.Number):
        a_arr = np.repeat(a, N)
    else:
        a_arr = a
    if b is None:
        b = 1
    if isinstance(b, numbers.Number):
        b_arr = np.repeat(b, N)
    else:
        b_arr = b
    if I is None:
        I = 0
    if isinstance(I, numbers.Number):
        I_arr = np.repeat(I, N)
    else:
        I_arr = I
    if ax is None:
        ax = plt.gca()

    V_arr, W_arr = rk4_xmtr(V0, W0, h, N, tau, a_arr, b_arr, I_arr)

    # Plot phase trajectory
    ax.scatter(V_arr, W_arr,
               c=np.linspace(10, 0, N),
               cmap=cm.jet,
               marker='.')

    x_bottom, x_top = ax.get_xlim()
    y_bottom, y_top = ax.get_ylim()

    # Plot nullclines, if a, b, I are constants
    x = np.linspace(x_bottom, x_top)

    try:
        V_nullcline = x - (1/3)*x**3 + I
        ax.plot(x, V_nullcline, color='m', linestyle='--')
    except:
        pass

    try:
        W_nullcline = (1/a) * (x + b)
        ax.plot(x, W_nullcline, color='k', linestyle='--')
    except ZeroDivisionError:
        ax.axvline(-b, y_bottom, y_top, color='k', linestyle='--')

    ax.spines['left'].set_position('zero')
    ax.spines['bottom'].set_position('zero')
    sns.despine()

    return V_arr, W_arr, ax


def dual_fhn(V0_1, W0_1, V0_2, W0_2, h, N, tau=None,
             a=None, b=None, I=None, gamma=None, axarr=None):
    '''
    Plots a phase portrait of the FitzHugh-Nagumo neuron

    Parameters
    ----------
    (V0, W0) = initial point in phase space
    h = step size
    N = number of steps to simulate for
    tau = time scale constant; defaults to 12.5 ("sufficiently large")
    a, b, I = parameters of FHN; default to 1, 1, 0
    '''

    if tau is None:
        tau = 12.5
    if a is None:
        a = 0.8
    if isinstance(a, numbers.Number):
```

```
    a_arr = np.repeat(a, N)
else:
    a_arr = a
if b is None:
    b = 0.7
if isinstance(b, numbers.Number):
    b_arr = np.repeat(b, N)
else:
    b_arr = b
if gamma is None:
    gamma = 1
if I is None:
    I = 0
if isinstance(I, numbers.Number):
    I_arr = np.repeat(I, N)
else:
    I_arr = I

if axarr is None:
    fig, axarr = plt.subplots(ncols=1, nrows=2, sharex=True, sharey=True, figsize=[14, 12])

V1_arr, W1_arr = rk4_xmtr(V0_1, W0_1, h, N, tau, a_arr, b_arr, I_arr)
V2_arr, W2_arr = rk4_rcvr(V0_2, W0_2, h, N, tau, a_arr, b_arr, gamma, V1_arr)

# Plot phase trajectory
axarr[0].scatter(V1_arr, W1_arr,
                 c=np.linspace(10, 0, N),
                 cmap=cm.jet,
                 marker='.')

axarr[1].scatter(V2_arr, W2_arr,
                 c=np.linspace(10, 0, N),
                 cmap=cm.jet,
                 marker='.')

x_bottom, x_top = axarr[0].get_xlim()
y_bottom, y_top = axarr[0].get_ylim()

# Plot nullclines, if a, b, I are constants
#x = np.linspace(x_bottom, x_top)
x = np.linspace(-1.7, 0.5)

try:
    V_nullcline = x - (1/3)*x**3 + I
    axarr[0].plot(x, V_nullcline, color='m', linestyle='--')
except:
    pass

try:
    W_nullcline = (1/a) * (x + b)
    axarr[0].plot(x, W_nullcline, color='k', linestyle='--')
except ZeroDivisionError:
    axarr[0].axvline(-b, y_bottom, y_top, color='k', linestyle='--')

axarr[0].spines['left'].set_position('zero')
axarr[0].spines['bottom'].set_position('zero')
axarr[1].spines['left'].set_position('zero')
axarr[1].spines['bottom'].set_position('zero')
sns.despine()

return V1_arr, W1_arr, V2_arr, W2_arr, axarr
```