Superconductors and Josephson Junctions

Adam Poklemba and Jack Dibachi

Contents

1 Choice of Paper and Summary 2

2 Book Problems and Solutions 3
2.1 Strogatz 4.6.4 . . . 3
2.1.1 Part A . . . 3
2.1.2 Part B . . . 4
2.1.3 Part C . . . 4
2.1.4 Part D . . . 4
2.1.5 Part E . . . 4
2.2 Strogatz 4.6.5 . . . 5
2.3 Strogatz 4.6.6 . . . 6

3 Graphs from Paper 7
3.1 Strogatz Fig. 4.6.4 . . . 7
3.2 Strogatz Phase Portraits CH 8 . . . 8
3.2.1 Strogatz Fig. 8.5.7 . . . 8
3.2.2 Strogatz Fig. 8.5.8 . . . 9
3.2.3 Strogatz Fig. 8.5.9 . . . 9

4 Extended Results 10
4.1 Graphs . . . 10
4.2 Bifurcations and Chaos . . . 13
4.2.1 Why a sin wave? . . . 16
4.3 Results Summary . . . 17

5 Roles 17
5.1 Jack Dibachi . . . 17
5.2 Adam Poklemba . . . 18
5.3 Shared Responsibilities . . . 18

A Python Script 18

1 Choice of Paper and Summary

To examine the nonlinear behavior of Josephson junctions more closely, we chose to investigate and expand upon the findings of the 1968 Stewart article, *Current-voltage Characteristics of Josephson Junctions* [2], as well as Chapters 4.6 and 8.5 of Strogatz's textbook *Nonlinear Dynamics and Chaos* [3].

Josephson junctions are microscopic electronic devices which take advantage of superconductivity to generate currents without applying a voltage. In general, they consist of two superconductors separated by a thin barrier, such as a semiconductor, a metal, or any other material with lower conductance than the separated materials. The electrons in the superconductor, which normally have a near-continuous distribution of energy levels, form Cooper pairs that merge into a single coherent wave function at the superconducting temperature. When two superconductors are weakly coupled, the phase difference $\delta(t)$ in the wave functions generates a difference in energy level between the superconductors. This allows Cooper pairs to tunnel through the barrier, resulting in an electric current $i_0 sin(\delta)$ across the junction without an externally applied voltage.

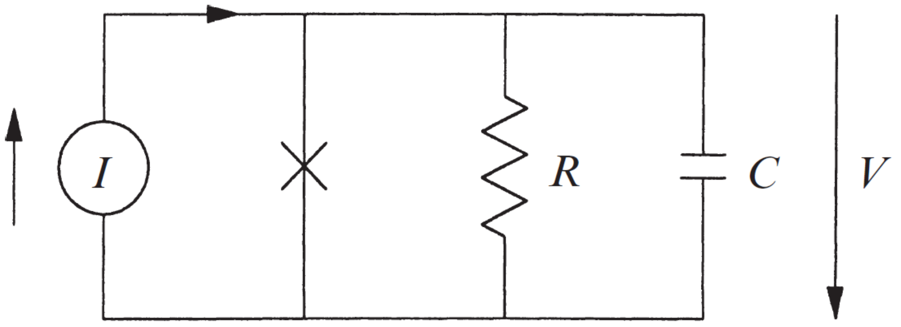


Figure 1: Model representation of a Josephson junction with an applied DC current.

While a steady current can be generated through the junction without an applied voltage, applying enough direct current to the junction results in dynamic, nonlinear behavior. The system can be modeled as a Josephson junction, a parallel resistor, and a parallel capacitor, as depicted in Figure 1. The resistance R and capacitance C represent the combined effects of electric fields between superconductors, magnetic vortices in the junction, Cooper pair tunneling losses, and other losses, which are assumed to be constant. Applying basic circuit theory produces the representative model in Equation 1.

$$C\dot{V} + \frac{V}{R} + i_0 sin(\delta) = I \tag{1}$$

The system can be rewritten strictly in terms of the phase difference by normalizing the coefficients, which gives

$$\ddot{\delta} + \frac{\dot{\delta}}{\tau} + \omega_0^2 sin(\delta) = \frac{\omega_0^2 I}{i_0}, \tag{2}$$

where $\tau = RC$, $\omega_0 = \sqrt{\frac{2\pi i_0}{C\phi_0}}$, and $\phi_0 = \frac{h}{2e}$. This system is analogous to the driven damped pendulum, and exhibits qualitatively different behaviors for three different conditions: heavy damping, intermediate damping, and light damping. For heavy damping, the second derivative term is negligible as $\omega_0\tau \ll 1$, resulting in the exact solution $(\frac{I}{i_0})^2 = (\frac{\bar{\dot{\delta}}}{\omega_0^2\tau})^2 + 1$. In intermediate damping cases, computer integration of the system reveals hysteresis in the voltage as a function of the applied current. As the applied current is increased from 0 to i_0, the voltage remains at zero as in the heavy damping cases, yet exhibit a discontinuous jump from 0 to the curve given by Figure 2. As the applied current is increased, the voltage follows this asymptote, but the applied current must be decreased below i_0 before exhibiting another discontinuous jump back to the vertical line at 0 voltage. As $\omega_0\tau \gg 1$, damping becomes significantly lighter and the point of return from the asymptotic curve to the vertical approaches 0.

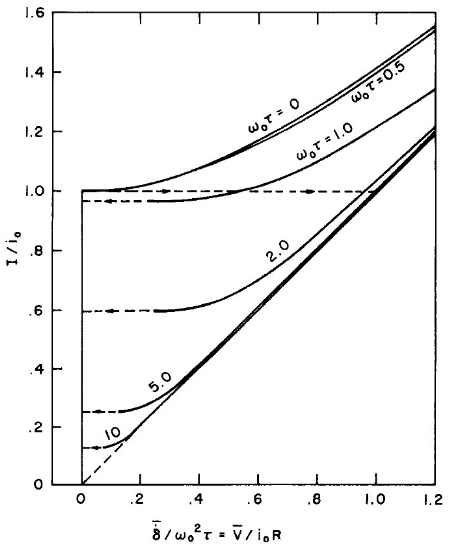


Figure 2: System behavior for applied DC current.

2 Book Problems and Solutions

2.1 Strogatz 4.6.4

Figure 3 shows an array of two identical overdamped Josephson junctions. The junctions are in series with each other, and in parallel with a resistive "load" R. The goal of this exercise is to derive the governing equations for this circuit. In particular, we want to find differential equations for ϕ_1 and ϕ_2.

2.1.1 Part A

By Kirchhoff's Current Law, the sum of the currents through parallel branches of a circuit must equal the input current. In Figure 3, the current source I_b diverts into two parallel branches, thus $I_b = I_a + I_R$.

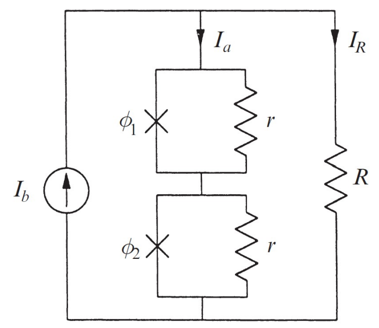


Figure 3: Series Josephson junctions.

2.1.2 Part B

Also by Kirchhoff's Current Law, the sum of the current
through Josephson junction and the resistor is equal to the input current I_a. So by Ohm's Law:

$$I_a = I_{\phi_1} + I_1 \qquad \Rightarrow \qquad I_a = I_c sin(\phi_1) + \frac{V_1}{r} \tag{3}$$

$$I_a = I_{\phi_2} + I_2 \qquad \Rightarrow \qquad I_a = I_c sin(\phi_2) + \frac{V_2}{r} \tag{4}$$

2.1.3 Part C

From page 110 of Strogatz, $V = \frac{\hbar}{2e}\dot{\phi}$, so the voltages for the two different Josephson junctions are given by:

$$V_1 = \frac{\hbar}{2e}\dot{\phi}_1 \qquad V_2 = \frac{\hbar}{2e}\dot{\phi}_2 \tag{5}$$

2.1.4 Part D

Combining Equations 3, 4, and 5 results in the following expression:

$$I_a = I_c sin(\phi_k) + \frac{\hbar}{2er}\dot{\phi}_k \qquad \text{for } k = 1, 2 \tag{6}$$

By basic circuit principles, the voltages across the two Josephson junctions add together, and must equal the voltage drop across the shunt resistor R. By Ohm's Law, $V = IR$, so

$$I_R = \frac{\hbar}{2eR}(\dot{\phi}_1 + \dot{\phi}_2) \tag{7}$$

Applying the results from Part A produces the following expression:

$$I_b = I_c sin(\phi_k) + \frac{\hbar}{2er}\dot{\phi}_k + \frac{\hbar}{2eR}(\dot{\phi}_1 + \dot{\phi}_2) \qquad \text{for } k = 1, 2 \tag{8}$$

2.1.5 Part E

The system can be nondimensionalized by first eliminating the $(\dot{\phi}_1 + \dot{\phi}_2)$ term. This is done by adding the equations for junction 1 and junction 2, then isolating the $(\dot{\phi}_1 + \dot{\phi}_2)$ term.

$$I_b = I_c sin(\phi_1) + \frac{\hbar}{2er}\dot{\phi}_1 + \frac{\hbar}{2eR}(\dot{\phi}_1 + \dot{\phi}_2) \qquad I_b = I_c sin(\phi_2) + \frac{\hbar}{2er}\dot{\phi}_2 + \frac{\hbar}{2eR}(\dot{\phi}_1 + \dot{\phi}_2) \tag{9}$$

$$2I_b = I_c(sin(\phi_1) + sin(\phi_2)) + \left(\frac{\hbar}{2er} + \frac{2\hbar}{2eR}\right)(\dot{\phi}_1 + \dot{\phi}_2) \tag{10}$$

$$\dot{\phi}_1 + \dot{\phi}_2 = \frac{2eR}{\hbar}\left(I_b - I_c sin(\phi_k) - \frac{\hbar}{2er}\dot{\phi}_k\right) \tag{11}$$

Substituting Equation 11 into Equation 10 gives Equation 12, which can be further simplified to Equation 15.

$$2I_b = I_c(sin(\phi_1) + sin(\phi_2)) + \left(\frac{2eR}{\hbar}\right)\left(\frac{\hbar}{2er} + \frac{2\hbar}{2eR}\right)\left(I_b - I_c sin(\phi_k) - \frac{\hbar}{2er}\dot{\phi}_k\right) \tag{12}$$

$$2I_b = I_c(sin(\phi_1) + sin(\phi_2)) + \left(\frac{R+2r}{r}\right)\left(I_b - I_c sin(\phi_k) - \frac{\hbar}{2er}\dot{\phi}_k\right) \tag{13}$$

$$\frac{\hbar(R+2r)}{2er^2}\dot{\phi}_k = I_c(sin(\phi_1) + sin(\phi_2)) + \left(\frac{R+2r}{r} - \frac{2r}{r}\right)I_b - \frac{R+2r}{r}I_c sin(\phi_k) \tag{14}$$

$$\dot{\phi}_k = \frac{2er^2 I_c}{\hbar(R+2r)}\sum_{j=1}^{2} sin(\phi_j) + \frac{2er^2 R I_b}{\hbar(R+2r)} - \frac{2erI_c}{\hbar}sin(\phi_k) \tag{15}$$

Finally, the coefficients in Equation 15 are represented as Ω, a, and K in Equation 16 to achieve a final simplified form for the system.

$$\dot{\phi}_k = \Omega + a sin(\phi_k) + K\sum_{j=1}^{2} sin(\phi_j), \quad \Omega = \frac{2er^2 R I_b}{\hbar(R+2r)}, \; a = -\frac{2erI_c}{\hbar}, \; K = \frac{2er^2 I_c}{\hbar(R+2r)} \tag{16}$$

2.2 Strogatz 4.6.5

To generalize the result from 2.1 for N Josephson junctions in series, prove the system can be written in dimensionless form with appropriate values of Ω, τ, and a.

$$\frac{d\phi_k}{d\tau} = \Omega + a sin(\phi_k) + \frac{1}{N}\sum_{j=1}^{N} sin(\phi_j) \qquad \text{for } k \in [1, N] \tag{17}$$

Generalizing the results from Equation 8 for N Josephson junctions in series produces Equation 18:

$$I_b = I_c sin(\phi_k) + \frac{\hbar}{2er}\dot{\phi}_k + \frac{\hbar}{2eR}\sum_{j=1}^{N}\dot{\phi}_j \qquad \text{for } k \in [1, N] \tag{18}$$

As in Part E of Strogatz 4.6.4, the equations for $k = 1 \cdots N$ are summed, then $\sum_{j=1}^{N}$ is plugged in from Equation 18 to Equation 19 to give Equation 20.

$$NI_b = I_c\sum_{k=1}^{N} sin(\phi_k) + \frac{\hbar}{2er}\sum_{k=1}^{N}\dot{\phi}_k + \frac{\hbar N}{2eR}\sum_{j=1}^{N}\dot{\phi}_j = I_c\sum_{k=1}^{N} sin(\phi_k) + \left(\frac{\hbar}{2er} + \frac{\hbar N}{2eR}\right)\sum_{j=1}^{N}\dot{\phi}_j \tag{19}$$

$$NI_b = I_c\sum_{k=1}^{N} sin(\phi_k) + \left(\frac{\hbar}{2er} + \frac{\hbar N}{2eR}\right)\left(\frac{2eR}{\hbar}\right)\left(I_b - I_c sin(\phi_k) - \frac{\hbar}{2er}\dot{\phi}_k\right) \tag{20}$$

The equation is further simplified to isolate $\dot{\phi}_k$.

$$NI_b = I_c \sum_{k=1}^{N} sin(\phi_k) + \left(\frac{R}{r} + N\right)\left(I_b - I_c sin(\phi_k) - \frac{\hbar}{2er}\dot{\phi}_k\right) \tag{21}$$

$$\left(\frac{R}{r} + N\right)\left(\frac{\hbar}{2er}\right)\dot{\phi}_k = I_c \sum_{k=1}^{N} sin(\phi_k) + \frac{R}{r}I_b - \left(\frac{R}{r} + N\right) I_c sin(\phi_k) \tag{22}$$

$$\dot{\phi}_k = \frac{2eI_c r^2}{\hbar(R+Nr)} \sum_{k=1}^{N} sin(\phi_k) + \frac{2erRI_b}{\hbar(R+Nr)} - \frac{2erI_c}{\hbar} sin(\phi_k) \tag{23}$$

To make the system dimensionless, apply $\dot{\phi}_k = \frac{d\phi_k}{d\tau}\frac{d\tau}{dt}$.

$$\frac{1}{N}\frac{d\tau}{dt} = \frac{2eI_c r^2}{\hbar(R+Nr)} \qquad a\frac{d\tau}{dt} = -\frac{2erI_c}{\hbar} \qquad \Omega\frac{d\tau}{dt} = \frac{2erRI_b}{\hbar(R+Nr)} \tag{24}$$

$$\frac{d\tau}{dt} = \frac{2NeI_c r^2}{\hbar(R+Nr)} \qquad a = -\frac{2erI_c}{\hbar}\frac{\hbar(R+Nr)}{2NeI_c r^2} \qquad \Omega = \frac{2erRI_b}{\hbar(R+Nr)}\frac{\hbar(R+Nr)}{2NeI_c r^2} \tag{25}$$

$$\tau = \frac{2NeI_c r^2}{\hbar(R+Nr)}t \qquad a = -\frac{R+Nr}{Nr} \qquad \Omega = \frac{RI_b}{NrI_c} \tag{26}$$

Thus, the system is successfully made dimensionless of the form:

$$\frac{d\phi_k}{d\tau} = \Omega + asin(\phi_k) + \frac{1}{N}\sum_{j=1}^{N} sin(\phi_j) \qquad \text{for } k \in [1, N] \tag{27}$$

$$\text{with } \tau = \frac{2NeI_c r^2}{\hbar(R+Nr)}t, \quad a = -\frac{R+Nr}{Nr}, \quad \text{and } \Omega = \frac{RI_b}{NrI_c} \tag{28}$$

2.3 Strogatz 4.6.6

Generalize Exercise 4.6.4 to the case where there are N junctions in series, and where the load is a resistor R in series with a capacitor C and an inductor L. Write differential equations for ϕ_k and for Q, where Q is the charge on the load capacitor.

From first principles, one obtains the expressions in Equation 29 and Equation 30 from Kirchhoff's Voltage Law.

$$I_d = C\frac{dV}{dt}, \quad Q = CV_c \quad V_i = L\frac{dI_d}{dt} \quad V_R = I_d R \tag{29}$$

$$V_R + V_c + V_i = NrI_a \quad \Rightarrow \quad I_d R + \frac{Q}{C} + L\frac{dI_d}{dt} = NrI_a, \quad I_d = I_b - I_a \tag{30}$$

$$I_b R + \frac{Q}{C} + L\frac{d(I_b - I_a)}{dt} = (Nr + R)I_a \tag{31}$$

Evaluating the terms in the differential equation and simplifying gives Equation 34, which describes the charge in the load capacitor Q. Applying results from Equations 29 and 30 produce Equation 35.

$$I_bR + \frac{Q}{C} = (Nr + R)I_a + L\frac{dI_a}{dt}, \quad I_a = I_c sin(\phi_k) + \frac{\hbar}{2er}\dot{\phi}_k \tag{32}$$

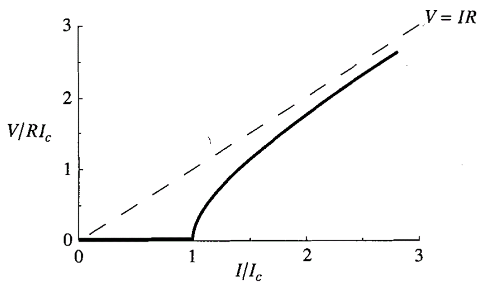
$$I_bR + \frac{Q}{C} = (Nr + R)\left(I_c sin(\phi_k) + \frac{\hbar}{2er}\dot{\phi}_k\right) + L\left(I_c\dot{\phi}_k cos(\phi_k) + \frac{\hbar}{2er}\ddot{\phi}_k\right) \tag{33}$$

$$Q = \frac{LC\hbar}{2er}\ddot{\phi}_k + \left(\frac{C\hbar(Nr + R)}{2er} + LCI_c cos(\phi_k)\right)\dot{\phi}_k + (Nr + R)CI_c sin(\phi_k) - RCI_b \tag{34}$$

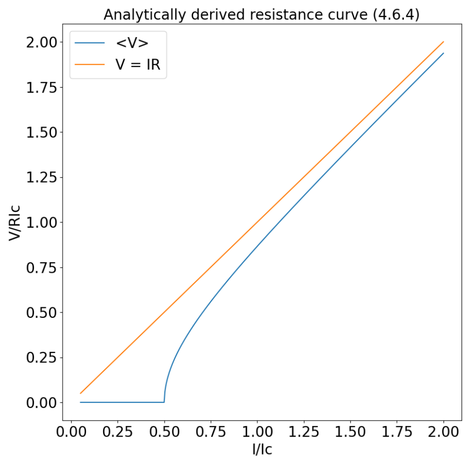
$$\dot{Q} = I_b - I_c sin(\phi_k) - \frac{\hbar}{2er}\dot{\phi}_k \tag{35}$$

3 Graphs from Paper

3.1 Strogatz Fig. 4.6.4



(a)



(b)

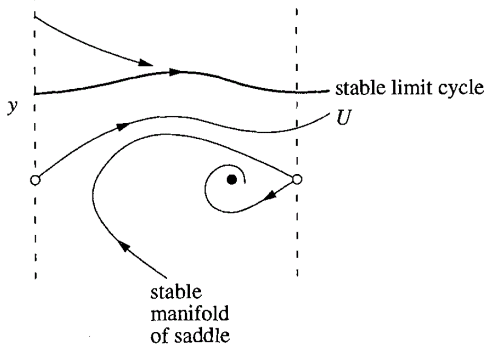
Figure 4: a) Original figure from Strogatz text and b) reproduced figure in Python

This figure from the book is meant to depict the behaviour of the junction's equivalent resistance as a function of $\frac{I}{I_c}$. This section of the book uses a simplified model of the junction, where $\delta'' = 0$. This corresponds to an over-damped oscillator, where transient terms quickly fade away and leave a steady-state result. Plotted here are the steady-state cycle-averaged values for scaled voltage for a collection of $\frac{I}{I_c}$s. The dashed line depicts the behaviour of the circuit if it had only the one resistor, which the simpler model circuit asymptotically approaches. Using the same equation as is in the book, it is straightforward to re-plot the figure in python.

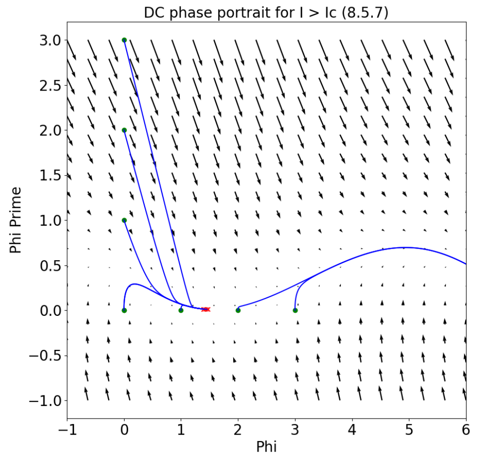
3.2 Strogatz Phase Portraits CH 8

We have now graduated to the complete model described in the paper. It does not ignore transient effects from nonzero δ''. The full system experiences several bifurcations as it passes through $\frac{I}{I_c} = 1$. The succeeding phase plots from the book explore this bifurcation.

3.2.1 Strogatz Fig. 8.5.7



(a)

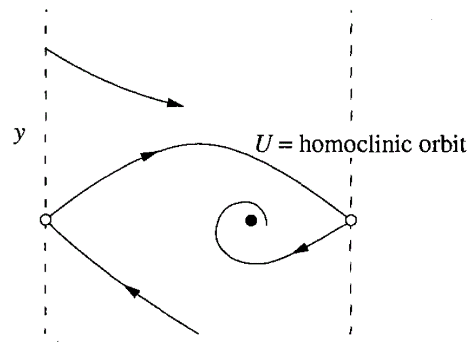


(b)

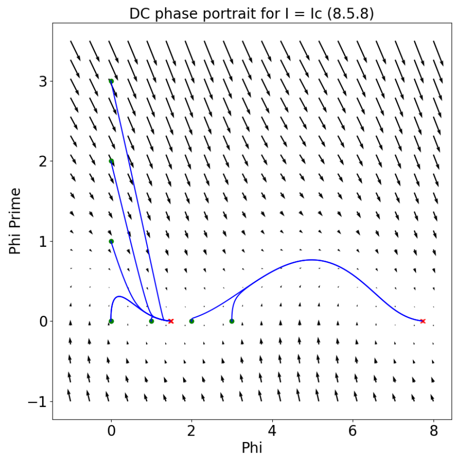
Figure 5: a) Original figure from Strogatz text and b) reproduced figure in Python

We start out thinking about the system when driving is sufficient to overcome critical current. This corresponds to $\frac{I}{I_c} > 1$. Looking at the system from the pendulum point of view, this corresponds to a driving force that can overcome the force of gravity. The system in this situation is bistable - there exist both a stable limit cycle and an attractor. In our pendulum example, the limit cycle exists when there is enough energy in the pendulum at initial conditions to overcome gravity and swing the mass over. Alternatively, the attractor corresponds to initial conditions that don't meet this goal, and all energy is dissipated in the form of friction. As these graphs are more symbolic than actually accurate, our reproduction will look a little different. To create our version of the graph, we initially plot the vector field corresponding to the condition $\alpha = .8$. Then, we select a few representative initial conditions, and plot how they evolve on the vector field. The points on the left half of the plot fall into the attractor, while a few points on the right fall into the stable limit cycle.

3.2.2 Strogatz Fig. 8.5.8



(a)



(b)

Figure 6: a) Original figure from Strogatz text and b) reproduced figure in Python

In this scenario, $\frac{I}{I_c} = 1$. The force of gravity is equal to the driving force, and the stable limit cycle is destroyed, as we cannot put enough energy into the system to combat friction. The pendulum may swing over a couple of times, but it will eventually settle into equilibrium. In the phase plots, this corresponds to the destruction of the stable limit cycle. We can see how the limit cycle that existed in our reproduction of 8.5.7 quickly settles into the attractor on the right half of the plot. The other trajectories settle immediately into the equivalent attractor 2π rad away.

3.2.3 Strogatz Fig. 8.5.9

In this scenario, $\frac{I}{I_c} < 1$. The driving force is always less than the force of gravity, and all trajectories settle into the attractor quickly. The rightmost point tilts into the positive theta direction, however it too is consumed by the attractor.

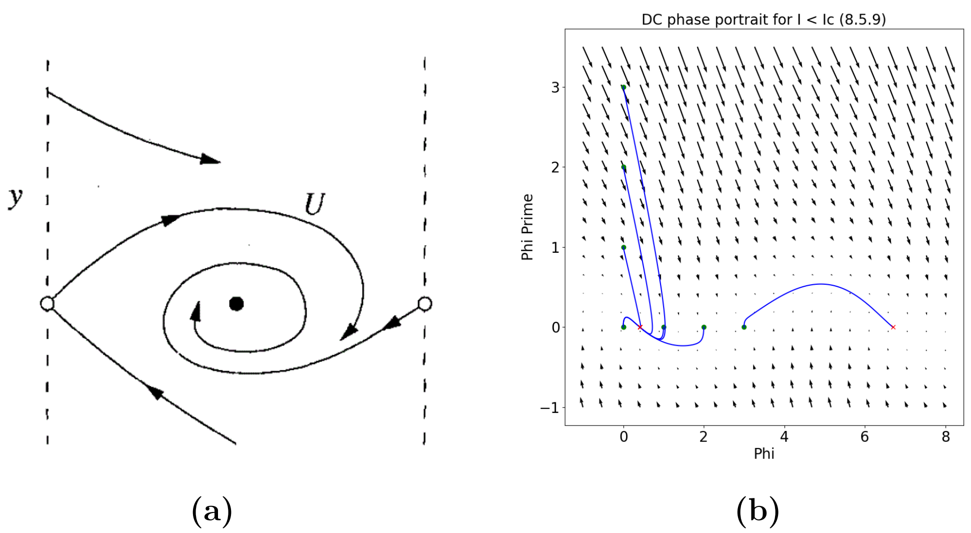


Figure 7: a) Original figure from Strogatz text and b) reproduced figure in Python

4 Extended Results

4.1 Graphs

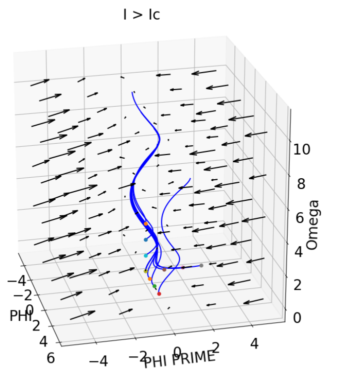
To extend the results of the text and paper, we considered the scenario where the DC current is replaced by an AC current. We model the AC current with $I(t) = I_0 sin(w_0\tau)$. The differential equation takes the following form:

$$\phi'' + \alpha\phi' + sin(\phi) = I(\tau) = I_0 sin(w_0\tau)$$

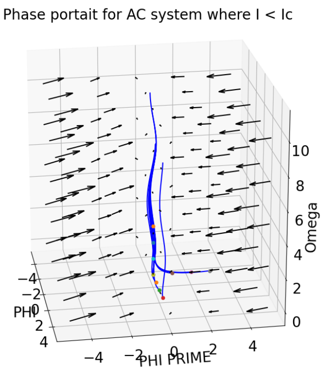
A problem arises immediately: all of the methods of graphical analysis we know for nonlinear systems requires them to be time-invariant! In order to remove the factor of τ, we create the following equivalent system [1]:

$$\begin{aligned} a &= \phi & a' &= \phi' = b \\ b &= \phi' & b' &= \phi'' = I(\tau) - \alpha\phi' - sin(\phi) = I_0 sin(c) - \alpha b + sin(a) \\ c &= w_0\tau & c' &= w_0 \end{aligned}$$

Though the AC system is 3-dimensional, it seems at a first glance that it is not much more complicated than its DC counterpart. However, 3-dimensional nonlinear systems are sometimes subject to chaos, so we must remain on the lookout. We begin our exploration of the system by plotting its phase portrait in 3D.



(a)



(b)

Figure 8: Phase portraits for a) large driving current and b) small driving current

The plots seem to show a very similar story to what we were seeing in the DC system. A larger driving current can result in a stable limit cycle, as we can see in **a)**, whereas in **b)** the same initial conditions result in a rapid damping, and the flows lead into a stable attractor. The only influence the extra ω parameter seems to have is to drag the trajectories statically upwards: what was a stable fixed point becomes a fixed line. Just to be sure, we plot the phase portrait of just two variables: ϕ and ϕ', so that we may more easily observe system behaviour.

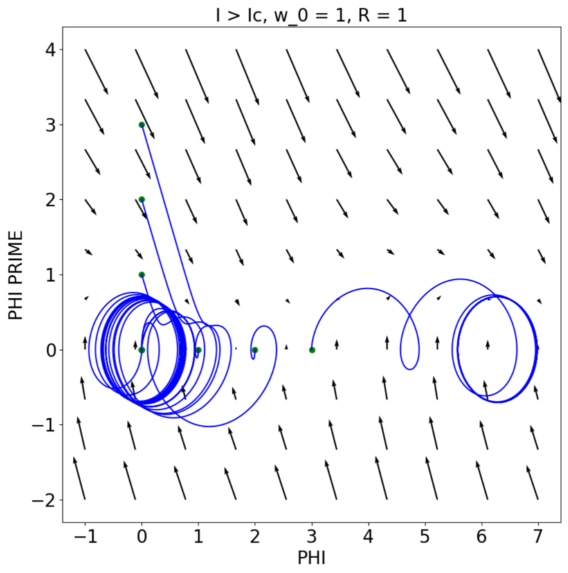


Figure 9: Phase portrait for large driving current

We can clearly see the limit cycles in 9. These limit cycles about $\phi = 0$ replace the fixed points that were there in the previous system. Furthermore, it appears that the limit cycle here has a thickness. This is not what we would expect from looking at a non-chaotic limit cycle, and our initial thoughts about the simplicity of the system need to be re-considered.

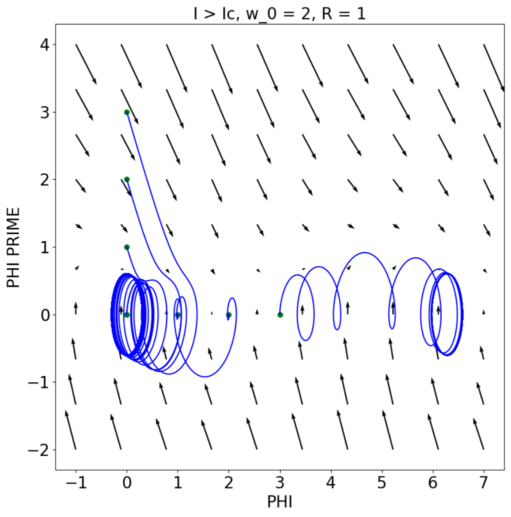


Figure 10: Phase portrait for large driving current, $\omega_0 = 2$

As we increase the parameter ω_0 in 10, we begin to see signs of chaos. The period of the rightmost orbit appears to quadruple, and the thickness of the limit cycles becomes more apparent.

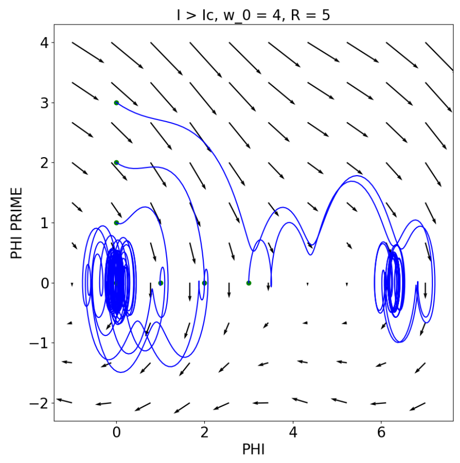


Figure 11: Phase portrait for large driving current, R = 4

Finally, increasing both ω_0 and R, the system seems to undergo several more bifurcations, and clearly exhibits chaos.

4.2 Bifurcations and Chaos

Let's try and find out more about the chaotic behaviour of our system, starting with a bifurcation diagram. We calculate the diagram using the following pseudocode:

Algorithm 1 Calculating Bifurcation Diagram

```
Result: array of fixed points
initialize time range and $n$
initialize static parameters
for $w0$ in list do
    initialize model with $w0$
    while $n > 0$ do
        set random initial conditions
        solve model for initial conditions
        append last point in solution to results
        $n = n - 1$
    end
end
```

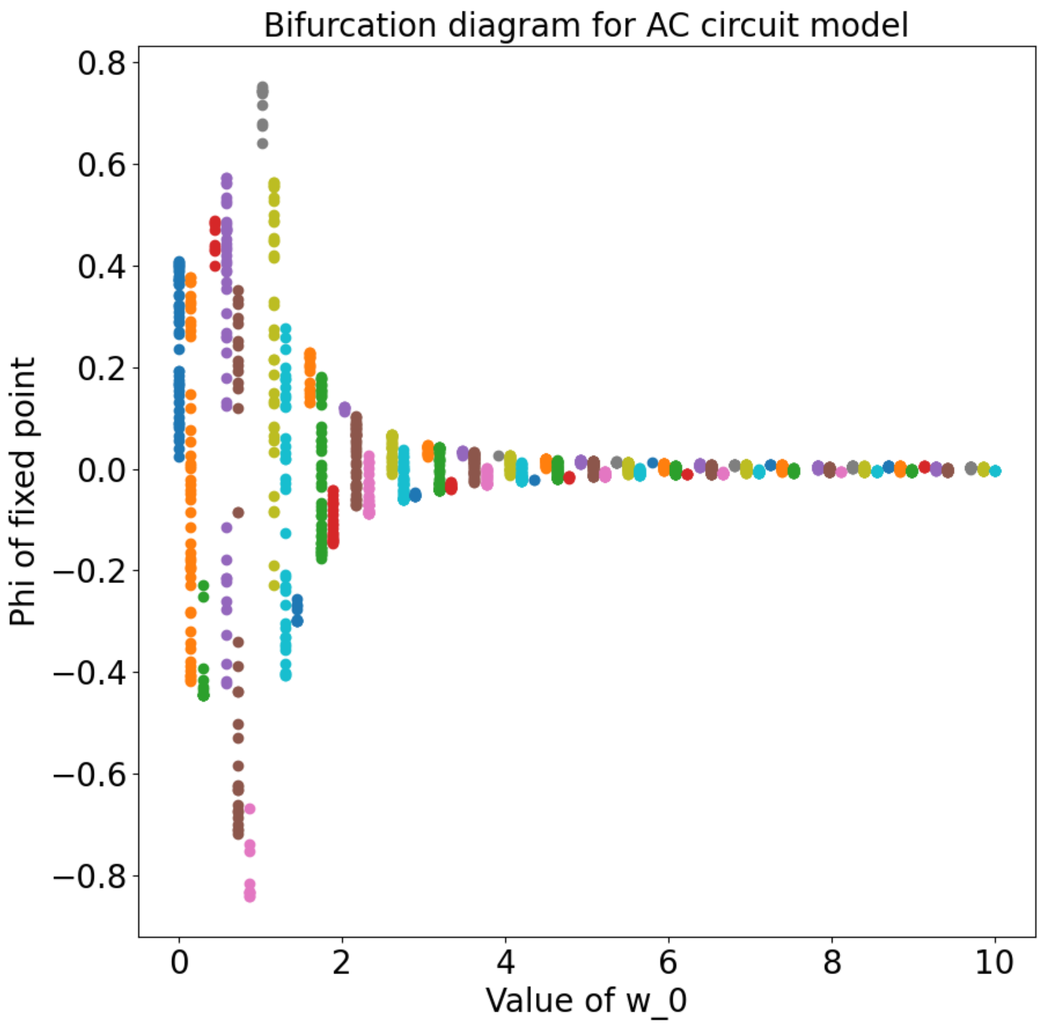
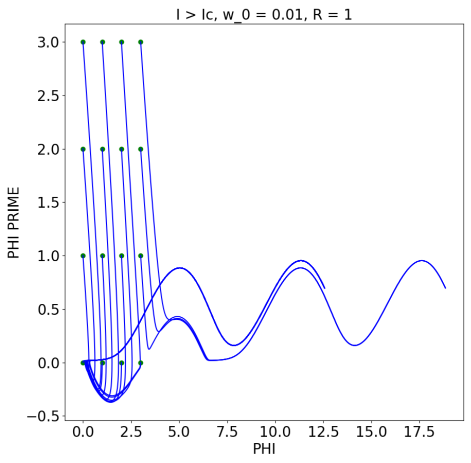
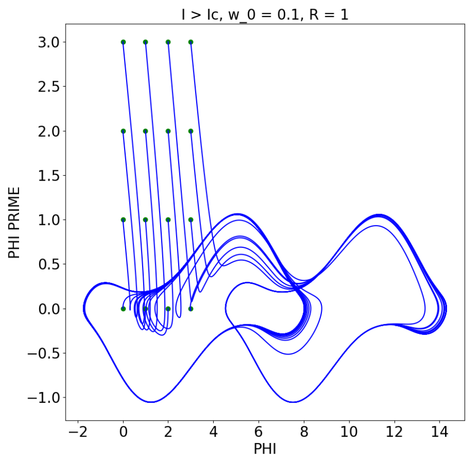


Figure 12: Bifurcation diagram for R = 1

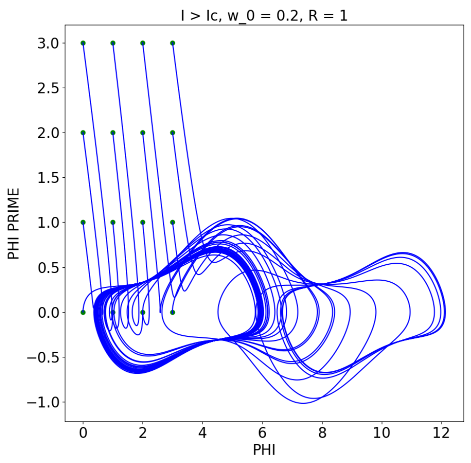
Because there is no easy way to find the locations of all manners of fixed points, we simply went with the assumption that after a sufficiently large time, the value of phi will settle down into an attractor. However, this assumption is dependant upon the length of our integration time being sufficient for the trajectory to settle into a fixed point. Moreover, our calculation method fails completely for other categories of fixed points, such as limit cycles. As such, the diagram is extremely noisy, and not entirely reliable. However there does appear to be a pretty clear bifurcation happening for $0 < \omega_0 < 4$ despite all this. Let's try and explore some representative ω_0 phase plots.



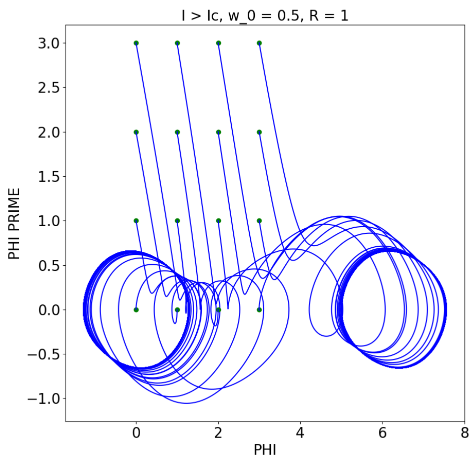
(a)



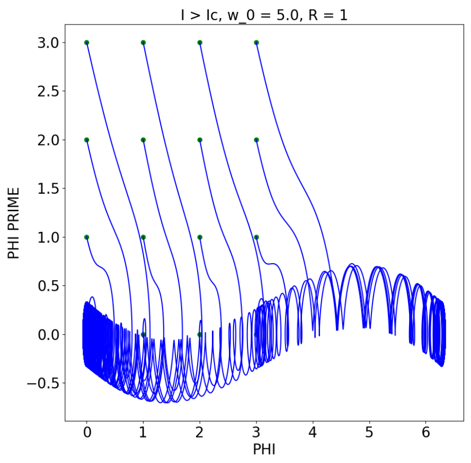
(b)



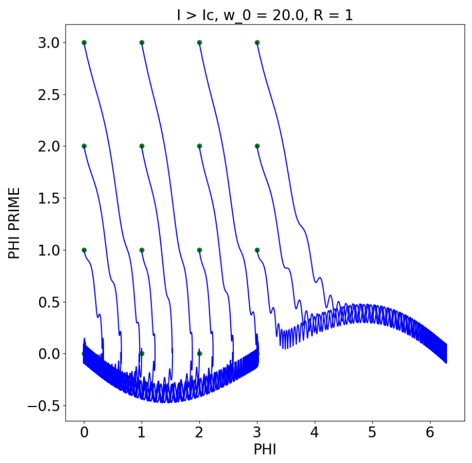
(c)



(d)



(e)



(f)

Figure 13: Exploring bifurcations with w_0

These figures give a solid insight into the behaviour of the system. We continue with analyzing using the pendulum analogy.

a) Initially, the driving frequency is so small that the system closely resembles the DC system. In some trajectories the pendulum swings over the top and enters a limit cycle, and in others it enters the attractor at $\phi = 0$

b) We increase omega, and two very clear limit cycles appear. Each of the cycles represents identical motion, with the right one corresponding to the pendulum initially swinging over the top. Once a point settles into one of these limit cycles, it stays there.

c) The limit cycle corresponding to no swing-over begins to dominate the system. Although these plots may resemble the Lorenz butterfly attractor, trajectories never seem to travel from one attractor to another.

d) The shape of the limit cycles changes drastically. We believe part of this to be due to the limit cycles ceasing to encompass regions where $|\phi| \approx \frac{n\pi}{2}$. Previously, those were the regions where the force of gravity had observably large effects compared to the driving force. These regions are no longer encompassed by the limit cycle because the driving happens too quickly for the pendulum to travel as far.

e) The frequency of forcing continues to increase, bringing with it the aforementioned inertial effects. The pendulum can travel less and less over the course of one driving cycle. The graph begins to resemble a sin wave. It seems that the period doubling behaviour begins somwhere between d) and e)

f) The graph represents a sin wave more and more. Period doubling is getting quite extreme.

4.2.1 Why a sin wave?

As it turns out, the solution to the pendulum system with *no* driving force resembles the sin wave that we were seeing in the phase portraits. That solution is shown in 14. Intuitively, this makes sense - increase the driving frequency too much gives the system no time to respond to the forcing. Let's look at the energy argument.

$$\begin{aligned} W_{forced} = \Delta E_{forced} &= \int_{\mathbf{s}} F_{forced} \cdot ds = \int_{\mathbf{s}} sin(w_0 \mathbf{s}) \cdot ds \\ &\approx \int_{x_i}^{x_f} sin(w_0 x) dx \end{aligned} \tag{36}$$

x_i and x_f here are used as an approximation for a 1-D straight-line path. The energy argument holds for 2 and 3 dimensions, but is not as pretty, so we'll leave it out.

$$W_{forced} \approx -\frac{1}{w_0} [cos(w_0 x)]_{x_i}^{x_f} \tag{37}$$

$$\lim_{w_0 \to \infty} -\frac{1}{w_0}[cos(w_0 x)]_{x_i}^{x_f} = 0 \tag{38}$$

$$\lim_{w_0 \to \infty} \Delta E_{forced} \approx 0 \tag{39}$$

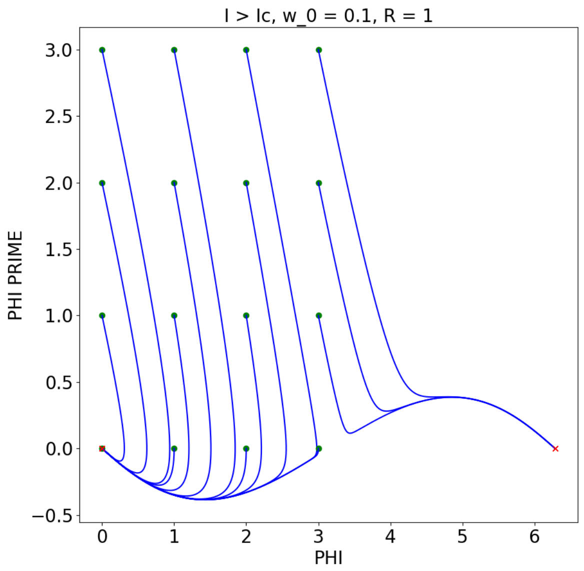


Figure 14: Phase portrait for no forcing

4.3 Results Summary

The AC system behaviour is surprisingly complex when compared to its DC counterpart. The question of whether or not the system can be considered chaotic has not been sufficiently explored, however for now it suffices to say that varying the driving frequency causes multiple difficult to predict bifurcations in the system, including one that looks like period-doubling.

5 Roles

5.1 Jack Dibachi

- wrote summary of Josephson junction functionality and paper findings
- solved Strogatz problems 4.6.4, 4.6.5, and 4.6.6
- organized and formatted document

5.2 Adam Poklemba

- reproduced figures from Stewart paper
- extended Stewart results by evaluating system dynamics with AC current

5.3 Shared Responsibilities

- read Strogatz sections 4.6, 6.7, and 8.5
- discussed paper results and what was learned

References

[1] Gray D. Davidson. “The Damped Driven Pendulum: Bifurcation Analysis of Experimental Data”. In: (2011).

[2] W.C. Stewart. “Current-Voltage Characteristics of Josephson Junctions”. In: *Applied Physics Letters* (Jan. 1968).

[3] Steven H. Strogatz. *Nonlinear Dynamics and Chaos*. CRC Press, 2018.

A Python Script

```
"""
This file contains code to generate all plots for the PHYS106 final project.
All plot-generating code is commented out. The total plots created if you un-
UN-COMMENT WITH CAUTION! THIS ENTIRE FILE TAKES 10+ MINUTES TO RUN!
"""


import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
import matplotlib
from mpl_toolkits.mplot3d import axes3d
import nolds
import warnings


# Shhhh no warnings, it's ok
warnings.filterwarnings("ignore")
```

```
matplotlib.rcParams["figure.figsize"] = (10.0, 10.0)
matplotlib.rcParams["axes.titlesize"] = 20
matplotlib.rcParams["axes.labelsize"] = 20
matplotlib.rcParams["xtick.labelsize"] = 20
matplotlib.rcParams["ytick.labelsize"] = 20
matplotlib.rcParams["legend.fontsize"] = 20

h_bar = 6.62607004e-34 / (2 * np.pi)  # m^2 kg / s
electron_charge = 1.60217662e-19  # Coulombs

alpha, beta, tau, crit_curr, I, RES = 0, 0, 0, 0, 0, 0
simple_period = 0
I0, w = 0, 0


def init(C, R, Ic, in_I=0, in_I0=0, in_w=np.pi / 2):
    """Initialize our reduced model variables"""
    global crit_curr, alpha, tau, I0, w, I, beta, RES, simple_period
    alpha = (h_bar / (2 * electron_charge * Ic * R ** 2 * C)) ** (1 / 2)
    tau = 2 * electron_charge * Ic / (h_bar * C)
    beta = (2 * electron_charge * Ic * R ** 2 * C) / h_bar
    crit_curr = Ic
    I0 = in_I0
    w = in_w
    I = in_I
    RES = R


def model_DC(x, t):
    phi, phidot = x
    dxdt = [phidot, I / crit_curr - alpha * phidot - np.sin(phi)]
    return dxdt


def simple_model_DC(x, t):
    phi = x
    dxdt = I / crit_curr - np.sin(phi)
    return dxdt


def model_AC(x, t):
    phi, phidot, theta = x
    dxdt = [
        phidot,
        I0 * np.sin(theta) / crit_curr - alpha * phidot - np.sin(phi),
        w,
    ]
    return dxdt


def voltage_simple(phidot):
    return crit_curr * RES * phidot


def voltage_simple_analytical(I):
    results = crit_curr * RES * ((I / crit_curr) ** 2 - 1) ** (1 / 2)
    results[I <= crit_curr] = 0
    return results


def lyapunov_exponent(series1, series2, t):
    dz = []
    for i in range(len(series1)):
        dz.append(np.linalg.norm(series1[i] - series2[i]))
    return 1 / t * np.log(dz / dz[0])


def model_NOC(x, t):
    phi, phidot, theta = x
    dxdt = [phidot, -alpha * phidot - np.sin(phi), 0]
    return dxdt


net_time = 50
t = np.linspace(0, net_time, 1000)
init_condit = 0
capacitance = 1e-16
resistance = 1
critical_current = 0.5
current = 0.8
ac_omega = 0.1
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

"""
# NOTE: Calculate figure 4.6.4
equivalent_resistance = []
i_range = np.linspace(crit_curr / 10, 4 * crit_curr, 500)
for I in i_range:
    current = I
    init(
        capacitance,
        resistance,
        critical_current,
        in_I0=current,
        in_w=ac_omega,
        in_I=current,
    )
    sol_dc_simple = odeint(simple_model_DC, init_condit, t)
    equivalent_resistance.append(
        voltage_simple(sol_dc_simple[-1]) / (RES * net_time)
    )


fig, ax = plt.subplots()
plt.plot(i_range / crit_curr, equivalent_resistance)
plt.plot(i_range / crit_curr, i_range * RES)
ax.legend(["<V>", "V = IR"])
plt.xlabel("I/Ic")
plt.ylabel("V/RIc")
plt.title("Purely model-driven resistance curve (4.6.4)")

fig, ax = plt.subplots()
plt.plot(i_range, voltage_simple_analytical(i_range) / (RES))
plt.plot(i_range, i_range * RES)
ax.legend(["<V>", "V = IR"])
plt.xlabel("I/Ic")
plt.ylabel("V/RIc")
plt.title("Analytically derived resistance curve (4.6.4)")
```

```
# NOTE: Calculate 8.5.7
fig, ax = plt.subplots()
net_time = 20
t = np.linspace(0, net_time, 1000)
current = 0.51
capacitance = 8e-17
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

phi, phiprime = np.meshgrid(np.linspace(-1, 6, 20), np.linspace(-1, 3, 20))
phivec, phiprimevec = model_DC([phi, phiprime], 0)

plt.quiver(phi, phiprime, phivec, phiprimevec)
for i in np.linspace(0, 3, 4):
    init_condit = [0, i]
    plt.plot(init_condit[0], init_condit[1], "go")
    results = odeint(model_DC, init_condit, t)
    plt.plot(results[:, 0], results[:, 1], "b")
    plt.plot(results[-1, 0], results[-1, 1], "rx")

    init_condit = [i, 0]
    plt.plot(init_condit[0], init_condit[1], "go")
    results = odeint(model_DC, init_condit, t)
    plt.plot(results[:, 0], results[:, 1], "b")
    plt.plot(results[-1, 0], results[-1, 1], "rx")

plt.xlim(-1, 6)
plt.xlabel("Phi")
plt.ylabel("Phi Prime")
plt.title("DC phase portrait for I > Ic (8.5.7)")

# NOTE: calculate 8.5.8
fig, ax = plt.subplots()
current = 0.5
capacitance = 1e-16
```

```
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)
net_time = 50
t = np.linspace(0, net_time, 1000)

phi, phiprime = np.meshgrid(
    np.linspace(-1, 8, 20), np.linspace(-1, 3.5, 20)
)
phivec, phiprimevec = model_DC([phi, phiprime], 0)
plt.quiver(phi, phiprime, phivec, phiprimevec)
for i in np.linspace(0, 3, 4):
    init_condit = [0, i]
    plt.plot(init_condit[0], init_condit[1], "go")
    results = odeint(model_DC, init_condit, t)
    plt.plot(results[:, 0], results[:, 1], "b")
    plt.plot(results[-1, 0], results[-1, 1], "rx")

    init_condit = [i, 0]
    plt.plot(init_condit[0], init_condit[1], "go")
    results = odeint(model_DC, init_condit, t)
    plt.plot(results[:, 0], results[:, 1], "b")
    plt.plot(results[-1, 0], results[-1, 1], "rx")

plt.xlabel("Phi")
plt.ylabel("Phi Prime")
plt.title("DC phase portrait for I = Ic (8.5.8)")

# NOTE: Calculate 8.5.9
fig, ax = plt.subplots()
current = 0.2
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)


phi, phiprime = np.meshgrid(
    np.linspace(-1, 8, 20), np.linspace(-1, 3.5, 20)
)
phivec, phiprimevec = model_DC([phi, phiprime], 0)
plt.quiver(phi, phiprime, phivec, phiprimevec)
for i in np.linspace(0, 3, 4):
    init_condit = [0, i]
    plt.plot(init_condit[0], init_condit[1], "go")
    results = odeint(model_DC, init_condit, t)
    plt.plot(results[:, 0], results[:, 1], "b")
    plt.plot(results[-1, 0], results[-1, 1], "rx")

    init_condit = [i, 0]
    plt.plot(init_condit[0], init_condit[1], "go")
    results = odeint(model_DC, init_condit, t)
    plt.plot(results[:, 0], results[:, 1], "b")
    plt.plot(results[-1, 0], results[-1, 1], "rx")

plt.xlabel("Phi")
plt.ylabel("Phi Prime")
plt.title("DC phase portrait for I < Ic (8.5.9)")


# NOTE: 3d Under-driven, w = 1
fig = plt.figure()
ax = fig.gca(projection="3d")

net_time = 8
t = np.linspace(0, net_time, 1000)
current = 0.2
ac_omega = 1
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

phi, phiprime, omega = np.meshgrid(
    np.linspace(-5, 5, 5), np.linspace(-5, 5, 5), np.linspace(0, 10, 5),
)
phivec, phiprimevec, omegavec = model_AC([phi, phiprime, omega], 0)

for axis in range(3):
    vec = np.zeros(3)
    vec[axis] = 1
    for i in np.linspace(0, 3, 4):
        init_condit = np.dot(vec, i)
        ax.scatter(init_condit[0], init_condit[1], init_condit[2], "go")
        results = odeint(model_AC, init_condit, t)
        ax.plot(results[:, 0], results[:, 1], results[:, 2], "b")


ax.quiver(
    phi,
    phiprime,
    omega,
    phivec,
    phiprimevec,
    omegavec,
    length=0.1,
    color="black",
)
ax.set_xlabel("PHI")
ax.set_ylabel("PHI PRIME")
ax.set_zlabel("Omega")
ax.set_title("Phase portait for AC system where I < Ic")

# NOTE: 3d over-Driven, w = 1
fig = plt.figure()
ax = fig.gca(projection="3d")
current = 0.9
ac_omega = 1
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

phi, phiprime, omega = np.meshgrid(
    np.linspace(-5, 5, 5), np.linspace(-5, 5, 5), np.linspace(0, 10, 5),
)
phivec, phiprimevec, omegavec = model_AC([phi, phiprime, omega], 0)

for axis in range(3):
    vec = np.zeros(3)
    vec[axis] = 1
    for i in np.linspace(0, 3, 4):
        init_condit = np.dot(vec, i)
        ax.scatter(init_condit[0], init_condit[1], init_condit[2], "go")
        results = odeint(model_AC, init_condit, t)
        ax.plot(results[:, 0], results[:, 1], results[:, 2], "b")


ax.quiver(
    phi,
    phiprime,
    omega,
    phivec,
    phiprimevec,
    omegavec,
    length=0.1,
    color="black",
)
ax.set_xlabel("PHI")
ax.set_ylabel("PHI PRIME")
ax.set_zlabel("Omega")
ax.set_title("I > Ic")


# NOTE: 2d over-driven,  w_0 = 1, R = 1
fig, ax = plt.subplots()

net_time = 50
t = np.linspace(0, net_time, 1000)
current = 0.9
ac_omega = 1
```

```
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

phi, phiprime, omega = np.meshgrid(
    np.linspace(−1, 7, 10), np.linspace(−2, 4, 10), ac_omega
)
phivec, phiprimevec, omegavec = model_AC([phi, phiprime, omega], 0)
ax.quiver(
    phi[:, :, 0],
    phiprime[:, :, 0],
    phivec[:, :, 0],
    phiprimevec[:, :, 0],
    width=0.003,
    color="black",
)


for axis in range(3):
    vec = np.zeros(3)
    vec[axis] = 1
    for i in np.linspace(0, 3, 4):
        init_condit = np.dot(vec, i)
        ax.plot(init_condit[0], init_condit[1], "go")
        results = odeint(model_AC, init_condit, t)
        ax.plot(results[:, 0], results[:, 1], "b")

plt.xlabel("PHI")
plt.ylabel("PHI PRIME")
plt.title("I > Ic, w_0 = 1, R = 1")

# NOTE: 2d over−driven, w_0 = 2, R = 1
fig, ax = plt.subplots()
current = 0.9
ac_omega = 2
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

phi, phiprime, omega = np.meshgrid(
    np.linspace(-1, 7, 10), np.linspace(-2, 4, 10), ac_omega
)
phivec, phiprimevec, omegavec = model_AC([phi, phiprime, omega], 0)
ax.quiver(
    phi[:, :, 0],
    phiprime[:, :, 0],
    phivec[:, :, 0],
    phiprimevec[:, :, 0],
    width=0.003,
    color="black",
)

for axis in range(2):
    vec = np.zeros(3)
    vec[axis] = 1
    for i in np.linspace(0, 3, 4):
        init_condit = np.dot(vec, i)
        ax.plot(init_condit[0], init_condit[1], "go")
        results = odeint(model_AC, init_condit, t)
        ax.plot(results[:, 0], results[:, 1], "b")

plt.xlabel("PHI")
plt.ylabel("PHI PRIME")
plt.title("I > Ic, w_0 = 2, R = 1")

# NOTE: 2d over-driven, w_0 = 4, R = 5
fig, ax = plt.subplots()
current = 0.9
ac_omega = 4
resistance = 5
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

phi, phiprime, omega = np.meshgrid(
    np.linspace(-1, 7, 10), np.linspace(-2, 4, 10), ac_omega
)
phivec, phiprimevec, omegavec = model_AC([phi, phiprime, omega], 0)
ax.quiver(
    phi[:, :, 0],
    phiprime[:, :, 0],
    phivec[:, :, 0],
    phiprimevec[:, :, 0],
    width=0.003,
    color="black",
)

for axis in range(2):
    vec = np.zeros(3)
    vec[axis] = 1
    for i in np.linspace(0, 3, 4):
        init_condit = np.dot(vec, i)
        ax.plot(init_condit[0], init_condit[1], "go")
        results = odeint(model_AC, init_condit, t)
        ax.plot(results[:, 0], results[:, 1], "b")

plt.xlabel("PHI")
plt.ylabel("PHI PRIME")
plt.title("I > Ic, w_0 = 4, R = 5")

# NOTE: 2d over-driven, w_0 exploration
net_time = 100
t = np.linspace(0, net_time, 1000)
current = 0.9
resistance = 1
ac_of_interest = np.array([0.01, 0.1, 0.2, 0.5, 5, 20, 500])
for ac_omega in ac_of_interest:
    fig, ax = plt.subplots()
    init(
        capacitance,
        resistance,
        critical_current ,
        in_I0=current ,
        in_w=ac_omega ,
        in_I=current ,
    )


    for theta_0 in np.linspace(0, 3, 4):
        for theta_dot_0 in np.linspace(0, 3, 4):
            init_condit = np.array([theta_0 , theta_dot_0 , 0])
            ax.plot(init_condit[0], init_condit[1], "go")
            results = odeint(model_AC, init_condit , t)
            ax.plot(results[:, 0], results[:, 1], "b")
            plt.plot(results[-1, 0], results[-1, 1], "rx")

    plt.xlabel("PHI")
    plt.ylabel("PHI PRIME")
    plt.title("I > Ic, w_0 = {}, R = 1".format(ac_omega))


# NOTE: calculate no-driving system
fig , ax = plt.subplots()
for theta_0 in np.linspace(0, 3, 4):
    for theta_dot_0 in np.linspace(0, 3, 4):
        init_condit = np.array([theta_0 , theta_dot_0 , 0])
        ax.plot(init_condit[0], init_condit[1], "go")
        results = odeint(model_NOC, init_condit , t)
        ax.plot(results[:, 0], results[:, 1], "b")
        plt.plot(results[-1, 0], results[-1, 1], "rx")
plt.xlabel("PHI")
plt.ylabel("PHI PRIME")
plt.title("I > Ic, w_0 = {}, R = 1".format(ac_omega))


# NOTE: Estimate max lyapunov exponent for DC circ
fig , ax = plt.subplots()
exponents = []
current = 0.9
init(
    capacitance ,
    resistance ,
    critical_current ,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)
for i in range(10):
    init_1 = np.random.rand(2) * 5
    t_max = 1e2
    t = np.linspace(0, t_max, 100)
    sol_1 = odeint(model_DC, init_1, t)
    exponentx = nolds.lyap_r(sol_1[:, 0])
    exponenty = nolds.lyap_r(sol_1[:, 1])
    exponent = [exponentx, exponenty]
    exponents.append(max(exponent))

plt.plot(exponents, "o")
plt.title("DC circuit Lyapunov")
plt.xlabel("Run number")
plt.ylabel("Max exp for that run")


# NOTE: Estimate max lyapunov exponent for AC circ
# NOTE: This one can take a while to run!
fig, ax = plt.subplots()
exponents = []
omegas = np.linspace(1, 2, 10)
for ac_omega in omegas:
    init(
        capacitance,
        resistance,
        critical_current,
        in_I0=current,
        in_w=ac_omega,
        in_I=current,
    )
    exponents_temp = []
    for i in range(15):
        init_1 = np.random.rand(3) * 5
        t_max = 1e2
        t = np.linspace(0, t_max, 100)
        sol_1 = odeint(model_AC, init_1, t)
        exponentx = nolds.lyap_r(sol_1[:, 0])
        exponenty = nolds.lyap_r(sol_1[:, 1])
        exponentz = nolds.lyap_r(sol_1[:, 2])
        exponent = [exponentx, exponenty, exponentz]
        exponents_temp.append(max(exponent))
    exponents.append(max(exponents_temp))

plt.figure(2)
plt.plot(omegas, exponents)
plt.title("AC circuit Lyapunov")
plt.xlabel("AC omega")
plt.ylabel("Max exponent")


# NOTE: looking for a  map
# NOTE: This one can take a while to run!
# 2d over-driven,  w_0 = 1, R = 1
fig, ax = plt.subplots()
current = 0.2
ac_omega = 22.444444
init(
    capacitance,
    resistance,
    critical_current,
    in_I0=current,
    in_w=ac_omega,
    in_I=current,
)

for axis in range(3):
    vec = np.zeros(3)
    vec[axis] = 1
    for i in np.linspace(0, 3, 4):
        init_condit = np.dot(vec, i)
        results = odeint(model_AC, init_condit, t)
        results_plus1 = np.zeros(np.shape(results))
        results_plus1[:-1, :] = results[1:, :]
        diff = results - results_plus1
        ax.plot(results[:, 0], diff[:, 0], "o")

plt.xlabel("x_n")
plt.ylabel("x_n+1")
plt.title("I < Ic, w_0 = 1, R = 1")
```

```
# NOTE: let's try and get a nice bifurcation diagram
# NOTE: This one can take a while to run!
fig, ax = plt.subplots()
theta_max = []
omegas = np.linspace(0, 20, 50)

capacitance = 4e-14
current = 0.8
for ac_omega in omegas:
    init(
        capacitance,
        resistance,
        critical_current,
        in_I0=current,
        in_w=ac_omega,
        in_I=current,
    )
    theta_max_of_omega = []
    for i in range(30):
        init_1 = np.random.rand(2)
        t_max = 100
        n_samp = 200
        t = np.linspace(0, t_max, n_samp)
        sol_1 = odeint(model_DC, init_1, t)

        while np.any(np.abs(sol_1) > np.pi):
            sol_1[sol_1 > np.pi] = sol_1[sol_1 > np.pi] - 2 * np.pi
            sol_1[sol_1 < -np.pi] = sol_1[sol_1 < -np.pi] + 2 * np.pi

        max_theta = sol_1[
            -1, 0
        ]
        theta_max_of_omega.append(max_theta)
    theta_max.append(theta_max_of_omega)
    ac_array = np.empty(np.shape(theta_max_of_omega))
    ac_array.fill(ac_omega)
    print(ac_omega)
    plt.scatter(ac_array, theta_max_of_omega)
plt.xlabel("Value of w_0")
plt.ylabel("Phi of fixed point")
plt.title("Bifurcation diagram for AC circuit model")
```

```
"""

plt.show()
```