Restricted Three Body Problem

Lizzy Jones and Avani Anne

18 December 2023

1 Introduction

A generalized version three body problem involves three massive bodies, each of which have three degrees of freedom. This system is not only incredibly difficult to solve, but its complexity also makes it difficult to find interesting cases/properties of the system. For this project, we are studying the simplified planar circular restricted 3-body problem, which retains many interesting properties of the generalized problem without the added complexity. Our system consists of two massive bodies, Jupiter and the Sun, where Jupiter orbits the Sun in a circular path. We also introduce a third body of negligible mass which could represent, for example, a comet or a satellite and which orbits in the same plane as Jupiter and the Sun. Our goal is to study the motion of this third massless particle for different initial conditions of the system.

2 Theory

2.1 Mathematical Model

For simplicity, we are choosing to study the system in a frame of reference such that center of mass is stationary at the origin. Though realistically the center of mass would have translational motion through space, this motion does not add any interesting properties of the system so we are choosing to ignore it for simplicity.

Figure 1 illustrates our mathematical model for the system. The Sun is positioned at $(-r_s, 0)$, Jupiter is at $(r_j, 0)$, and the total distance between the Sun and Jupiter is $R = r_s + r_j$. The masses of the sun and Jupiter are M_s and M_j, respectively, and the total mass of the system is $M = M_s + M_j$. The position of the particle is defined by the vector $\mathbf{r}$, which can be decomposed into the two general coordinates x and y.

Before deriving an equation for the Lagrangian, it is helpful to find expressions for the velocity of the rotating frame, $\boldsymbol{\omega}$, and the potential energy on the massless particle due to gravitational force from Jupiter and Saturn, $U(\mathbf{r})$.

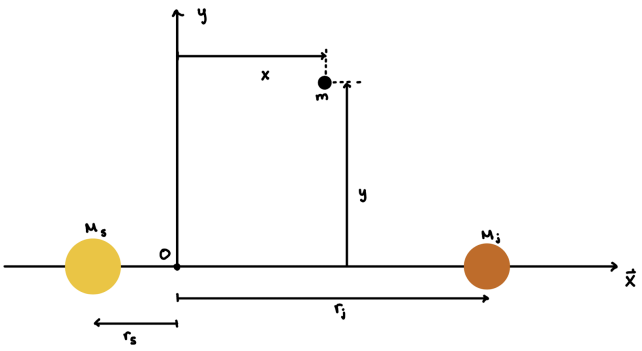

Figure 1: Mathematical Model. This diagram illustrates the sun, Jupiter, and massless particle. The system's center of mass is at the origin, where distances r_s, r_j, x, and y are defined from.

To find $\boldsymbol{\omega}$, we consider a non-rotating frame of reference in which the stationary center of mass is at the origin.
The lagrangian for such a system is described by Equation 9.26 from Helliwell and Sahakian's Classical Mechanics:

$$L = \frac{1}{2}m{\mathbf{v_{rot}}}^2 + m\mathbf{v_{rot}} \cdot (\boldsymbol{\omega} \times \mathbf{r}) + \frac{1}{2}m\omega^2 r^2 - \frac{1}{2}(\boldsymbol{\omega} \cdot \mathbf{r})^2 - U(\mathbf{r}) \tag{1}$$

First, we can find an expression for ω, the rotational velocity of the frame of reference. This value must be constant because angular momentum is conserved. We derive this value using relationships for circular motion from Newtonian mechanics:

$$a_s = G\frac{M_j}{R^2} = \omega_s^2 r_s \tag{2}$$

$$\omega_s = \sqrt{\frac{GM_j}{R^2 r_s}} \tag{3}$$

We can do the same for the acceleration of Jupiter:

$$a_j = G\frac{M_s}{R^2} = \omega_j^2 r_j \tag{4}$$

$$\omega_j = \sqrt{\frac{GM_s}{R^2 r_j}} \tag{5}$$

Note that because the Sun's angular speed in Equation 3 and Jupiter's angular speed in Equation 5 must be equal,

$$\frac{M_s}{r_j} = \frac{M_j}{r_s} \tag{6}$$

Therefore given numerical values for R, M_s, and M_j, we determine a numerical value for ω. Combining equations (3) and (5), we can also re-express ω in terms of more general variables as follows:

$$\omega = \sqrt{\frac{GM}{R^3}} \tag{7}$$

2.2 Deriving the Lagrangian

Returning to Equation 1 for the Lagrangian, we can now find each of the five terms individually:

$$\frac{1}{2}m\mathbf{v_{rot}}^2 = \frac{1}{2}m(\dot{x}\hat{x} + \dot{y}\hat{y})^2 \tag{8}$$

$$m\mathbf{v_{rot}} \cdot (\boldsymbol{\omega} \times \mathbf{r}) = m(\dot{x}\hat{x} + \dot{y}\hat{y}) \cdot (\omega\hat{z} \times (x\hat{x} + y\hat{y})) \tag{9}$$

$$\frac{1}{2}m\omega^2 r^2 = \frac{1}{2}m\omega^2(\dot{x}\hat{x} + \dot{y}\hat{y})^2 \tag{10}$$

$$-\frac{1}{2}(\boldsymbol{\omega} \cdot \mathbf{r})^2 = -\frac{1}{2}(\omega\hat{z} \cdot (x\hat{x} + y\hat{y})) = 0 \tag{11}$$

$$-U(\mathbf{r}) = G(\frac{M_j m}{\sqrt{(r_j - x)^2 + y^2}} + \frac{M_s m}{\sqrt{(r_s + x)^2 + y^2}}) \tag{12}$$

We can combine these terms and divide by m to express the Lagrangian for a massless particle:

$$\frac{L}{m} = \frac{1}{2}(\dot{x} + \dot{y})^2 + \omega(\dot{y}x - \dot{x}y) + \frac{1}{2}\omega(x^2 + y^2) + G(\frac{M_j}{\sqrt{(r_j - x)^2 + y^2}} + \frac{M_s}{\sqrt{(r_s + x)^2 + y^2}}) \tag{13}$$

2.3 Euler Lagrange Equations

Now, we are able to find the Euler-Lagrange equations.

For x:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = \frac{\partial L}{\partial x} \tag{14}$$

$$\ddot{x} = 2\dot{y}\sqrt{\frac{Gm_j}{R^2 r_s}} + x(\frac{Gm_j}{R^2 r_s}) + \frac{GM_j(r_j - x)}{((r_j - x)^2 + y^2)^{\frac{3}{2}}} - \frac{GM_s(r_s + x)}{((r_s + x)^2 + y^2)^{\frac{3}{2}}} \tag{15}$$

For y:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{y}} = \frac{\partial L}{\partial y} \tag{16}$$

$$\ddot{y} = -2\dot{x}\sqrt{\frac{Gm_j}{R^2 r_s}} + y(\frac{Gm_j}{R^2 r_s}) - \frac{GM_j y}{((r_j - x)^2 + y^2)^{\frac{3}{2}}} - \frac{GM_s y}{((r_s + x)^2 + y^2)^{\frac{3}{2}}} \tag{17}$$

2.4 Non-dimensionalizing

Let us define $t' = \frac{t}{\tau}, x' = \frac{x}{l}$, and $y' = \frac{y}{l}$ where $\tau = \frac{1}{\omega}$ and $l = R$. Then, we can define dimensionless $\ddot{x}$ and $\ddot{y}$:

$$t' = \frac{t}{\tau} \tag{18}$$

$$x' = \frac{x}{l} \tag{19}$$

$$\dot{x}' = \frac{l}{\tau}\frac{dx'}{dt} \tag{20}$$

$$\ddot{x}' = \frac{l}{\tau^2}\frac{d^2x'}{dt'^2} \tag{21}$$

$$y' = \frac{y}{l} \tag{22}$$

$$\dot{y}' = \frac{l}{\tau}\frac{dy'}{dt} \tag{23}$$

$$\ddot{y}' = \frac{l}{\tau^2}\frac{d^2y'}{dt'^2} \tag{24}$$

$$\tag{25}$$

Recall that $w = \sqrt{\frac{GM}{R^3}}$. Since the units of w are s^{-1}, then we can yield a unitless w with the following definition:

$$w' = \tau w \tag{26}$$

$$. \tag{27}$$

Since $\tau = \frac{1}{w}$, $w' = 1$.
Finally, we can nondimensionalize our distances and masses by using the fact that $M_s r_s = M_j r_j$ and $r_s + r_j = R = l$. Using these relationships, we can see $\frac{M_s R}{M_j + M_s} = r_j$.

$$r_j' = \frac{r_j}{l} = \frac{r_j}{R} = \frac{M_s R}{R(M_j + M_s)} = \frac{M_s}{M} \tag{28}$$

$$r_s' = \frac{r_s}{l} = \frac{M_j}{M} \tag{29}$$

$$(GM_j)' = \frac{GM_j\tau^2}{l^3} = \frac{M_j}{M_s + M_j} = \frac{M_j}{M} \tag{30}$$

$$(GM_s)' = \frac{GM_s\tau^2}{l^3} = \frac{M_s}{M_s + M_j} = \frac{M_s}{M} \tag{31}$$

We can substitute these dimensionless variables and the fact that $M_s + M_j = M$ to get our

Lagrangian in terms of only dimensionless variables:

$$\frac{L}{m} = \frac{l^2}{\tau^2}(\frac{1}{2}(\dot{x'^2} + \dot{y'^2}) + (x'\dot{y'} - y'\dot{x'}) + \frac{1}{2}(x'^2 + y'^2) + \frac{1}{M}(\frac{M_s}{((r'_s + x')^2 + y'^2)^{\frac{1}{2}}} + \frac{M_j}{((r'_j - x')^2 + y'^2)^{\frac{1}{2}}}) \tag{32}$$

We can define $\mu = \frac{M_j}{M}$ and $1 - \mu = \frac{M_s}{M}$. Then, let $r_1 = ((x + \mu)^2 + y^2))^{\frac{1}{2}}$ and let $r_2 = ((x - 1 + \mu)^2 + y^2))^{\frac{1}{2}}$. Our Lagrangian then simplifies so that it is dependent on only a single parameter, μ:

$$\frac{L}{m} = \frac{l^2}{\tau^2}(\frac{1}{2}(\dot{x'^2} + \dot{y'^2}) + (x'\dot{y'} - y'\dot{x'}) + \frac{1}{2}(x'^2 + y'^2) + (\frac{1-\mu}{r_1} + \frac{\mu}{r_2}) \tag{33}$$

.

We can also define a U_{eff} as:

$$U_{eff} = -(x'\dot{y'} - y'\dot{x'}) - \frac{1}{2}(x'^2 + y'^2) + (\frac{1-\mu}{r_1} - \frac{\mu}{r_2}) \tag{34}$$

.

2.5 Simplified Final Equations

Then the Lagrangian becomes:

$$\frac{L}{m} = \frac{l^2}{\tau^2}(\frac{1}{2}(\dot{x'^2} + \dot{y'^2}) - U_{eff}) \tag{35}$$

.

Now, we can write the nondimensionalized Euler-Lagrange equations. So:

$$\ddot{x'} = 2\dot{y'} + x' + \frac{\mu(1-\mu-x')}{((1-\mu-x')^2 + y'^2)^{\frac{3}{2}}} - \frac{(1-\mu)(\mu+x')}{((\mu+x')^2 + y'^2)^{\frac{3}{2}}} \tag{36}$$

$$\ddot{y'} = -2\dot{x'} + y' + \frac{\mu y'}{(1-\mu-x')^2 + y'^2)^{\frac{3}{2}}} - \frac{(1-\mu))y'}{((\mu+x')^2 + y'^2)^{\frac{3}{2}}} \tag{37}$$

The Hamiltonian can be found by performing a Legendre transform on the Lagrangian: $H = \sum_i p_i \dot{q_i} - L$ where p_i is the generalized momentum $\frac{\partial L}{\partial \dot{q_i}}$. Using non-dimensionalized units,

$$\frac{H}{m} = \frac{l^2}{\tau^2}(\frac{1}{2}(\dot{x'}^2 + \dot{y'}^2) - \frac{1}{2}(x'^2 + y'^2) - \frac{1-\mu}{r1} + \frac{\mu}{r2}) \tag{38}$$

$$\frac{H}{m} = \frac{l^2}{\tau^2}(\frac{1}{2}(\dot{x'}^2 + \dot{y'}^2) + U_{eff}) \tag{39}$$

Since $\frac{\partial H}{\partial t} = 0$, the Hamiltonian is conserved (although it is not equal to total mechanical energy). When calculating the value of the Hamiltonian for the computational portion of our project, we ignored the $\frac{l^2}{\tau^2}$ term since it is a scalar and makes the numbers very small.

3 Analysis

We used numerical integration in python to solve for the state variables of our system. We then looked for interesting cases with varying initial conditions. We created plots to show position, phase space, and velocity in order to better understand the system.

We also calculated and plotted Hill regions to visualize the allowed regions for a particle given specific initial conditions. While the Hamiltonian is not equal to energy, the Hamiltonian acts as pseudo-energy for the restricted three body problem (Ross). For a particle to move freely, the Hamiltonian must be greater than or equal to the effective potential. This region is known as the "Hill region". On the boundary of this region, the Hamiltonian is exactly equal to the effective potential, so that the velocity of the particle is zero. When the Hamiltonian is less than the effective potential, the particle would need imaginary velocity in order to move, which is nonphysical. So these "forbidden regions", colored grey on our diagrams, show where the particle is not allowed to move.

3.1 Circular Path

The simplest case for the system is when the particle moves in a circular orbit of radius x_0 around the sun. This occurs when the mass of Jupiter is set to zero and the particle begins with tangential velocity v such that $\frac{mv^2}{x_0}$ is equal to the gravitational acceleration due to the sun. For initial conditions, we arbitrarily chose $x_0 = \frac{r_j}{2R}$. To simplify the system, we used $y_0 = \dot{x}_0 = 0$. Finally, we found the initial $\dot{y}_0$ needed to keep the particle in a circular orbit. In a stationary frame, $\dot{y}_0 = x_0\sqrt{\frac{GM_s}{x_0^3}}$ as determined by Newtonian mechanics for circular motion. Since we are analyzing motion in a frame rotating at speed ω, we instead have $\dot{y}_0 = x_0(\sqrt{\frac{GM_s}{x_0^3}} - \omega)$.

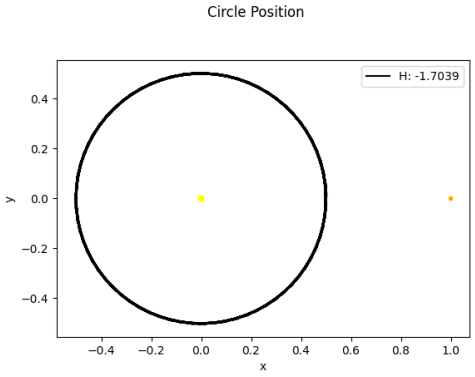

Figure 2: A circular trajectory with $\dot{y} = .915$, $x_0 = \frac{r_j}{2R}$, $y_0 = \dot{x} = 0$, and H = -1.704.

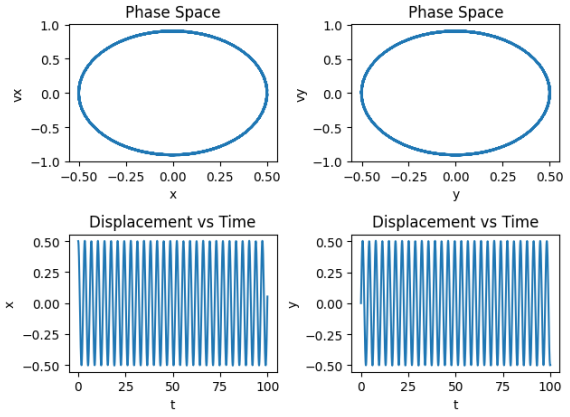

Figure 3: Phase Space and Displacement Plots for a circular trajectory

After verifying that these initial conditions produced a circular path, we added back the

mass of Jupiter. Because the mass of Jupiter is small compared to the sun and our initial position x_0 places the particle far from Jupiter, the orbit is still nearly circular. Our results for these initial conditions are plotted in Figures 2 and 3, where we ran the simulation for 100 radians of non-dimensionalized time.

Figure 2 shows the trajectory of the particle in the xy plane. As predicted, this path is circular about the Sun. In Figure 3, the top two plots represent the phase space of the particle whereas the bottom two plots show the x and y positions of the particles with respect to time. The phase space plots are elliptical, similar to the phase space plot of a pendulum undergoing simple harmonic motion. This is because the particle is moving in a closed circular orbit so its motion is harmonic. This oscillatory nature is even more apparent in the position vs. time plots, where the particle's displacement is sinusoidal.

We used circular motion as the base case for the rest of our analysis. We calculated the Hamiltonian to be -1.704, which we used as an initial test value for other cases. Additionally, the initial value for $\dot{y}$ is 0.915, which we use in future cases even as we vary the value of x_0.

3.2 Varying Initial x Values

To get an idea of different initial conditions that might be interesting, we plotted a range of initial x values while keeping the other three initial values constant at y $= \dot{x} = 0$ and $\dot{y} = 0.915$.

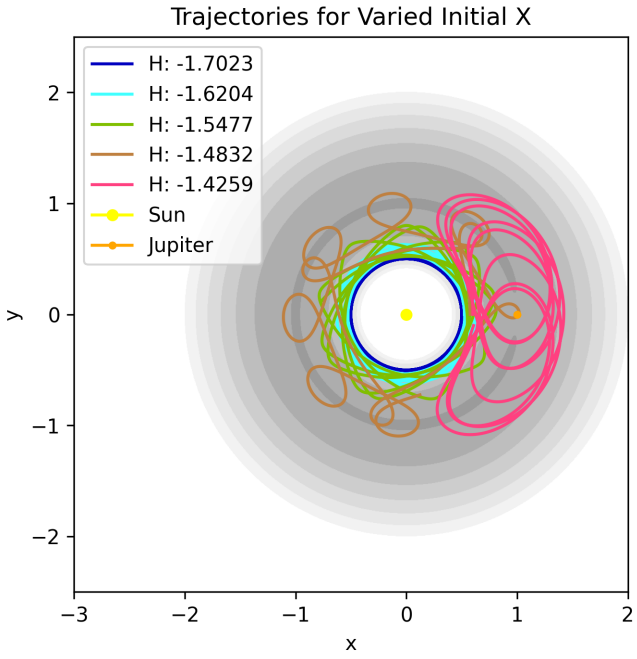

Figure 4: Trajectories for varied initial x conditions corresponding to different Hamiltonian values, H superimposed on constant Hamiltonian contour lines.

Figure 4 shows a set of trajectories for initial x values varying from 0.50 to 0.60 (corresponding to Hamiltonians varying from -1.7023 to -1.4259, respectively) over a time scale of 50 radians. We also plotted contours of constant Hamiltonian values in a manner analogous to a Jacobi contour plot to illustrate the energy levels and corresponding "forbidden region" for each trajectory. This plot illustrates how small variances in initial conditions can create widely varied outcomes. Specific cases are analyzed further in the following sections.

3.3 Bound Weak Coupling

For the initial conditions, $x_0 = .567$, $\dot{x} = 0$, y = 0, and H = -1.503, the trajectory and phase space plots are illustrated in Figures 5 and **??**. The simulation was run for 550 radians of nondimensionalized time. The particle's trajectory is constrained inside the Hills region, where H > U_{eff}. When the particle reaches the boundary, its velocity in the rotating frame is equal to 0, H = U_{eff}, and it continues in the opposite direction to remain within the Hills region (Ross). Because the particle does not leave the region, even when running the simulation for much longer time intervals, the particle is in a bound state.

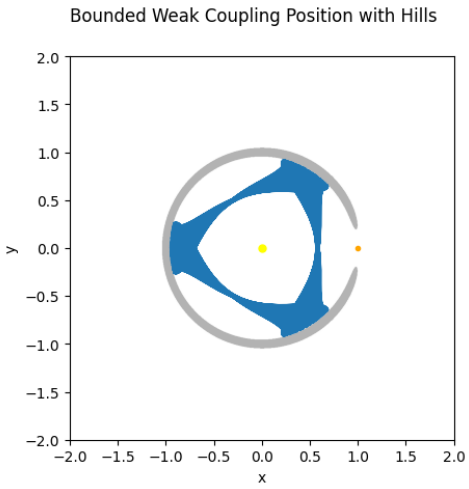

Figure 5: A bounded weak coupling trajectory with $x_0 = .567$, $\dot{x} = 0$, y = 0, and H = -1.503

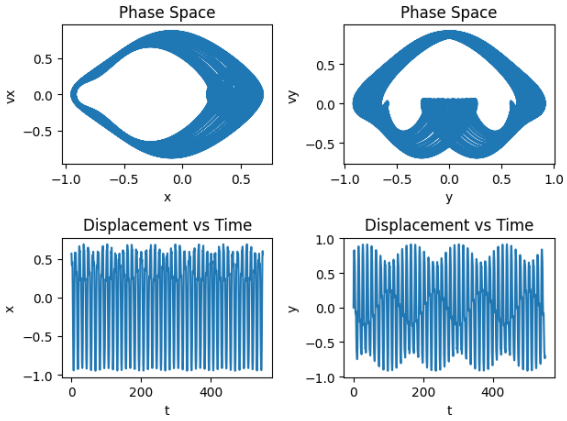

Figure 6: Bounded Weak Coupling Phase Space + Displacement

We can also look at the phase space and displacement plots for this system as shown in Figure 6. In the x vs. t and y vs. t plots there is both a slow and rapid oscillation frequency, which mirrors the behavior of a weakly-coupled oscillator. Using previously written code to perform a Fast Fourier Transform on the displacement plots, we found that the slow oscillation had a period of about 12 seconds and the fast oscillation had a period of about 6 seconds. A single period of slow oscillation corresponds to an almost complete cycle in the v_x vs. x, v_y vs. y, and x vs. y plots. While the particle does not return to the exact same position after each cycle, it returns to a position close to its original one. The particle

then moves through a similar path to its initial path, but slightly shifted. This creates the shape appearing in Figure 5.

3.4 Spiral

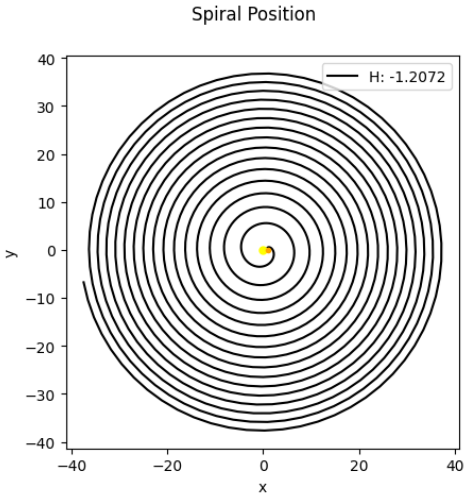

Figure 7: A spiral trajectory with $x_0 = .74$, $\dot{x}$ = 0, y = 0, = .915, and H = -1.207.

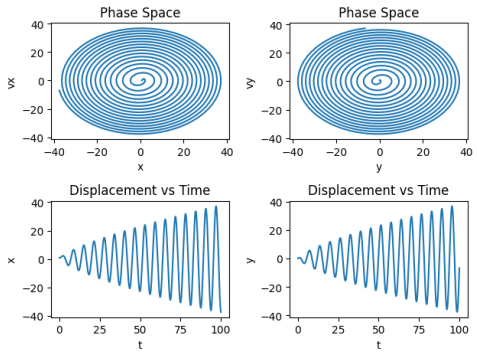

Figure 8: Phase Space and Displacement for Spiral in Figure 7

The particle adopts a spiraling motion for high values of the Hamiltonian, as illustrated in Figure 13. This is because the Hamilton is correlated with the energy in the system, so if the particle has enough initial energy, it overcomes the gravitational force of the Sun and Jupiter and escapes outward. Unlike previous cases in which the particle's orbit is bound, in this scenario the particle is not in a bound state. For initial conditions of $y = 0$, $\dot{x} = 0$, and $\dot{y} = 0.915$, the particle transitions from a bound state to a spiral for between $x_0 = 0.73$ and $x_0 = 0.74$, which corresponds to Hamiltonian values of -1.218 and -1.207 respectively. A plot for the latter case is shown in Figure 7. In the phase space plot in Figure 8, the spirals become more closely spaced as the particle moves outward. According to Liouville's theorem, area in phase space is conserved, meaning that a small square of initial values beginning near the initial condition we chose would smear out into very thin lines as time progresses.

We also performed a Fast Fourier Transform and found that the oscillation period is 6.281 radians, which is nearly 2π radians. However, this may be a coincidence because the oscillation period for the circular case is 3.472 radians which does not have any correlation to π.

3.5 Unbound Chaos

For the next trajectory, we set $x_0 = .88$, $\dot{x} = .092$, $y_0 = 0$, and $\dot{y} = .14163$ which corresponds to a Hamiltonian value of -1.515. We modeled these initial conditions after the ones in the paper "Phase reconstruction in the restricted three-body problem" (Gidea et. al). Our results are presented in Figures 9 and 10.

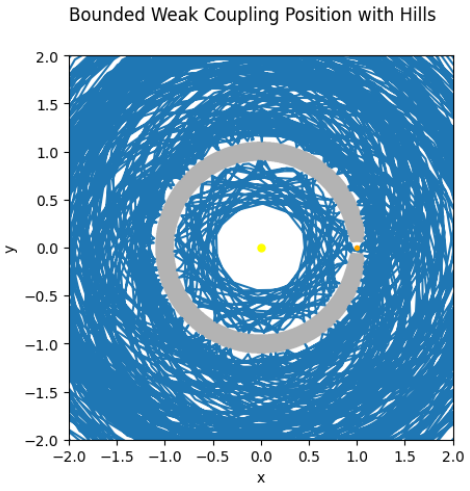

Figure 9: An unbound chaotic trajectory with $x_0 = .88$, $\dot{x} = .092$, y = 0, $\dot{y} = .14163$, and $H = -1.515$. Initial conditions from Gidea et. al. The grey region denotes unreachable particle positions.

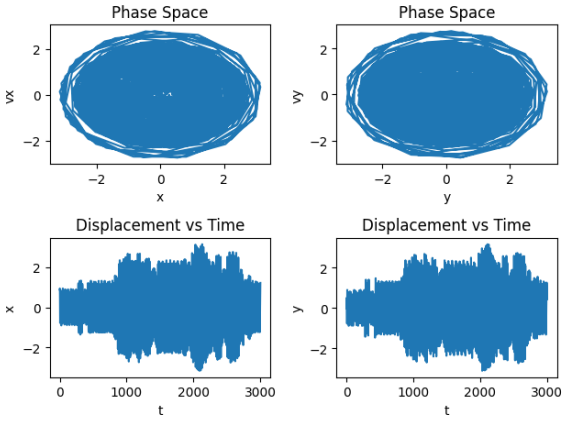

Figure 10: Phase Space + Displacement for unbound chaotic trajectory Figure 9

As illustrated in Figure 9, the grey restricted region is an almost complete ring around the sun and Jupiter. In this scenario, the particle is no longer bounded to a closed orbit by the restricted region. When the particle reaches a position close enough to Jupiter, it is able to escape the inside of the ring and move outside the ring.

Based on the random behaviour within the x vs. $\dot{x}$, y vs. $\dot{y}$, and displacement plots, this trajectory can be classified as chaotic. Furthermore, with long time periods, the particle moves further away from the orbits, modeling an unbound state. As is visible in the phase space and displacement plots, on long time scales there is no obvious signs of regular oscillation in the particle's motion. Interestingly, though, the shape of the phase space plot remains largely elliptical, similar to that of a simple harmonic oscillator.

Though the motion is chaotic, it exhibits some regular oscillatory behavior, especially on short time scales. To analyze the motion further, we performed a fast Fourier transform (FFT) on the displacement plots to determine the frequencies present. We ran the FFT from $t = 0$ to $t = 550$ radians of non-dimensionalized time, as shown in Figure 11. As is visible in the plot, there are two dominant frequencies present during this time interval. From time zero to when the particle exits the ring at $t = 275$ radians, the frequency is

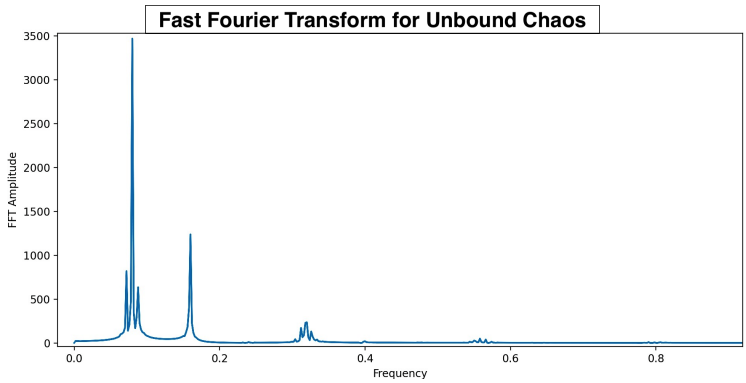

Figure 11: The Fast Fourier Transform for 550 radians of the unbound chaotic trajectory from Figure 9.

approximately 0.111, corresponding to a period of 9.01 radians. This corresponds to the larger peak on the FFT diagram. From the time the particle exits the ring until $t = 550$ radians, the dominant frequency is 0.161, corresponding to a period of approximately 6.21 radians. Over time, the particle changes frequency, sometimes maintaining a consistent frequency for random lengths of time as it remains bound in a particular region before escaping to a new area.

If we slightly alter the initial conditions, the motion of the particle changes dramatically. For example, using initial conditions $x_0 = 0.84$, $\dot{x} = .092$, y = 0, $\dot{y} = 0.14163$, we obtain the graph in Figure 12.

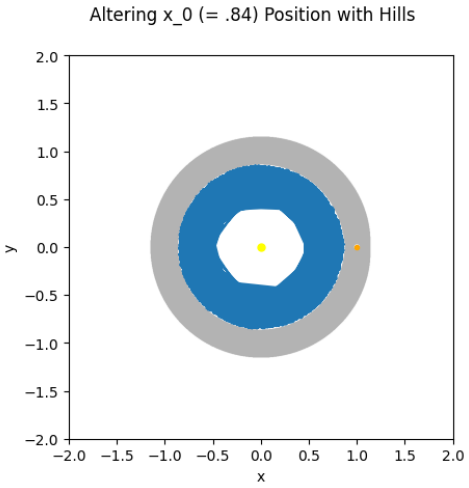

Figure 12: An bound trajectory with $x_0 = .84$, $\dot{x} = .092$, y = 0, $\dot{y} = .14163$, and H = -1.515

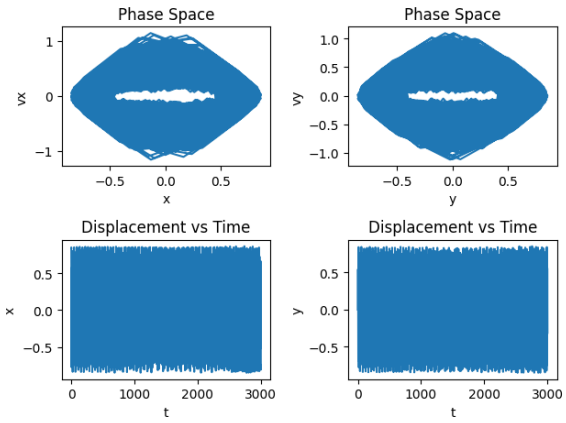

Figure 13: Phase Space + Displacement for bound trajectory in Figure 12

Note that we only changed x_0 by .04, but the outcome is completely different to the previous scenario. Based on the Hill space contour in the plot, this particle's motion is completely constrained inside of the ring, modeling a bound state.

Additionally, if we increase $\dot{y}$ by a small increment (e.g. $\dot{y} = .17$), while keeping all other variables the same, we will get a completely unbound state as modeled in Figure 14.

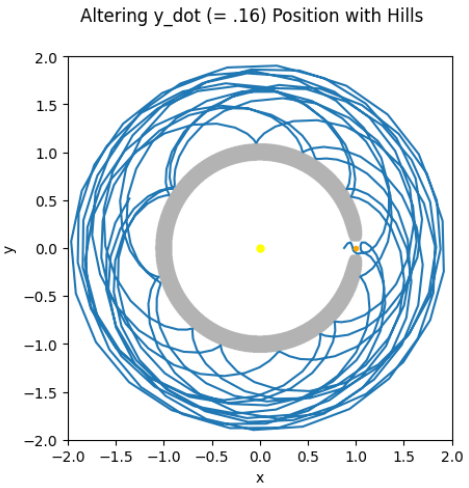

Figure 14: A bound trajectory with $x_0 = .88$, $\dot{x} = .092$, y = 0, $\dot{y} = .16$, and $H = -1.51$. This trajectory has the same conditions as figures 9 and 14 except for $\dot{y}$. The grey denotes the unreachable particle positions.

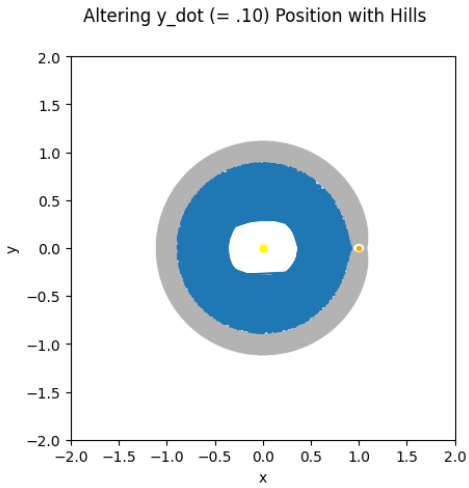

Figure 15: A bound trajectory with $x_0 = .88$, $\dot{x} = .092$, y = 0, $\dot{y} = .10$, and H = -1.52. This trajectory has the same conditions as figure 9 except for $\dot{y}$. The grey denotes the unreachable particle positions.

However, if we decrease $\dot{y}$ by a small increment (e.g. $\dot{y} = .10$), we will again yield a bound state as modeled in figure 15 . Interestingly, altering $\dot{x} = .092$ and y = 0 by small increments do not seem to assist in yielding bound states.

However, if we drastically change x and y, we again yield bound states. For example, if we say x = 0 and y = .88, we will yield a bound state. Thus, which initial conditions we alter at certain points can affect the state's bound nature.

3.6 Mass Variation

Our final case consisted of varying the mass of Jupiter. To understand what a more evenly-balanced system might look like, we set $\mu = 0.487$ so that Jupiter is nearly the same mass as the Sun.

We then plotted the Hamiltonian contour lines as shown in Figure 16 to understand what the bound states look like for this scenario. As the figure shows, the contour lines are nearly symmetric about the Sun and Jupiter. Because Sun and Jupiter have similar masses, they exert similar gravitational force on the particle which results in a symmetric energy cost in the xy plane. The light contours correspond to low Hamiltonian values whereas the

dark contours correspond to high values: as the particle gains more energy, it is able to access more locations in space resulting in smaller "forbidden regions". We also plotted the trajectory of a particle for initial conditions $x = \dot{x} = 0$, $y = .092$, and $\dot{y} = 0.1416$. This resulted in a Hamiltonian value of -1.988, which allowed the particle to travel in between the two planets but kept it in a bound orbit.

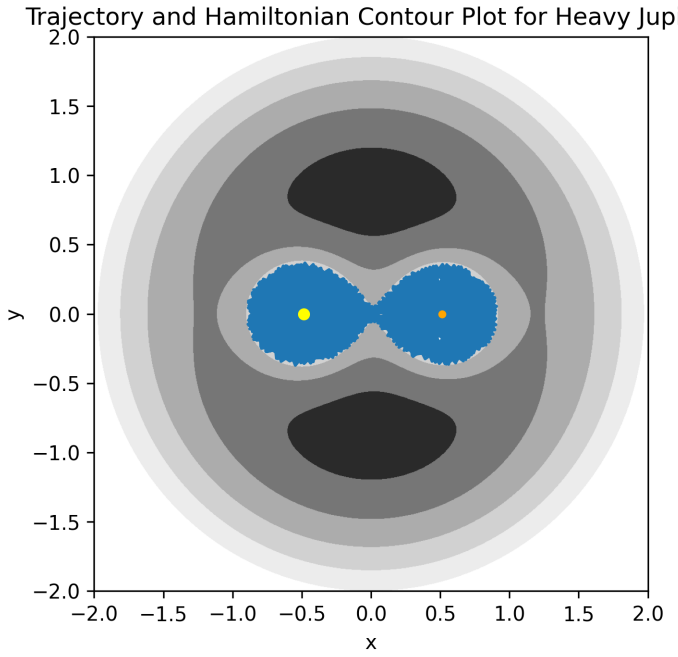

Figure 16: The blue region highlights a massless particle's trajectory for a heavy Jupiter. The grey contour lines represent different Hill's regions for varying Hamiltonian values.

We repeated this experiment for different initial values of x and $\dot{y}$. Figure 17 uses the same initial conditions state above, where the Hamiltonian contour is plotted as a single line. As is visible in the figure, the particle's trajectory reaches zero as it hits the contour line and it does not have enough energy to cross it. Figure 18 uses initial conditions $x = -0.88$ and $\dot{y} = 0.25$. The value of the Hamiltonian for this case is $H = -2.001$. This value is slightly lower than the previous case, shrinking the region in which the particle is able to travel so that it has two separated lobes. The particle begins near the sun and cannot cross the center-line to Jupiter because it cannot cross the Hamiltonian contour line. Finally, Figure 19 uses initial conditions $x = 0.90$ and $\dot{y} = 0.22$. The value of the Hamiltonian is $H = -2.001$, which is identical to the previous case. However, the initial x value positions the particle on the right lobe of the plot, constraining it to the region around Jupiter. In this case, the particle exhibits a periodic, resonant motion as it orbits, creating a star-shaped trajectory.

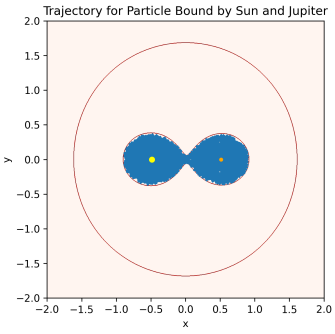

Figure 17: Trajectory for Particle Bound by Sun and Jupiter where the particle's initial position is in the center of the system. The red lines represent the Hill's region boundaries.

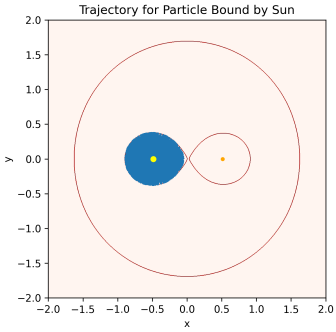

Figure 18: Trajectory for Particle Bound by Sun and Jupiter where the particle begins closer to the sun. The red lines represent the Hill's region boundaries.

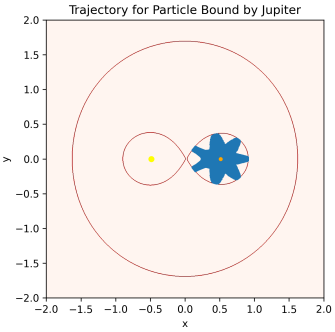

Figure 19: Trajectory for Particle Bound by Sun and Jupiter where the particle begins closer to Jupiter. The red lines represent the Hill's region boundaries.

4 Conclusion

The motion of a particle as part of the three body problem has been studied for a range of initial positions, velocities, and masses. Depending on its initial conditions, the particle has certain trajectories and regions of space accessible to it. Circular path illustrates a simple harmonic oscillation of the particle about the sun and mirrors the behavior of a pendulum. There is also bound weak coupling wherein the particle moves within the Hills region, exhibiting both slow and fast oscillation. The spiral case illustrated regular unbound motion where the particle had sufficient energy to escape the gravitational pull of Jupiter and the Sun. The unbound chaos case demonstrated how slightly altering initial conditions can cause the state of the trajectory to easily shift from bound to unbound states. Finally, changing the mass of Jupiter to match the Sun's mass resulted hourglass-shaped Hill regions which were symmetric about the center line. This system might give insight to a binary star system, for example, where both stars are of similar mass. There are many unique trajectories possible created by the Hill's region shaped like flowers, limacons, and beans. Further explorations might include understanding these shapes, understanding resonance with relation to chaos, and looking more closely at characterizing regularity within chaos at small time scales.

A Works Cited

Gidea, Marian, et al. Phase Space Reconstruction in the Restricted Three-Body Problem, web.ma.utexas.edu/mp_arc/c/07/07-58.pdf. Accessed 18 Dec. 2023.

Helliwell, T. M., and V. V. Sahakian. "Modern Classical Mechanics." Higher Education from Cambridge University Press, Cambridge University Press, 30 Nov. 2020, www.cambridge.org/highereducation/books/modern-classical-mechanics/A51650BB7F0983EE3D155B4C82B2B32A.

Inside Mines, inside.mines.edu/fs_home/tohno/teaching/PH505_2011/Paper_TianyuanGuan.pdf. Accessed 18 Dec. 2023.

Ross, Shane. \3-Body Problem Jacobi Constant, Zero Velocity Curves, Hill Regions of Possible Motion | Topic 6." YouTube, YouTube, 3 July 2022, www.youtube.com/watch?v=_pzpf_j6ZVM.

The Circular Restricted Three-Body Problem - Northern Arizona University, jan.ucc.nau.edu/~ns46/student/2010/Frnka_2010.pdf. Accessed 18 Dec. 2023.

\The Jacobi Constant." The Jacobi Constant - Orbital Mechanics & Astrodynamics, orbital-mechanics.space/the-n-body-problem/jacobi-constant.html. Accessed 17 Dec. 2023.

B Source Code

```
#%%
"""phys111_computational_project.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/14MznZ_fyZWUGuIn66vopVbCk8y6ffgFl

Phys 111 Computational Project:
Restricted 3 Body Problem

things to ask:
- incorperating non-dimensionalizing parameters in code?
"""

import numpy as np
from scipy.integrate import odeint
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import pandas as pd
import random
from matplotlib import cm
```

```
import matplotlib.cbook as cbook
import matplotlib.colors as colors

from scipy.optimize import curve_fit
from scipy.fft import fft, fftfreq


#%%
# define initial values/time bounds

# can I figure out how to non-dimensionalize so that G = 1??

G = 6.7*10**-11
Mj = 1.9*10**30
Ms = 2.0*10**30
M = Mj + Ms
R =  7.78e11 #778000000*10**3
rs = Mj*R/(Mj+Ms) #location of COM from sun (set sun at origin)
rj = R - rs

mu =  Mj / M


#%%

def Veff(vec, mu):
    x, vx, y, vy = vec

    r1 = ((mu + x)**2+y**2)**(1/2)
    r2 = (((1 - mu) - x)**2+y**2)**(1/2)

    # effective potential
    V =  -.5*(x**2 + y**2) - (1-mu)/r1 - mu/r2

    return V

def Veff_derivs(vec, mu):
    x, vx, y, vy = vec

    d1 = ((mu+x)**2 + y**2)**(3/2)
    d2 = ((1-mu-x)**2+y**2)**(3/2)

    Veff_dx = x - (1-mu)*(mu+x)/d1 + mu*(1-mu-x)/d2
    Veff_dy = y - (1-mu)*y/d1 - mu*y/d2

    return Veff_dx, Veff_dy

#%% Defining the vector field

def vectorfield(vec, t, mu):
    """
    Defines the differential equations for the coupled spring-mass system.
```

```
75
76      Arguments:
77          vec :  vector of the state variables:
78                     w = [x1,y1,x2,y2]
79          t :  time
80          p :  vector of the parameters:
81                     p = [G, Mj, Ms, R, rs, rj, w]
82      """
83      x, vx, y, vy = vec
84      Veff_dx, Veff_dy = Veff_derivs(vec, mu)
85      #Mj, Ms, M = p
86
87      # new version with just mu
88      vx_dot = 2*vy + Veff_dx
89      vy_dot = -2*vx + Veff_dy
90
91      # our old version
92      # vx_dot = 2*vy + x + (Mj/M)*((Ms/M)-x)*(((Ms/M)-x)**2 + y**2)**(-3/2) -
          (Ms/M)*((Mj/M)+x)*(((Mj/M)+x)**2 + y**2)**(-3/2)
93      # vy_dot = -2*vx + y - (Mj/M)*y*(((Ms/M)-x)**2 + y**2)**(-3/2) - (Ms/M)*
        y*(((Mj/M)+x)**2 + y**2)**(-3/2)
94
95      # Create f = (x',vx',y',vy'):
96      f = [vx,
97           vx_dot,
98           vy,
99           vy_dot]
100     return f
101
102 #%%
103
104 def solve_odes(mu, time_vec, init_cond_vec):
105     # vector of parameters
106     #param_vec = [Mj, Ms, M]
107
108     # time vector
109     t_start, t_stop, num_points = time_vec
110     t = np.linspace(t_start, t_stop, num_points)
111
112     # solving the equation
113     wsol = odeint(vectorfield, init_cond_vec, t, args=(mu,))
114
115     # saving data to arrays
116     x = wsol[:,0]
117     vx = wsol[:,1]
118     y = wsol[:,2]
119     vy = wsol[:,3]
120
121     return t, [x, vx, y, vy]
122
123
124 def ham(state_var_vec, mu):
```

```
    """ham calculates the hamiltonian of the system
    """
    x, vx, y, vy = state_var_vec

    # r1 and r2
    r1 = ((mu + x)**2+y**2)**(1/2)
    r2 = (((1-mu) - x)**2+y**2)**(1/2)

    # effective potential
    V =  -.5*(x**2 + y**2) - (1-mu)/r1 - (mu)/r2

    #Hamiltonian
    H = .5*(vx**2 + vy**2) + V
    return H

def H_to_vy(x,y,vx,H, mu):
    """given values for H and x0, calculates vx
    """
    #x, vx, y, vy = state_var_vec

    # r1 and r2
    r1 = ((mu + x)**2+y**2)**(1/2)
    r2 = (((1-mu) - x)**2+y**2)**(1/2)

    # effective potential
    V =  .5*(x**2 + y**2) + (1-mu)/r1 + (mu)/r2

    #Vx
    vy = (2*(H + V) - vx**2)**(1/2)

    return vy

def jac(H):
    return -2*H


#%% Frequency Analysis

def fit_fn(t, state_var_vec):
    x, vx, y, vy = state_var_vec
    raw_sig = x
    sig = raw_sig - raw_sig.mean()  # removes dc component (shift)

    ff = fftfreq(len(t), t[1]-t[0]) # assumes uniform time spacing, computes
     bins for frequency

    Fyy = abs(fft(sig))

    #plot the fft amplitude vs frequency
    fig = plt.figure()
    ax = fig.add_subplot()
    ax.plot(np.abs(ff), np.abs(Fyy))
```

```
    ax.set(xlabel='Frequency', ylabel='FFT Amplitude')

    f_idx = np.argmax(Fyy)
    f = ff[f_idx]
    w = f*np.pi*2
    A = np.sqrt(np.max(Fyy))
    p = np.angle(fft(sig)[f_idx])   #phase shift angle

    fps = 1/(t[1]-t[0])  # Hz


    guess_freq = abs(ff[np.argmax(Fyy[1:])+1])      # excluding the zero
    frequency "peak", which is related to offset
    guess_amp = np.std(sig) * 2.**0.5               # gives spectral density
    guess_offset = np.mean(sig)
    guess = np.array([guess_amp, .2, 2.*np.pi*guess_freq, 0., guess_offset])

    def sinfunc(t, A, k, w, p, c):  return A * np.exp(-k*t)*np.cos(w*t + p)
    + c
    popt, pcov = curve_fit(sinfunc, t, sig, p0=guess)
    A, k, w, p, c = popt
    f = w/(2.*np.pi)
    fitfunc = lambda t: A * np.exp(-k*t) * np.cos(w*t + p) + c
    expfunc = lambda t: A * np.exp(-k*t)

    return {"amp": A, "omega": w, 'exponent': k, "phase": p, "offset": c, "
    freq": f, "period": 1./f, "fitfunc": fitfunc, "expfunc": expfunc, "
    maxcov": np.max(pcov), "rawres": (guess,popt,pcov)}



#%% plotting

def plot_const_vx(t, state_var_vec, H, ax1, col):
    h_val = round(H.mean(),4)

    x, vx, y, vy = state_var_vec

    ax1.plot(x, y, color=col, label = "H: {}".format(h_val))
    ax1.plot(-rs/R, 0, marker="o", color='yellow', markersize=5)
    ax1.plot(rj/R, 0, marker="o", color='orange', markersize=3)
    ax1.set(xlabel='x', ylabel='y')
    ax1.legend()
    ax1.set_aspect('equal')

    fig, (ax1) = plt.subplots(1,1)
    ax1.plot(x, y, label = "H: {}".format(h_val))
    ax1.plot(-rs/R, 0, marker="o", color='yellow', markersize=5)
    ax1.plot(rj/R, 0, marker="o", color='orange', markersize=3)
    ax1.set(xlabel='x', ylabel='y', title='Trajectory for Particle Bound by
    Jupiter')
    ax1.set_aspect('equal')
```

```
    #ig.savefig('x_vs_y_h{}_heavy_jupiter_near_sun.png'.format(h_val), dpi
=300)

    xmin, xmax = (-2, 2) #ax1.get_xlim()
    ymin, ymax = (-2, 2) #ax1.get_ylim()

    xs = np.linspace(xmin, xmax, 1000)
    ys = np.linspace(ymin, ymax, 1000)
    x1, y1 = np.meshgrid(xs, ys)

    vx1 = np.zeros_like(x1)
    vy1 = np.zeros_like(x1)
    vec = [x1, vx1, y1, vy1]

    poss_hs = [h_val]#np.linspace(h_val-.5, h_val+.5, 5)

    for h in poss_hs:
        diff = h*np.ones_like(Veff(vec, mu)) - Veff(vec, mu)
        masked_array = diff>0
        plot_array = masked_array*1

        x_derivs, y_derivs = np.gradient(plot_array)
        hill_line_image = (np.abs(x_derivs)+np.abs(y_derivs)) != 0

        #ax1.imshow(-plot_array, cmap = 'Greys', extent=(xmin, xmax, ymin,
ymax), alpha=.3, interpolation=None)
        ax1.imshow(hill_line_image, cmap = 'Reds', extent=(xmin, xmax, ymin,
 ymax), interpolation=None)

    fig.tight_layout()
    fig.savefig('test.png'.format(h_val), dpi=300)

    # fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2,2)
    # ax3.plot(t,x)
    # ax3.set(xlabel='t', ylabel='x')
    # ax4.plot(t,y)
    # ax4.set(xlabel='t', ylabel='y')
    # #fig.suptitle('Displacement vs Time')

    # ax1.plot(x,vx)
    # ax1.set(xlabel='x', ylabel='vx')
    # ax2.plot(y,vy)
    # ax2.set(xlabel='y', ylabel='vy')
    # #fig.suptitle('Phase Space')

    # fig.tight_layout()
    # fig.savefig('phase_space_h{}_heavy_jupiter_near_sun.png'.format(h_val)
, dpi=300)

    # model = fit_fn(t, state_var_vec)
    # period = model['period']
    # print(round(period,3))
```

```
271
272     #hamiltonian
273     ## fig, (ax1) = plt.subplots(1,1)
274     # ax6.plot(t, H, 'b', label = "H")
275     # ax6.set(xlabel='t', ylabel='H')
276     # ax6.set_ylim([H.min()-.5, H.max()+.5])
277     ## fig.suptitle('Hamiltonian: {}'.format(h_val))
278
279
280
281
282 #%% Set values
283 #velocity of J is 1 in our units
284
285 # given G = 1 now, find what omega should be.
286
287 x00 = (rj/2)/R
288 w = (G*M/(R**3))**(1/2)
289 w_test = (G*M/((x00*R)**3))**(1/2)
290
291 # x0 = [.567]#np.linspace(.5,.65,10)
292 # vx0 = 0
293 # y0 = 0
294 # vy0 = x00*(w_test-w)/w# 0
295
296 # # # values from the paper!
297 x0 = [.9]
298 vx0 = .215#.092
299 y0 = 0
300 vy0 = 0.14163 #x00*(w_test-w)/w# 0
301
302 # defining time
303 t_start = 0
304 t_stop = 600 #radians
305 num_points = 2000
306 time_vec = [t_start, t_stop, num_points]
307
308 #%% run it!
309
310 #initializing plot
311 fig, (ax1) = plt.subplots(1,1)
312 colorvals = np.linspace(0,1,len(x0))
313 b_colorvals = np.hstack((colorvals[len(colorvals)//2+1:],colorvals[:len(
        colorvals)//2+1]))
314
315 #plotting for varied x0,H!
316 for idx in range(len(x0)):
317     x0_val = x0[idx]
318     init_cond_vec = [x0_val, vx0, y0, vy0]
319     t, state_var_vec = solve_odes(mu, time_vec, init_cond_vec)
320     H = ham(state_var_vec, mu)
321
```

```
    col = (colorvals[idx], colorvals[-idx], b_colorvals[idx])
    plot_const_vx(t, state_var_vec, H, ax1, col)
    print('H:', H.mean())



    # model = fit_fn(t, state_var_vec)
    # freq = model['freq']
    # print("Freq: {}".format(round(freq,3)))


# 1st, circle, ellipse, werid one (.5ish), bean, spiral
# initial conditions from the paper

#%% Initial conditions for constant Hamiltonian analysis!

H = -1.583

x0 = np.linspace(.5,.6,5)
vx0 = 0
y0 = 0
vy0 = x00*(w_test-w)/w# 0

# defining time
time_vec = [t_start, t_stop, num_points]
t_start = 0
t_stop = 10 #radians
num_points = 10000

#initializing plot
fig, (ax1) = plt.subplots(1,1)
fig2, (ax8, ax9) = plt.subplots(1,2)
colorvals = np.linspace(0,1,len(x0))
b_colorvals = np.hstack((colorvals[len(colorvals)//2+1:],colorvals[:len(
    colorvals)//2+1]))

#plotting for constant H!
for idx in range(len(x0)):
    x0_val = x0[idx]
    H_val = -1.583
    vy0 = H_to_vy(x0_val,y0,vx0,H_val, mu)

    init_cond_vec = [x0_val, vx0, y0, vy0]
    t, state_var_vec = solve_odes(mu, time_vec, init_cond_vec)
    H = ham(state_var_vec, mu)

    col = (colorvals[idx], colorvals[-idx], b_colorvals[idx])
    plot_results(t, state_var_vec, H, ax1, col)
    print('H:', H.mean())

    x, vx, y, vy = state_var_vec

```

```
    ax8.plot(x, vx, color=col, label = "x0: {}".format(x0_val))
    ax8.set(xlabel='x', ylabel='vx')
    ax8.legend()

    ax9.plot(y, vy, color=col, label = "x0: {}".format(x0_val))
    ax9.set(xlabel='y', ylabel='vy')
    ax9.legend()

    # model = fit_fn(t, state_var_vec)
    # freq = model['freq']
    # print("Freq: {}".format(round(freq,3)))


#%% Jacobian stuff

#C = jac(H)

# hill's region: where U < E
# Hill's surfaces: where U = E, so velocity --> 0



# for x and y in list of x and y's
    # caluclate Veff
    # calculate H
    # plot where H = Veff
    # 0 and 128


#%% Hamiltonian!

H = ham(x, y, vx, vy, mu)
print('These should be the same:')
print('H min:', H.min())
print('H max:', H.max())


#%% fft (from lab 2)



#%%

model = fit_fn(t, state_var_vec)
period = model['period']
print(round(period,3))



# %%
```